Junos® OS

Routing Policies, Firewall Filters, and Traffic Policers Feature Guide

Modified: 2017-08-22

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

END USER LICENSE AGREEMENT

Table of Contents

List of Figures

List of Tables

About the Documentation

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at http://www.juniper.net/techpubs/.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at http://www.juniper.net/books.

Supported Platforms

For the features described in this document, the following platforms are supported:

- M Series
- MX Series
- PTX Series
- T Series

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming configuration into the current candidate configuration. The example does not become active until you commit the candidate configuration.

If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

If the example configuration does not start at the top level of the hierarchy, the example is a *snippet*. In this case, use the **load merge relative** command. These procedures are described in the following sections.

Merging a Full Example

To merge a full example, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration example into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following configuration to a file and name the file **ex-script.conf**. Copy the **ex-script.conf** file to the **/var/tmp** directory on your routing platform.

```
system {
  scripts {
    commit {
      file ex-script.xsl;
    }
  }
}
interfaces {
  fxp0 {
    disable;
    unit 0 {
      family inet {
        address 10.0.0.1/24;
      }
    }
  }
}
```

2. Merge the contents of the file into your routing platform configuration by issuing the **load merge** configuration mode command:

```
[edit]
user@host# load merge /var/tmp/ex-script.conf
load complete
```

Merging a Snippet

To merge a snippet, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration snippet into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following snippet to a file and name the file **ex-script-snippet.conf**. Copy the **ex-script-snippet.conf** file to the **/var/tmp** directory on your routing platform.

```
commit {
```

```
file ex-script-snippet.xsl; }
```

2. Move to the hierarchy level that is relevant for this snippet by issuing the following configuration mode command:

```
[edit]
user@host# edit system scripts
[edit system scripts]
```

3. Merge the contents of the file into your routing platform configuration by issuing the **load merge relative** configuration mode command:

```
[edit system scripts]
user@host# load merge relative /var/tmp/ex-script-snippet.conf
load complete
```

For more information about the **load** command, see CLI Explorer.

Documentation Conventions

Table 1 on page xxxiii defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.
	Tip	Indicates helpful information.
	Best practice	Alerts you to a recommended use or implementation.

Table 2 on page xxxiv defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

<table>
<tr><th>Convention</th><th>Description</th><th>Examples</th></tr>
<tr><td>Bold text like this</td><td>Represents text that you type.</td><td>To enter configuration mode, type the configure command:

user@host> configure</td></tr>
<tr><td><code>Fixed-width text like this</code></td><td>Represents output that appears on the terminal screen.</td><td><code>user@host></code> show chassis alarms

<code>No alarms currently active</code></td></tr>
<tr><td>Italic text like this</td><td>• Introduces or emphasizes important new terms.
• Identifies guide names.
• Identifies RFC and Internet draft titles.</td><td>• A policy term is a named structure that defines match conditions and actions.
• Junos OS CLI User Guide
• RFC 1997, BGP Communities Attribute</td></tr>
<tr><td>Italic text like this</td><td>Represents variables (options for which you substitute a value) in commands or configuration statements.</td><td>Configure the machine’s domain name:

[edit]
root@# set system domain-name
  domain-name</td></tr>
<tr><td>Text like this</td><td>Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.</td><td>• To configure a stub area, include the stub statement at the [edit protocols ospf area area-id] hierarchy level.
• The console port is labeled CONSOLE.</td></tr>
<tr><td>< > (angle brackets)</td><td>Encloses optional keywords or variables.</td><td>stub <default-metric <i>metric</i>>;</td></tr>
<tr><td>| (pipe symbol)</td><td>Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.</td><td>broadcast | multicast

(string1 | string2 | string3)</td></tr>
<tr><td># (pound sign)</td><td>Indicates a comment specified on the same line as the configuration statement to which it applies.</td><td>rsvp { # Required for dynamic MPLS only</td></tr>
<tr><td>[ ] (square brackets)</td><td>Encloses a variable for which you can substitute one or more values.</td><td>community name members [community-ids]</td></tr>
<tr><td>Indention and braces ({ })</td><td>Identifies a level in the configuration hierarchy.</td><td rowspan="2">[edit]
routing-options {
  static {
    route default {
      nexthop address;
      retain;
    }
  }
}</td></tr>
<tr><td>; (semicolon)</td><td>Identifies a leaf statement at a configuration hierarchy level.</td></tr>
<tr><td colspan="3">GUI Conventions</td></tr>
</table>

Table 2: Text and Syntax Conventions *(continued)*

Convention	Description	Examples
Bold text like this	Represents graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can provide feedback by using either of the following methods:

- Online feedback rating system—On any page of the Juniper Networks TechLibrary site at http://www.juniper.net/techpubs/index.html, simply click the stars to rate the content, and use the pop-up form to provide us with information about your experience. Alternately, you can use the online feedback form at http://www.juniper.net/techpubs/feedback/.
- E-mail—Send your comments to techpubs-comments@juniper.net. Include the document or topic name, URL or page number, and software version (if applicable).

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or Partner Support Service support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf.
- Product warranties—For product warranty information, visit http://www.juniper.net/support/warranty/.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: http://www.juniper.net/customers/support/
- Search for known bugs: http://www2.juniper.net/kb/
- Find product documentation: http://www.juniper.net/techpubs/
- Find solutions and answer questions using our Knowledge Base: http://kb.juniper.net/
- Download the latest versions of software and review release notes: http://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: http://kb.juniper.net/InfoCenter/
- Join and participate in the Juniper Networks Community Forum: http://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: http://www.juniper.net/cm/

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: https://tools.juniper.net/SerialNumberEntitlementSearch/

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at http://www.juniper.net/cm/.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see http://www.juniper.net/support/requesting-support.html.

PART 1

Overview

CHAPTER 1

Understanding the Junos OS Policy Framework

Policy Framework Overview

The Junos® operating system (Junos OS) provides a *policy framework*, which is a collection of Junos OS policies that allows you to control flows of routing information and packets.

The Junos OS policy architecture is simple and straightforward. However, the actual implementation of each policy adds layers of complexity to the policy as well as adding power and flexibility to your router's capabilities. Configuring a policy has a major impact on the flow of routing information or packets within and through the router. For example, you can configure a routing policy that does not allow routes associated with a particular customer to be placed in the routing table. As a result of this routing policy, the customer routes are not used to forward data packets to various destinations and the routes are not advertised by the routing protocol to neighbors.

Before configuring a policy, determine what you want to accomplish with it and thoroughly understand how to achieve your goal using the various match conditions and actions. Also, make certain that you understand the default policies and actions for the policy you are configuring.

Routing Policy and Firewall Filters

The policy framework is composed of the following policies:

- Routing policy—Allows you to control the routing information between the routing protocols and the routing tables and between the routing tables and the forwarding table. All routing protocols use the Junos OS routing tables to store the routes that they learn and to determine which routes they should advertise in their protocol packets. Routing policy allows you to control which routes the routing protocols store in and retrieve from the routing table.
- Firewall filter policy—Allows you to control packets transiting the router to a network destination and packets destined for and sent by the router.

NOTE: The term *firewall filter policy* is used here to emphasize that a firewall filter is a policy and shares some fundamental similarities with a routing policy. However, when referring to a firewall filter policy in the rest of this manual, the term *firewall filter* is used.

Reasons to Create a Routing Policy

The following are typical circumstances under which you might want to preempt the default routing policies in the routing policy framework by creating your own routing policies:

- You do not want a protocol to import all routes into the routing table. If the routing table does not learn about certain routes, they can never be used to forward packets and they can never be redistributed into other routing protocols.
- You do not want a routing protocol to export all the active routes it learns.
- You want a routing protocol to announce active routes learned from another routing protocol, which is sometimes called *route redistribution*.
- You want to manipulate route characteristics, such as the preference value, AS path, or community. You can manipulate the route characteristics to control which route is selected as the active route to reach a destination. In general, the active route is also advertised to a router's neighbors.
- You want to change the default BGP route flap-damping parameters.
- You want to perform per-packet load balancing.
- You want to enable class of service (CoS).

Router Flows Affected by Policies

The Junos OS policies affect the following router flows:

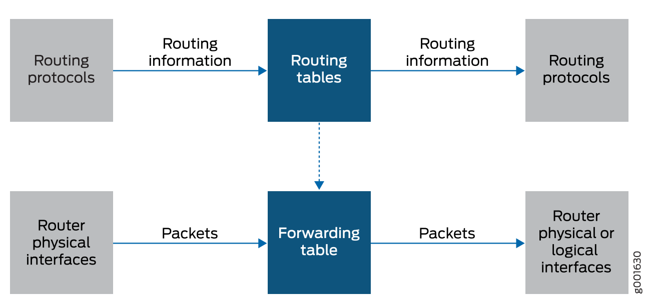
- Flow of routing information between the routing protocols and the routing tables and between the routing tables and the forwarding table. The Routing Engine handles this flow. *Routing information* is the information about routes learned by the routing protocols from a router's neighbors. This information is stored in routing tables and is subsequently advertised by the routing protocols to the router's neighbors. Routing policies allow you to control the flow of this information.

- Flow of data packets in and out of the router's physical interfaces. The Packet Forwarding Engine handles this flow. *Data packets* are chunks of data that transit the router as they are being forwarded from a source to a destination. When a router receives a data packet on an interface, it determines where to forward the packet by looking in the forwarding table for the best route to a destination. The router then forwards the data packet toward its destination through the appropriate interface. Firewall filters allow you to control the flow of these data packets.
- Flow of local packets from the router's physical interfaces and to the Routing Engine. The Routing Engine handles this flow. *Local packets* are chunks of data that are destined for or sent by the router. Local packets usually contain routing protocol data, data for IP services such as Telnet or SSH, and data for administrative protocols such as the Internet Control Message Protocol (ICMP). When the Routing Engine receives a local packet, it forwards the packet to the appropriate process or to the kernel, which are both part of the Routing Engine, or to the Packet Forwarding Engine. Firewall filters allow you to control the flow of these local packets.

NOTE: In the rest of this chapter, the term *packets* refers to both data and local packets unless explicitly stated otherwise.

Figure 1 on page 5 illustrates the flows through the router. Although the flows are very different from each other, they are also interdependent. Routing policies determine which routes are placed in the forwarding table. The forwarding table, in turn, has an integral role in determining the appropriate physical interface through which to forward a packet.

Figure 1: Flows of Routing Information and Packets

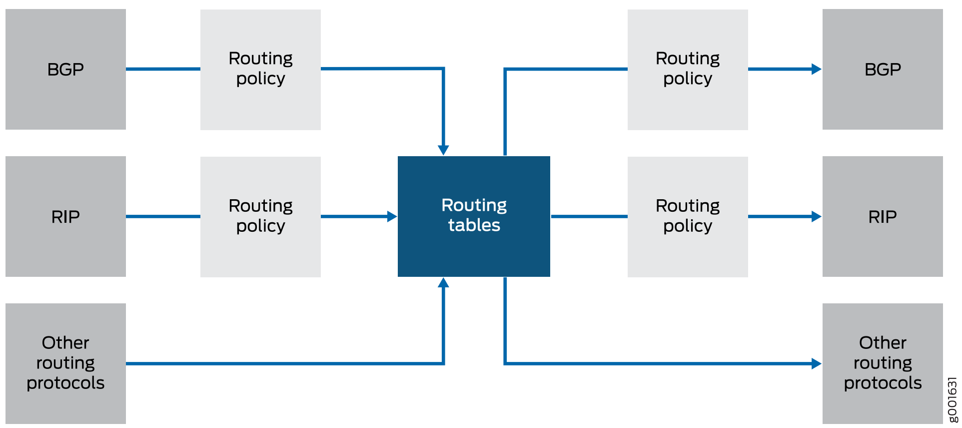


You can configure routing policies to control which routes the routing protocols place in the routing tables and to control which routes the routing protocols advertise from the routing tables (see Figure 2 on page 6). The routing protocols advertise active routes only from the routing tables. (An *active route* is a route that is chosen from all routes in the routing table to reach a destination.)

You can also use routing policies to do the following:

- Change specific route characteristics, which allow you to control which route is selected as the active route to reach a destination. In general, the active route is also advertised to a router's neighbors.
- Change to the default BGP route flap-damping values.
- Perform per-packet load balancing.
- Enable class of service (CoS).

Figure 2: Routing Policies to Control Routing Information Flow

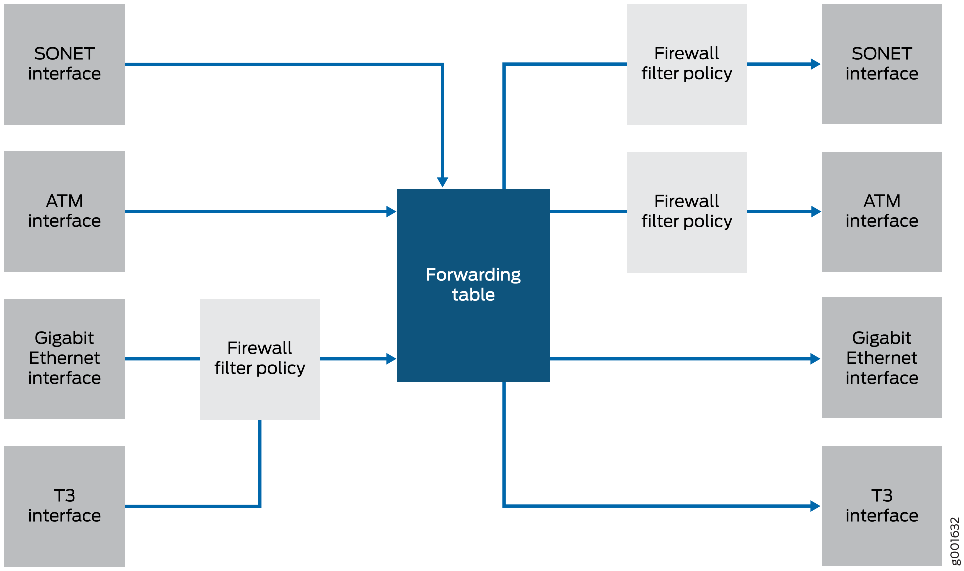


You can configure firewall filters to control the following aspects of packet flow (see Figure 3 on page 7):

- Which data packets are accepted on and transmitted from the physical interfaces. To control the flow of data packets, you apply firewall filters to the physical interfaces.
- Which local packets are transmitted from the physical interfaces and to the Routing Engine. To control local packets, you apply firewall filters on the loopback interface, which is the interface to the Routing Engine.

Firewall filters provide a means of protecting your router from excessive traffic transiting the router to a network destination or destined for the Routing Engine. Firewall filters that control local packets can also protect your router from external incidents such as denial-of-service attacks.

Figure 3: Firewall Filters to Control Packet Flow

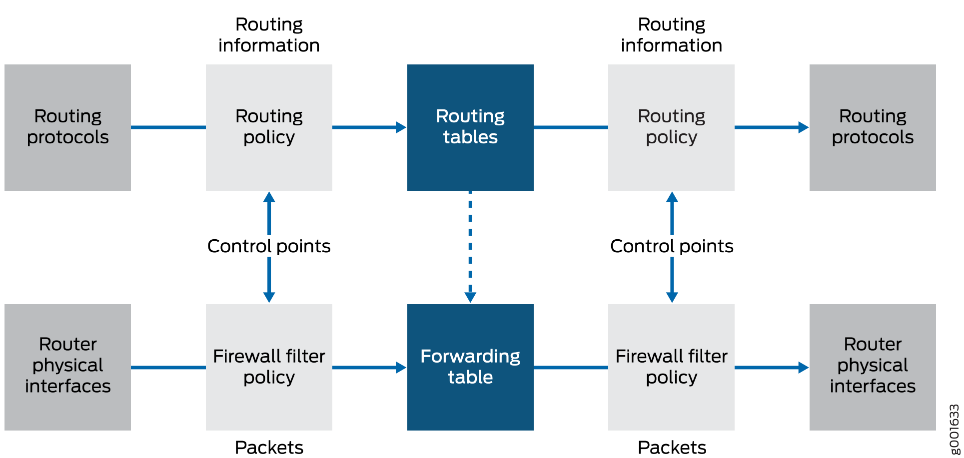


Control Points

All policies provide two points at which you can control routing information or packets through the router (see Figure 4 on page 7). These control points allow you to control the following:

- Routing information before and after it is placed in the routing table.
- Data packets before and after a forwarding table lookup.
- Local packets before and after they are received by the Routing Engine. (Figure 4 on page 7 appears to depict only one control point but because of the bidirectional flow of the local packets, two control points actually exist.)

Figure 4: Policy Control Points



Because there are two control points, you can configure policies that control the routing information or data packets before and after their interaction with their respective tables, and policies that control local packets before and after their interaction with the Routing Engine. *Import routing policies* control the routing information that is placed in the routing tables, whereas *export routing policies* control the routing information that is advertised from the routing tables. *Input firewall filters* control packets that are received on a router interface, whereas *output firewall filters* control packets that are transmitted from a router interface.

Policy Components

All policies are composed of the following components that you configure:

- *Match conditions*—Criteria against which a route or packets are compared. You can configure one or more criteria. If all criteria match, one or more actions are applied.
- *Actions*—What happens if all criteria match. You can configure one or more actions.
- *Terms*—Named structures in which match conditions and actions are defined. You can define one or more terms.

The policy framework software evaluates each incoming and outgoing route or packet against the match conditions in a term. If the criteria in the match conditions are met, the defined action is taken.

In general, the policy framework software compares the route or packet against the match conditions in the first term in the policy, then goes on to the next term, and so on. Therefore, the order in which you arrange terms in a policy is relevant.

The order of match conditions within a term is not relevant because a route or packet must match all match conditions in a term for an action to be taken.

Related Documentation

- Comparison of Routing Policies and Firewall Filters on page 8
- *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*

Comparison of Routing Policies and Firewall Filters

Although routing policies and firewall filters share an architecture, their purposes, implementation, and configuration are different. Table 3 on page 8 describes their purposes. Table 4 on page 9 compares the implementation details for routing policies and firewall filters, highlighting the similarities and differences in their configuration.

Table 3: Purpose of Routing Policies and Firewall Filters

Policies	Source	Policy Purpose
Routing policies	Routing information is generated by internal networking peers.	To control the size and content of the routing tables, which routes are advertised, and which routes are considered the best to reach various destinations.
Firewall filters	Packets are generated by internal and external devices through which hostile attacks can be perpetrated.	To protect your router and network from excessive incoming traffic or hostile attacks that can disrupt network service, and to control which packets are forwarded from which router interfaces.

Table 4: Implementation Differences Between Routing Policies and Firewall Filters

<table>
<tr><th>Policy Architecture</th><th>Routing Policy Implementation</th><th>Firewall Filter Implementation</th></tr>
<tr><td>Control points</td><td>Control routing information that is placed in the routing table with an import routing policy and advertised from the routing table with an export routing policy.</td><td>Control packets that are accepted on a router interface with an input firewall filter and that are forwarded from an interface with an output firewall filter.</td></tr>
<tr><td>Configuration tasks:
Define policy
Apply policy</td><td>Define a policy that contains terms, match conditions, and actions.

Apply one or more export or import policies to a routing protocol. You can also apply a policy expression, which uses Boolean logical operators with multiple import or export policies.

You can also apply one or more export policies to the forwarding table.</td><td>Define a policy that contains terms, match conditions, and actions.

Apply one input or output firewall filter to a physical interface or physical interface group to filter data packets received by or forwarded to a physical interface (on routing platforms with an Internet Processor II application-specific integrated circuit [ASIC] only).

You can also apply one input or output firewall filter to the routing platform's loopback interface, which is the interface to the Routing Engine (on all routing platforms). This allows you to filter local packets received by or forwarded from the Routing Engine.</td></tr>
<tr><td>Terms</td><td>Configure as many terms as desired. Define a name for each term.

Terms are evaluated in the order in which you specify them.

Evaluation of a policy ends after a packet matches the criteria in a term and the defined or default policy action of accept or reject is taken. The route is not evaluated against subsequent terms in the same policy or subsequent policies.</td><td>Configure as many terms as desired. Define a name for each term.

Terms are evaluated in the order in which you specify them.

Evaluation of a firewall filter ends after a packet matches the criteria in a term and the defined or default action is taken. The packet is not evaluated against subsequent terms in the firewall filter.</td></tr>
<tr><td>Match conditions</td><td>Specify zero or more criteria that a route must match. You can specify criteria based on source, destination, or properties of a route. You can also specify the following match conditions, which require more configuration:
Autonomous system (AS) path expression—A combination of AS numbers and regular expression operators.
Community—A group of destinations that share a common property.
Prefix list—A named list of prefixes.
Route list—A list of destination prefixes.
Subroutine—A routing policy that is called repeatedly from other routing policies.</td><td>Specify zero or more criteria that a packet must match. You must match various fields in the packet's header. The fields are grouped into the following categories:
Numeric values, such as port and protocol numbers.
Prefix values, such as IP source and destination prefixes.
Bit-field values—Whether particular bits in the fields are set, such as IP options, Transmission Control Protocol (TCP) flags, and IP fragmentation fields. You can specify the fields using Boolean logical operators.</td></tr>
</table>

Table 4: Implementation Differences Between Routing Policies and Firewall Filters *(continued)*

Policy Architecture	Routing Policy Implementation	Firewall Filter Implementation
Actions	Specify zero or one action to take if a route matches all criteria. You can specify the following actions: • Accept—Accept the route into the routing table, and propagate it. After this action is taken, the evaluation of subsequent terms and policies ends. • Reject—Do not accept the route into the routing table, and do not propagate it. After this action is taken, the evaluation of subsequent terms and policies ends. In addition to the preceding actions, you can also specify zero or more of the following types of actions: • Next term—Evaluate the next term in the routing policy. • Next policy—Evaluate the next routing policy. • Actions that manipulate characteristics associated with a route as the routing protocol places it in the routing table or advertises it from the routing table. • Trace action, which logs route matches.	Specify zero or one action to take if a packet matches all criteria. (We recommend that you always explicitly configure an action.) You can specify the following actions: • Accept—Accept a packet. • Discard—Discard a packet silently, without sending an ICMP message. • Reject—Discard a packet, and send an ICMP destination unreachable message. • Routing instance—Specify a routing table to which packets are forwarded. • Next term—Evaluate the next term in the firewall filter. In addition to zero or the preceding actions, you can also specify zero or more action modifiers. You can specify the following action modifiers: • Count—Add packet to a count total. • Forwarding class—Set the packet forwarding class to a specified value from 0 through 3. • IPsec security association—Used with the source and destination address match conditions, specify an IP Security (IPsec) security association (SA) for the packet. • Log—Store the header information of a packet on the Routing Engine. • Loss priority—Set the packet loss priority (PLP) bit to a specified value, 0 or 1. • Policer—Apply rate-limiting procedures to the traffic. • Sample—Sample the packet traffic. • Syslog—Log an alert for the packet.

Table 4: Implementation Differences Between Routing Policies and Firewall Filters *(continued)*

Policy Architecture	Routing Policy Implementation	Firewall Filter Implementation

Default policies and actions	If an incoming or outgoing route arrives and a policy related to the route is not explicitly configured, the action specified by the default policy for the associated routing protocol is taken.
	
	The following default actions exist for routing policies:
	
	• If a policy does not specify a match condition, all routes evaluated against the policy match.
	• If a match occurs but the policy does not specify an accept, reject, next term, or next policy action, one of the following occurs:
	  ◦ The next term, if present, is evaluated.
	  ◦ If no other terms are present, the next policy is evaluated.
	  ◦ If no other policies are present, the action specified by the default policy is taken.
	• If a match does not occur with a term in a policy and subsequent terms in the same policy exist, the next term is evaluated.
	• If a match does not occur with any terms in a policy and subsequent policies exist, the next policy is evaluated.
	• If a match does not occur by the end of a policy and no other policies exist, the accept or reject action specified by the default policy is taken.
		If an incoming or outgoing packet arrives on an interface and a firewall filter is not configured for the interface, the default policy is taken (the packet is accepted).
		
		The following default actions exist for firewall filters:
		
		• If a firewall filter does not specify a match condition, all packets are considered to match.
		• If a match occurs but the firewall filter does not specify an action, the packet is accepted.
		• If a match occurs, the defined or default action is taken and the evaluation ends. Subsequent terms in the firewall filter are not evaluated, unless the next term action is specified.
		• If a match does not occur with a term in a firewall filter and subsequent terms in the same filter exist, the next term is evaluated.
		• If a match does not occur by the end of a firewall filter, the packet is discarded.

Related Documentation

- Policy Framework Overview on page 3
- *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*

Prefix Prioritization Overview

Junos OS routes have a predetermined order for route installation. This is no longer sufficient as it is sometimes required to prioritize certain routes or prefixes over others for better convergence or to provide differentiated services. In a network with a large number of routes, it is sometimes important to control the order in which routes get updated due to a change in the network topology. For example, it would be useful to first update OSPF routes that correspond to an IBGP neighbor, and only then update the rest of the OSPF routes. At a system level, Junos OS implements reasonable defaults based on heuristics to determine the order in which routes get updated. However, the default behavior is not always optimal. Prefix prioritization allows the user to control the order in which the routes get updated from LDP or OSPF to rpd, and from rpd to kernel. In Junos OS Release 16.1 and later, you can control the order in which the routes get updated from LDP or OSPF to rpd, and from rpd to kernel.

In Junos OS Release 16.1 and later, the Junos OS policy language is extended to let the user set the relative priority **high** and **low** for prefixes via the existing protocol import policy. Based on the tagged priority, the routes are placed in different priority queues. Routes that are not explicitly assigned a priority are treated as medium priority. Within the same priority level, routes will continue to be updated in lexicographic order. Prefix prioritization would need each supporting protocol to prioritize its routes internally. Prefix prioritization ensures that high priority IGP and LDP routes get updated to the FIB (forwarding table) before medium and low priority routes.

NOTE: There is an upper limit on how many high priority prefixes are allowed in the kernel. Not more than 10,000 high priority prefixes can coexist in the kernel. If this threshold is crossed in the kernel, then any new high priority prefix addition request will be considered as normal priority. This is a "best effort" prioritization scheme and will not be handled if the kernel is highly loaded.

Related Documentation

- Example: Configuring the Priority for Route Prefixes in the RPD Infrastructure on page 293
- Configuring Priority for Route Prefixes in RPD Infrastructure on page 304

Accounting of the Policer Overhead Attribute at the Interface Level

A bandwidth profile (BWP) enforces limits on bandwidth utilization according to the service level specification (SLS) that has been agreed upon by the subscriber and the service provider as part of the service level agreement (SLA).There are two types of bandwidth profiles:

- Ingress Bandwidth Profile
- Egress Bandwidth Profile

Need for Policer Overhead adjustment

The Metro Ethernet Forum (MEF) Carrier Ethernet (CE) 2.0 set of standards has stringent requirements for the bandwidth profile enforcement at the user network interface (UNI). MEF CE 2.0 certification compliance allows only a 2 percent deviation from UNI commited or peak rate across all frame sizes. This means that the policers should take into account the frame size at the UNI interface, including frame checksum and excluding all additional overheads that might be added inside the service provider network (such as S-VLANs). Therefore, this translates into two customer requirements:

- Junos OS egress policers use frame length before output VLAN manipulation. If VLANs are added on output, those extra bytes will not be counted. In order to address MEF CE 2.0 requirements, adjust the length of the packet that is used for policing purposes for Junos egress policers that use frame length before output VLAN manipulation. Therefore, if VLANs are added on output, the extra bytes will not be counted.
- In some network designs, bandwidth profile enforcement is implemented at the Layer 2 (L2) VPN Provider Edge Router and not at the Ethernet access device (EAD) terminating the physical UNI interface. The EAD typically adds an S-VLAN that identifies the port in the access network. The S-VLAN that is added is considered internal to the service provider network and typically should not be taken into account for bandwidth profile enforcement purposes at the Provider Edge device in both ingress and egress directions. This also translates into a requirement to allow adjusting the packet length used for policing purposes on ingress and egress.

In order to address these requirements, **policer-overhead** adjustment is defined on a per logical interface (IFL)/direction granularity, which is the range of -16 bytes to +16 bytes. The **policer-overhead** adjustment is applied for all the policers that take into account Layer 1 (L1) or L2 packet length that are exercised in the specified IFL/direction, including corresponding logical interface family (IFF) feature policers, and is applied only to interface or filter-based policers.

Policer Overhead Adjustment Applicability for Policers

The ingress or egress **policer overhead** adjustment configuration is applicable for the following types of policers on ingress or egress IFL and IFF, respectively:

- L2 two-color and three-color policers.
- IFL-level policers (configured at the IFL or in a filter attached to IFL).
- Family-level policers that use L2 packet length, or policers in filters attached to L2 IFF (family bridge, vpls, ccc).

NOTE: The list is applicable for all types of policers, including regular two-color policers, three-color policers, and hierarchical policers, provided that the policer operates on an L1 or L2 packet length.

Ingress policer overhead adjustment configuration is applied to any policers attached to ingress L2 routing instances.

NOTE: Note that any IFF policer on the L3 family (inet inet6), which considers only L3 packet length, will not consider this adjustment. The **policer overhead** adjustment value (+ve or -ve) is added to the actual L2 packet length to obtain the number of bytes to charge the policer. Therefore, this is used only as an intermediate value, and does not affect actual L2 packet length. This feature is applied before VLAN normalization, and is independent of the **egress-shaping-overhead** or **ingress-shaping-overhead** configuration under class of service.

Related Documentation

- policer-overhead-adjustment on page 1336
- *Configuring the Accounting of Policer Overhead in Interface Statistics*

PART 2

Configuring Routing Policies

CHAPTER 2

Understanding How Routing Policies Control Routing Information and Packet Flows

Understanding Routing Policies

For some routing platform vendors, the flow of routes occurs between various protocols. If, for example, you want to configure redistribution from RIP to OSPF, the RIP process tells the OSPF process that it has routes that might be included for redistribution. In Junos OS, there is not much direct interaction between the routing protocols. Instead, there are central gathering points where all protocols install their routing information. These are the main unicast routing tables inet.0 and inet6.0.

From these tables, the routing protocols calculate the best route to each destination and place these routes in a forwarding table. These routes are then used to forward routing protocol traffic toward a destination, and they can be advertised to neighbors.

Importing and Exporting Routes

Two terms—*import* and *export*—explain how routes move between the routing protocols and the routing table.

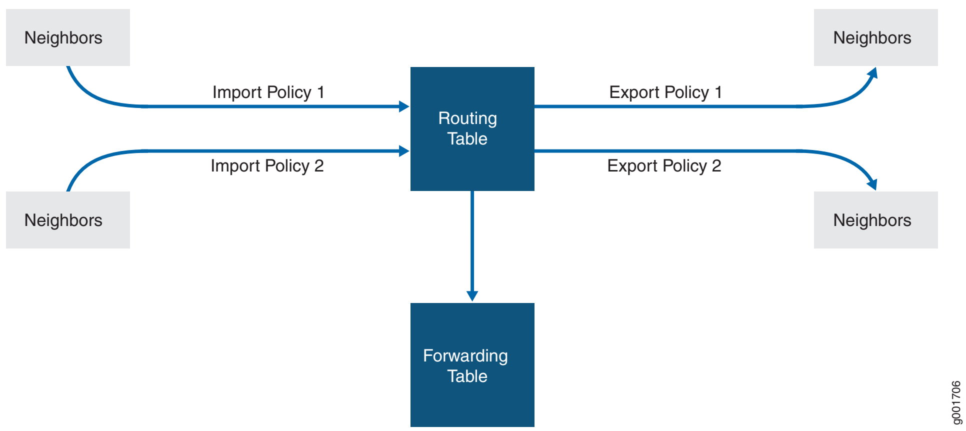
- When the Routing Engine places the routes of a routing protocol into the routing table, it is *importing* routes into the routing table.

- When the Routing Engine uses active routes from the routing table to send a protocol advertisement, it is *exporting* routes from the routing table.

NOTE: **The process of moving routes between a routing protocol and the routing table is described always *from the point of view of the routing table*. That is, routes are *imported into* a routing table from a routing protocol and they are *exported from* a routing table to a routing protocol. Remember this distinction when working with routing policies.**

As shown in Figure 5 on page 18, you use import routing policies to control which routes are placed in the routing table, and export routing policies to control which routes are advertised from the routing table to neighbors.

Figure 5: Importing and Exporting Routes



In general, the routing protocols place all their routes in the routing table and advertise a limited set of routes from the routing table. The general rules for handling the routing information between the routing protocols and the routing table are known as the *routing policy framework*.

The routing policy framework is composed of default rules for each routing protocol that determine which routes the protocol places in the routing table and advertises from the routing table. The default rules for each routing protocol are known as *default routing policies*.

You can create routing policies to preempt the default policies, which are always present. A *routing policy* allows you to modify the routing policy framework to suit your needs. You can create and implement your own routing policies to do the following:

- Control which routes a routing protocol places in the routing table.
- Control which active routes a routing protocol advertises from the routing table. An *active route* is a route that is chosen from all routes in the routing table to reach a destination.
- Manipulate the route characteristics as a routing protocol places the route in the routing table or advertises the route from the routing table.

You can manipulate the route characteristics to control which route is selected as the active route to reach a destination. The active route is placed in the forwarding table and is used to forward traffic toward the route's destination. In general, the active route is also advertised to a router's neighbors.

Active and Inactive Routes

When multiple routes for a destination exist in the routing table, the protocol selects an active route and that route is placed in the appropriate routing table. For equal-cost routes, the Junos OS places multiple next hops in the appropriate routing table.

When a protocol is exporting routes from the routing table, it exports active routes only. This applies to actions specified by both default and user-defined export policies.

When evaluating routes for export, the Routing Engine uses only active routes from the routing table. For example, if a routing table contains multiple routes to the same destination and one route has a preferable metric, only that route is evaluated. In other words, an export policy does not evaluate all routes; it evaluates only those routes that a routing protocol is allowed to advertise to a neighbor.

NOTE: By default, BGP advertises active routes. However, you can configure BGP to advertise *inactive routes*, which go to the same destination as other routes but have less preferable metrics.

Explicitly Configured Routes

An *explicitly configured route* is a route that you have configured. *Direct routes* are not explicitly configured. They are created as a result of IP addresses being configured on an interface. Explicitly configured routes include aggregate, generated, local, and static routes. (An *aggregate route* is a route that distills groups of routes with common addresses into one route. A *generated route* is a route used when the routing table has no information about how to reach a particular destination. A *local route* is an IP address assigned to a router interface. A *static route* is an unchanging route to a destination.)

The policy framework software treats direct and explicitly configured routes as if they are learned through routing protocols; therefore, they can be imported into the routing table. Routes cannot be exported from the routing table to the pseudoprotocol, because this protocol is not a real routing protocol. However, aggregate, direct, generated, and static routes can be exported from the routing table to routing protocols, whereas local routes cannot.

Dynamic Database

In Junos OS Release 9.5 and later, you can configure routing policies and certain routing policy objects in a dynamic database that is not subject to the same verification required by the standard configuration database. As a result, you can quickly commit these routing policies and policy objects, which can be referenced and applied in the standard configuration as needed. BGP is the only protocol to which you can apply routing policies that reference policies configured in the dynamic database. After a routing policy based on the dynamic database is configured and committed in the standard configuration, you can quickly make changes to existing routing policies by modifying policy objects in the dynamic database. Because Junos OS does not validate configuration changes to the dynamic database, when you use this feature, you should test and verify all configuration changes before committing them.

Related Documentation

- Example: Configuring Dynamic Routing Policies on page 527
- *Example: Redistributing OSPF Routes into IS-IS*

Protocol Support for Import and Export Policies

Table 5: Protocol Support for Import and Export Policies

Protocol	Import Policy	Export Policy	Supported Levels
BGP	Yes	Yes	Import: global, group, peer Export: global, group, peer
DVMRP	Yes	Yes	Global
IS-IS	No	Yes	Export: global
LDP	Yes	Yes	Global
MPLS	No	No	–
OSPF	Yes	Yes	Export: global Import: external routes only
PIM dense mode	Yes	Yes	Global
PIM sparse mode	Yes	Yes	Global
Pseudoprotocol—Explicitly configured routes, which include the following: • Aggregate routes • Generated routes	Yes	No	Import: global
RIP and RIPng	Yes	Yes	Import: global, neighbor Export: group

Example: Applying Routing Policies at Different Levels of the BGP Hierarchy

This example shows BGP configured in a simple network topology and explains how routing polices take effect when they are applied at different levels of the BGP configuration.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

For BGP, you can apply policies as follows:

- BGP global **import** and **export** statements—Include these statements at the **[edit protocols bgp]** hierarchy level (for routing instances, include these statements at the **[edit routing-instances *routing-instance-name* protocols bgp]** hierarchy level).
- Group **import** and **export** statements—Include these statements at the **[edit protocols bgp group *group-name*]** hierarchy level (for routing instances, include these statements at the **[edit routing-instances *routing-instance-name* protocols bgp group *group-name*]** hierarchy level).
- Peer **import** and **export** statements—Include these statements at the **[edit protocols bgp group *group-name* neighbor *address*]** hierarchy level (for routing instances, include these statements at the **[edit routing-instances *routing-instance-name* protocols bgp group *group-name* neighbor *address*]** hierarchy level).

A peer-level **import** or **export** statement overrides a group **import** or **export** statement. A group-level **import** or **export** statement overrides a global BGP **import** or **export** statement.

In this example, a policy named **send-direct** is applied at the global level, another policy named **send-192.168.0.1** is applied at the group level, and a third policy named **send-192.168.20.1** is applied at the neighbor level.

```
user@host# show protocols
bgp {
  local-address 172.16.1.1;
  export send-direct;
  group internal-peers {
    type internal;
    export send-192.168.0.1;
    neighbor 172.16.2.2 {
      export send-192.168.20.1;
    }
    neighbor 172.16.3.3;
```



```
    }
    group other-group {
        type internal;
        neighbor 172.16.4.4;
    }
}
```

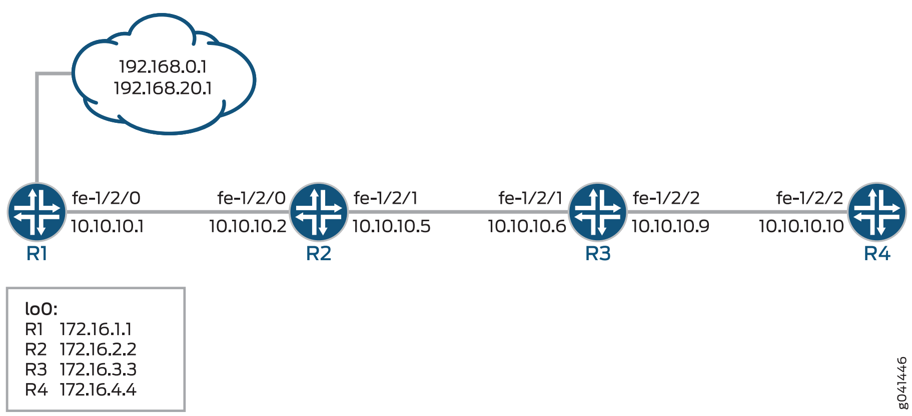
A key point, and one that is often misunderstood and that can lead to problems, is that in such a configuration, only the most explicit policy is applied. A neighbor-level policy is more explicit than a group-level policy, which in turn is more explicit than a global policy.

The neighbor 172.16.2.2 is subjected only to the send-192.168.20.1 policy. The neighbor 172.16.3.3, lacking anything more specific, is subjected only to the send-192.168.0.1 policy. Meanwhile, neighbor 172.16.4.4 in group other-group has no group or neighbor-level policy, so it uses the send-direct policy.

If you need to have neighbor 172.16.2.2 perform the function of all three policies, you can write and apply a new neighbor-level policy that encompasses the functions of the other three, or you can apply all three existing policies, as a chain, to neighbor 172.16.2.2.

Figure 6 on page 22 shows the sample network.

Figure 6: Applying Routing Policies to BGP



"CLI Quick Configuration" on page 22 shows the configuration for all of the devices in Figure 6 on page 22.

The section "Step-by-Step Procedure" on page 24 describes the steps on Device R1.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 description to-R2
set interfaces fe-1/2/0 unit 0 family inet address 10.10.10.1/30
set interfaces lo0 unit 0 family inet address 172.16.1.1/32
```

```
set protocols bgp local-address 172.16.1.1
set protocols bgp export send-direct
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers export send-static-192.168.0
set protocols bgp group internal-peers neighbor 172.16.2.2 export send-static-192.168.20
set protocols bgp group internal-peers neighbor 172.16.3.3
set protocols bgp group other-group type internal
set protocols bgp group other-group neighbor 172.16.4.4
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static-192.168.0 term 1 from protocol static
set policy-options policy-statement send-static-192.168.0 term 1 from route-filter
  192.168.0.0/24 orlonger
set policy-options policy-statement send-static-192.168.0 term 1 then accept
set policy-options policy-statement send-static-192.168.20 term 1 from protocol static
set policy-options policy-statement send-static-192.168.20 term 1 from route-filter
  192.168.20.0/24 orlonger
set policy-options policy-statement send-static-192.168.20 term 1 then accept
set routing-options static route 192.168.0.1/32 discard
set routing-options static route 192.168.20.1/32 discard
set routing-options router-id 172.16.1.1
set routing-options autonomous-system 17
```

Device R2

```
set interfaces fe-1/2/0 unit 0 description to-R1
set interfaces fe-1/2/0 unit 0 family inet address 10.10.10.2/30
set interfaces fe-1/2/1 unit 0 description to-R3
set interfaces fe-1/2/1 unit 0 family inet address 10.10.10.5/30
set interfaces lo0 unit 0 family inet address 172.16.2.2/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 172.16.2.2
set protocols bgp group internal-peers neighbor 172.16.3.3
set protocols bgp group internal-peers neighbor 172.16.1.1
set protocols bgp group internal-peers neighbor 172.16.4.4
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set routing-options router-id 172.16.2.2
set routing-options autonomous-system 17
```

Device R3

```
set interfaces fe-1/2/1 unit 0 description to-R2
set interfaces fe-1/2/1 unit 0 family inet address 10.10.10.6/30
set interfaces fe-1/2/2 unit 0 description to-R4
set interfaces fe-1/2/2 unit 0 family inet address 10.10.10.9/30
set interfaces lo0 unit 0 family inet address 172.16.3.3/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 172.16.3.3
set protocols bgp group internal-peers neighbor 172.16.2.2
set protocols bgp group internal-peers neighbor 172.16.1.1
set protocols bgp group internal-peers neighbor 172.16.4.4
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
```

```
set routing-options router-id 172.16.3.3
set routing-options autonomous-system 17
```

Device R4

```
set interfaces fe-1/2/2 unit 0 description to-R3
set interfaces fe-1/2/2 unit 0 family inet address 10.10.10.10/30
set interfaces lo0 unit 0 family inet address 172.16.4.4/32
set protocols bgp group internal-peers type internal
set protocols bgp group internal-peers local-address 172.16.4.4
set protocols bgp group internal-peers neighbor 172.16.2.2
set protocols bgp group internal-peers neighbor 172.16.1.1
set protocols bgp group internal-peers neighbor 172.16.3.3
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set routing-options router-id 172.16.4.4
set routing-options autonomous-system 17
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure an IS-IS default route policy:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 0 description to-R2
user@R1# set fe-1/2/0 unit 0 family inet address 10.10.10.1/30

user@R1# set lo0 unit 0 family inet address 172.16.1.1/32
```

2. Enable OSPF, or another interior gateway protocols (IGP), on the interfaces.

```
[edit protocols OSPF area 0.0.0.0]
user@R1# set interface lo0.0 passive
user@R1# set interface fe-1/2/0.0
```

3. Configure static routes.

```
[edit routing-options]
user@R1# set static route 192.168.0.1/32 discard
user@R1# set static route 192.168.20.1/32 discard
```

4. Enable the routing policies.

```
[edit protocols policy-options]
user@R1# set policy-statement send-direct term 1 from protocol direct
user@R1# set policy-statement send-direct term 1 then accept

user@R1# set policy-statement send-static-192.168.0 term 1 from protocol static
user@R1# set policy-statement send-static-192.168.0 term 1 from route-filter
  192.168.0.0/24 orlonger
user@R1# set policy-statement send-static-192.168.0 term 1 then accept
```



```
user@R1# set policy-statement send-static-192.168.20 term 1 from protocol static
user@R1# set policy-statement send-static-192.168.20 term 1 from route-filter
  192.168.20.0/24 orlonger
user@R1# set policy-statement send-static-192.168.20 term 1 then accept
```

5. Configure BGP and apply the export policies.

```
[edit protocols bgp]
user@R1# set local-address 172.16.1.1
user@R1# set group internal-peers type internal
user@R1# set group internal-peers export send-static-192.168.0
user@R1# set group internal-peers neighbor 172.16.2.2 export send-static-192.168.20
user@R1# set group internal-peers neighbor 172.16.3.3
user@R1# set group other-group type internal
user@R1# set group other-group neighbor 172.16.4.4
```

6. Configure the router ID and autonomous system (AS) number.

```
[edit routing-options]
user@R1# set router-id 172.16.1.1
user@R1# set autonomous-system 17
```

7. If you are done configuring the device, commit the configuration.

```
[edit]
user@R1# commit
```

Results

From configuration mode, confirm your configuration by issuing the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
fe-1/2/0 {
  unit 0 {
    description to-R2;
    family inet {
      address 10.10.10.1/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 172.16.1.1/32;
    }
  }
}

user@R1# show protocols
```

```
bgp {
   local-address 172.16.1.1;
   export send-direct;
   group internal-peers {
      type internal;
      export send-static-192.168.0;
      neighbor 172.16.2.2 {
         export send-static-192.168.20;
      }
      neighbor 172.16.3.3;
   }
   group other-group {
      type internal;
      neighbor 172.16.4.4;
   }
}
ospf {
   area 0.0.0.0 {
      interface lo0.0 {
         passive;
      }
      interface fe-1/2/0.0;
   }
}

user@R1# show policy-options
policy-statement send-direct {
   term 1 {
      from protocol direct;
      then accept;
   }
}
policy-statement send-static-192.168.0 {
   term 1 {
      from {
         protocol static;
         route-filter 192.168.0.0/24 orlonger;
      }
      then accept;
   }
}
policy-statement send-static-192.168.20 {
   term 1 {
      from {
         protocol static;
         route-filter 192.168.20.0/24 orlonger;
      }
      then accept;
   }
}

user@R1# show routing-options
static {
   route 192.168.0.1/32 discard;
   route 192.168.20.1/32 discard;
}
router-id 172.16.1.1;
```



```
autonomous-system 17;
```

Verification

Confirm that the configuration is working properly.

- Verifying BGP Route Learning on page 27
- Verifying BGP Route Receiving on page 29

Verifying BGP Route Learning

Purpose Make sure that the BGP export policies are working as expected by checking the routing tables.

Action

```
user@R1> show route protocol direct

inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.1/32         *[Direct/0] 1d 22:19:47
                    > via lo0.0
10.10.10.0/30      *[Direct/0] 1d 22:19:47
                    > via fe-1/2/0.0
```

```
user@R1> show route protocol static

inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.1/32     *[Static/5] 02:20:03
                      Discard
192.168.20.1/32    *[Static/5] 02:20:03
                      Discard
```

```
user@R2> show route protocol bgp
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.20.1/32    *[BGP/170] 02:02:40, localpref 100, from 172.16.1.1
                      AS path: I, validation-state: unverified
                    > to 10.10.10.1 via fe-1/2/0.0
```

```
user@R3> show route protocol bgp
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.0.1/32     *[BGP/170] 02:02:51, localpref 100, from 172.16.1.1
                      AS path: I, validation-state: unverified
                    > to 10.10.10.5 via fe-1/2/1.0
```

```
user@R4> show route protocol bgp
inet.0: 9 destinations, 11 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.1/32          [BGP/170] 1d 20:38:54, localpref 100, from 172.16.1.1
                      AS path: I, validation-state: unverified
                    > to 10.10.10.9 via fe-1/2/2.0
10.10.10.0/30       [BGP/170] 1d 20:38:54, localpref 100, from 172.16.1.1
                      AS path: I, validation-state: unverified
                    > to 10.10.10.9 via fe-1/2/2.0
```

Meaning

On Device R1, the **show route protocol direct** command displays two direct routes: 172.16.1.1/32 and 10.10.10.0/30. The **show route protocol static** command displays two static routes: 192.168.0.1/32 and 192.168.20.1/32.

On Device R2, the **show route protocol bgp** command shows that the only route that Device R2 has learned through BGP is the 192.168.20.1/32 route.

On Device R3, the **show route protocol bgp** command shows that the only route that Device R3 has learned through BGP is the 192.168.0.1/32 route.

On Device R4, the **show route protocol bgp** command shows that the only routes that Device R4 has learned through BGP are the 172.16.1.1/32 and 10.10.10.0/30 routes.

Verifying BGP Route Receiving

Purpose Make sure that the BGP export policies are working as expected by checking the BGP routes received from Device R1.

Action `user@R2>` **show route receive-protocol bgp 172.16.1.1**

```
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 192.168.20.1/32         172.16.1.1                      100        I
```

`user@R3>` **show route receive-protocol bgp 172.16.1.1**

```
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 192.168.0.1/32          172.16.1.1                      100        I
```

`user@R4>` **show route receive-protocol bgp 172.16.1.1**

```
inet.0: 9 destinations, 11 routes (9 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.1.1/32              172.16.1.1                      100        I
  10.10.10.0/30           172.16.1.1                      100        I
```

Meaning On Device R2, the **route receive-protocol bgp 172.16.1.1** command shows that Device R2 received only one BGP route, 192.168.20.1/32, from Device R1.

On Device R3, the **route receive-protocol bgp 172.16.1.1** command shows that Device R3 received only one BGP route, 192.168.0.1/32, from Device R1.

On Device R4, the **route receive-protocol bgp 172.16.1.1** command shows that Device R4 received two BGP routes, 172.16.1.1/32 and 10.10.10.0/30, from Device R1.

In summary, when multiple policies are applied at different CLI hierarchies in BGP, only the most specific application is evaluated, to the exclusion of other, less specific policy applications. Although this point might seem to make sense, it is easily forgotten during router configuration, when you mistakenly believe that a neighbor-level policy is combined with a global or group-level policy, only to find that your policy behavior is not as anticipated.

Related Documentation

- Example: Configuring Policy Chains and Route Filters on page 186
- Example: Configuring a Policy Subroutine on page 203

- Example: Configuring Routing Policy Prefix Lists on page 282
- export on page 1185
- import on page 1197

Default Routing Policies

If an incoming or outgoing route or packet arrives and there is no explicitly configured policy related to the route or to the interface upon which the packet arrives, the action specified by the default policy is taken. A *default policy* is a rule or a set of rules that determine whether the route is placed in or advertised from the routing table, or whether the packet is accepted into or transmitted from the router interface.

You must be familiar with the default routing policies to know when you need to modify them to suit your needs. Table 6 on page 30 summarizes the default routing policies for each routing protocol that imports and exports routes. The actions in the default routing policies are taken if you have not explicitly configured a routing policy. This table also shows direct and explicitly configured routes, which for the purposes of this table are considered a pseudoprotocol. Explicitly configured routes include aggregate, generated, and static routes.

Table 6: Default Import and Export Policies for Protocols

Importing or Exporting Protocol	Default Import Policy	Default Export Policy
BGP	Accept all received BGP IPv4 routes learned from configured neighbors and import into the inet.0 routing table. Accept all received BGP IPv6 routes learned from configured neighbors and import into the inet6.0 routing table.	Readvertise all active BGP routes to all BGP speakers, while following protocol-specific rules that prohibit one IBGP speaker from readvertising routes learned from another IBGP speaker, unless it is functioning as a route reflector.
DVMRP	Accept all DVMRP routes and import into the inet.1 routing table.	Accept and export active DVMRP routes.
IS-IS	Accept all IS-IS routes and import into the inet.0 and inet6.0 routing tables. (You cannot override or change this default policy.)	Reject everything. (The protocol uses flooding to announce local routes and any learned routes.)
LDP	Accept all LDP routes and import into the inet.3 routing table.	Reject everything.
MPLS	Accept all MPLS routes and import into the inet.3 routing table.	Accept and export active MPLS routes.
OSPF	Accept all OSPF routes and import into the inet.0 routing table. (You cannot override or change this default policy.)	Reject everything. (The protocol uses flooding to announce local routes and any learned routes.)

Table 6: Default Import and Export Policies for Protocols *(continued)*

Importing or Exporting Protocol	Default Import Policy	Default Export Policy
PIM dense mode	Accept all PIM dense mode routes and import into the inet.1 routing table.	Accept active PIM dense mode routes.
PIM sparse mode	Accept all PIM sparse mode routes and import into the inet.1 routing table.	Accept and export active PIM sparse mode routes.
Pseudoprotocol: • Direct routes • Explicitly configured routes: ◦ Aggregate routes ◦ Generated routes ◦ Static routes	Accept all direct and explicitly configured routes and import into the inet.0 routing table.	The pseudoprotocol cannot export any routes from the routing table because it is not a routing protocol. Routing protocols can export these or any routes from the routing table.
RIP	Accept all RIP routes learned from configured neighbors and import into the inet.0 routing table.	Reject everything. To export RIP routes, you must configure an export policy for RIP.
RIPng	Accept all RIPng routes learned from configured neighbors and import into the inet6.0 routing table.	Reject everything. To export RIPng routes, you must configure an export policy for RIPng.
Test policy	Accept all routes. For additional information about test policy, see “Example: Testing a Routing Policy with Complex Regular Expressions” on page 544.	

OSPF and IS-IS Import Policies

You cannot change the default import policy for IS-IS. For OSPF, import policies apply to external routes only. An external route is a route that is outside the OSPF autonomous system (AS). For internal routes (routes learned from OSPF), you cannot change the default import policy for OSPF. As link-state protocols, IS-IS and OSPF exchange routes between systems within an autonomous system (AS). All routers and systems within an AS must share the same link-state database, which includes routes to reachable prefixes and the metrics associated with the prefixes. If an import policy is configured and applied to IS-IS or OSPF, some routes might not be learned or advertised or the metrics for learned routes might be altered, which would make a consistent link-state database impossible.

The default export policy for IS-IS and OSPF protocols is to reject everything. These protocols do not actually export their internally learned routes (the directly connected routes on interfaces that are running the protocol). Both IS-IS and OSPF protocols use a procedure called flooding to announce local routes and any routes learned by the protocol. The flooding procedure is internal to the protocol, and is unaffected by the policy framework. Exporting can be used only to announce information from other protocols, and the default is not to do so.

Automatic Export

For Layer 3 VPNs, the automatic export feature can be configured to overcome the limitation of local prefix leaking and automatically export routes between local VPN routing and forwarding (VRF) routing instances.

In Layer 3 VPNs, multiple CE routers can belong to a single VRF routing instance on a PE router. A PE router can have multiple VRF routing instances. In some cases, shared services might require routes to be written to multiple VRF routing tables, both at the local and remote PE router. This requires the PE router to share route information among each configured VRF routing instance. This exchange of route information is accomplished with custom **vrf-export** and **vrf-import** policies that utilize BGP extended community attributes to create hub-and-spoke topologies. This exchange of routing information, such as route prefixes, is known as prefix leaking.

The automatic export feature leaks prefixes between VRF routing instances that are locally configured on a given PE router. The automatic export feature is enabled by using the **auto-export** statement.

Automatic export is always applied on the local PE router, because it takes care of only local prefix leaking by evaluating the export policy of each VRF and determining which route targets can be leaked locally. The standard VRF import and export policies still affect only the remote PE prefix leaking.

If the **vrf-export** policy examined by the automatic export does not have an explicit **then accept** action, the automatic export essentially ignores the policy and, therefore, does not leak the route targets specified within it.

For more information, see Technology Overview: Understanding the Auto Export Feature.

Related Documentation

- Protocol Support for Import and Export Policies on page 20
- Technology Overview: Understanding the Auto Export Feature.

Example: Configuring a Conditional Default Route Policy

This example shows how to configure a conditional default route on one routing device and redistribute the default route into OSPF.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, OSPF area 0 contains three routing devices. Device R3 has a BGP session with an external peer, for example, an Internet Service Provider (ISP).

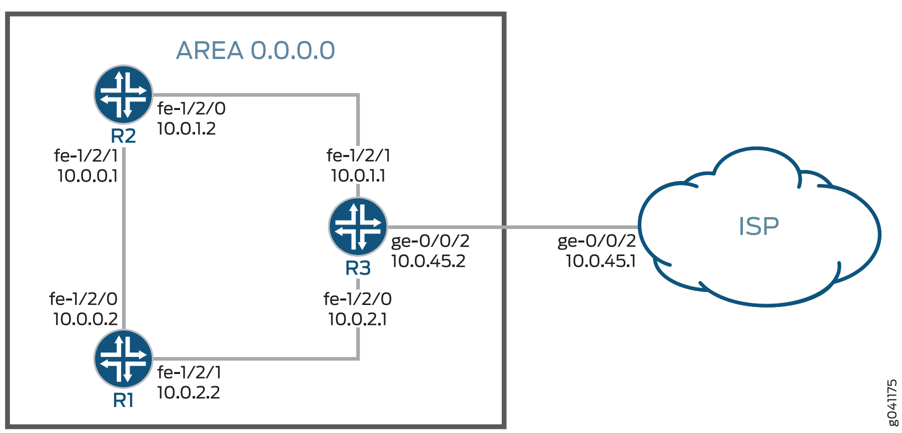
To propagate a static route into BGP, this example includes the **discard** statement when defining the route. The ISP injects a default static route into BGP, which provides the customer network with a default static route to reach external networks. The static route has a discard next hop. This means that if a packet does not match a more specific route, the packet is rejected and a reject route for this destination is installed in the routing table, but Internet Control Message Protocol (ICMP) unreachable messages are not sent. The discard next hop allows you to originate a summary route, which can be advertised through dynamic routing protocols.

Device R3 exports the default route into OSPF. The route policy on Device R3 is conditional such that if the connection to the ISP goes down, the default route is no longer exported into OSPF because it is no longer active in the routing table. This policy prevents packets from being silently dropped without notification (also known as blackholing).

This example shows the configuration for all of the devices and the step-by-step configuration on Device R3.

Figure 7 on page 33 shows the sample network.

Figure 7: OSPF with a Conditional Default Route to an ISP



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 description R1->R3
set interfaces fe-1/2/0 unit 0 family inet address 10.0.1.2/30
set interfaces fe-1/2/1 unit 2 description R1->R2
set interfaces fe-1/2/1 unit 2 family inet address 10.0.0.1/30
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.2
```

Device R2

```
set interfaces fe-1/2/0 unit 1 description R2->R1
set interfaces fe-1/2/0 unit 1 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 4 description R2->R3
set interfaces fe-1/2/1 unit 4 family inet address 10.0.2.2/30
set protocols ospf area 0.0.0.0 interface fe-1/2/0.1
set protocols ospf area 0.0.0.0 interface fe-1/2/1.4
```

Device R3

```
set interfaces fe-1/2/0 unit 3 description R3->R2
set interfaces fe-1/2/0 unit 3 family inet address 10.0.2.1/30
set interfaces fe-1/2/1 unit 5 description R3->R1
set interfaces fe-1/2/1 unit 5 family inet address 10.0.1.1/30
set interfaces ge-0/0/2 unit 0 description R3->ISP
set interfaces ge-0/0/2 unit 0 family inet address 10.0.45.2/30
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64500
set protocols bgp group ext neighbor 10.0.45.1
set protocols ospf export gendefault
set protocols ospf area 0.0.0.0 interface fe-1/2/1.4
set protocols ospf area 0.0.0.0 interface fe-1/2/0.3
set policy-options policy-statement gendefault term upstreamroutes from protocol bgp
set policy-options policy-statement gendefault term upstreamroutes from as-path
  upstream
set policy-options policy-statement gendefault term upstreamroutes from route-filter
  0.0.0.0/0 upto /16
set policy-options policy-statement gendefault term upstreamroutes then next-hop
  10.0.45.1
set policy-options policy-statement gendefault term upstreamroutes then accept
set policy-options policy-statement gendefault term end then reject
set policy-options as-path upstream "^64500 "
set routing-options autonomous-system 64501
```

Device ISP

```
set interfaces ge-0/0/2 unit 0 family inet address 10.0.45.1/30
set protocols bgp group ext type external
set protocols bgp group ext export advertise-default
set protocols bgp group ext peer-as 64501
set protocols bgp group ext neighbor 10.0.45.2
set policy-options policy-statement advertise-default term 1 from route-filter 0.0.0.0/0
  exact
set policy-options policy-statement advertise-default term 1 then accept
set routing-options static route 0.0.0.0/0 discard
set routing-options autonomous-system 64500
```


Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R3:

1. Configure the interfaces.

   ```
   [edit interfaces]
   user@R3# set fe-1/2/0 unit 3 description R3->R2
   user@R3# set fe-1/2/0 unit 3 family inet address 10.0.2.1/30
   user@R3# set fe-1/2/1 unit 5 description R3->R1
   user@R3# set fe-1/2/1 unit 5 family inet address 10.0.1.1/30
   user@R3# set ge-0/0/2 unit 0 description R3->ISP
   user@R3# set ge-0/0/2 unit 0 family inet address 10.0.45.2/30
   ```

2. Configure the autonomous system (AS) number.

   ```
   [edit routing-options]
   user@R3# set autonomous-system 64501
   ```

3. Configure the BGP session with the ISP device.

   ```
   [edit protocols bgp group ext]
   user@R3# set type external
   user@R3# set peer-as 64500
   user@R3# set neighbor 10.0.45.1
   ```

4. Configure OSPF.

   ```
   [edit protocols ospf area 0.0.0.0]
   user@R3# set interface fe-1/2/1.4
   user@R3# set interface fe-1/2/0.3
   ```

5. Configure the routing policy.

   ```
   [edit policy-options policy-statement gendefault]
   user@R3# set term upstreamroutes from protocol bgp
   user@R3# set term upstreamroutes from as-path upstream
   user@R3# set term upstreamroutes from route-filter 0.0.0.0/0 upto /16
   user@R3# set term upstreamroutes then next-hop 10.0.45.1
   user@R3# set term upstreamroutes then accept

   user@R3# set term end then reject

   [edit policy-options]
   user@R3# set as-path upstream "^64500 "
   ```

6. Apply the export policy to OSPF.

   ```
   [edit protocols ospf]
   user@R3# set export gendefault
   ```


7. If you are done configuring the device, commit the configuration.

```
[edit]
user@R3# commit
```

Results

Confirm your configuration by issuing the **show** command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R3# show
interfaces {
   fe-1/2/0 {
      unit 3 {
         description R3->R2;
         family inet {
            address 10.0.2.1/30;
         }
      }
   }
   fe-1/2/1 {
      unit 5 {
         description R3->R1;
         family inet {
            address 10.0.1.1/30;
         }
      }
   }
   ge-1/2/0 {
      unit 0 {
         description R3->ISP;
         family inet {
            address 10.0.45.2/30;
         }
      }
   }
}
protocols {
   bgp {
      group ext {
         type external;
         peer-as 64500;
         neighbor 10.0.45.1;
      }
   }
   ospf {
      export gendefault;
      area 0.0.0.0 {
         interface fe-1/2/1.4;
         interface fe-1/2/0.3;
      }
   }
}
```



```
policy-options {
  policy-statement gendefault {
    term upstreamroutes {
      from {
        protocol bgp;
        as-path upstream;
        route-filter 0.0.0.0/0 upto /16;
      }
      then {
        next-hop 10.0.45.1;
        accept;
      }
    }
    term end {
      then reject;
    }
  }
  as-path upstream "^64500 ";
}
routing-options {
  autonomous-system 64501;
}
```

Verification

Confirm that the configuration is working properly.

- Verifying That the Route to the ISP Is Working on page 37
- Verifying That the Static Route Is Redistributed on page 37
- Testing the Policy Condition on page 38

Verifying That the Route to the ISP Is Working

Purpose Make sure connectivity is established between Device R3 and the ISP's router.

Action

```
user@R3> ping 10.0.45.1
PING 10.0.45.1 (10.0.45.1): 56 data bytes
64 bytes from 10.0.45.1: icmp_seq=0 ttl=64 time=1.185 ms
64 bytes from 10.0.45.1: icmp_seq=1 ttl=64 time=1.199 ms
64 bytes from 10.0.45.1: icmp_seq=2 ttl=64 time=1.186 ms
```

Meaning The **ping** command confirms reachability.

Verifying That the Static Route Is Redistributed

Purpose Make sure that the BGP policy is redistributing the static route into Device R3's routing table. Also make sure that the OSPF policy is redistributing the static route into the routing tables of Device R1 and Device R2.

Action

```
user@R3> show route protocol bgp

inet.0: 9 destinations, 10 routes (9 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 00:00:25, localpref 100
                      AS path: 64500 I
                    > to 10.0.45.1 via ge-0/0/2.6
```

```
user@R1> show route protocol ospf

inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[OSPF/150] 00:03:58, metric 0, tag 0
                    > to 10.0.1.1 via fe-1/2/0.0
10.0.2.0/30        *[OSPF/10] 03:37:45, metric 2
                      to 10.0.1.1 via fe-1/2/0.0
                    > to 10.0.0.2 via fe-1/2/1.2
172.16.233.5/32       *[OSPF/10] 03:38:41, metric 1
                      MultiRecv
```

```
user@R2> show route protocol ospf
inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[OSPF/150] 00:04:04, metric 0, tag 0
                    > to 10.0.2.1 via fe-1/2/1.4
10.0.1.0/30        *[OSPF/10] 03:37:46, metric 2
                      to 10.0.0.1 via fe-1/2/0.1
                    > to 10.0.2.1 via fe-1/2/1.4
172.16.233.5/32       *[OSPF/10] 03:38:47, metric 1
                      MultiRecv
```

Meaning The routing tables contain the default 0.0.0.0/0 route. If Device R1 and Device R2 receive packets destined for networks not specified in their routing tables, those packets will be sent to Device R3 for further processing. If Device R3 receives packets destined for networks not specified in its routing table, those packets will be sent to the ISP for further processing.

Testing the Policy Condition

Purpose Deactivate the interface to make sure that the route is removed from the routing tables if the external network becomes unreachable.

Action

```
user@R3> deactivate interfaces ge-0/0/2 unit 0 family inet address 10.0.45.2/30
user@R3> commit

user@R1> show route protocol ospf

inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.2.0/30        *[OSPF/10] 03:41:48, metric 2
                      to 10.0.1.1 via fe-1/2/0.0
                    > to 10.0.0.2 via fe-1/2/1.2
172.16.233.5/32       *[OSPF/10] 03:42:44, metric 1
                      MultiRecv

user@R2> show route protocol ospf
inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.1.0/30        *[OSPF/10] 03:42:10, metric 2
                      to 10.0.0.1 via fe-1/2/0.1
                    > to 10.0.2.1 via fe-1/2/1.4
172.16.233.5/32       *[OSPF/10] 03:43:11, metric 1
                      MultiRecv
```

Meaning The routing tables on Device R1 and Device R2 do not contain the default 0.0.0.0/0 route. This verifies that the default route is no longer present in the OSPF domain. To reactivate the ge-0/0/2.6 interface, issue the **activate interfaces ge-0/0/2 unit 0 family inet address 10.0.45.2/30** configuration mode command.

CHAPTER 3

Evaluating Routing Policies Using Match Conditions, Actions, Terms, and Expressions

How a Routing Policy Is Evaluated

Figure 8 on page 42 shows how a single routing policy is evaluated. This routing policy consists of multiple terms. Each term consists of match conditions and actions to apply to matching routes. Each route is evaluated against the policy as follows:

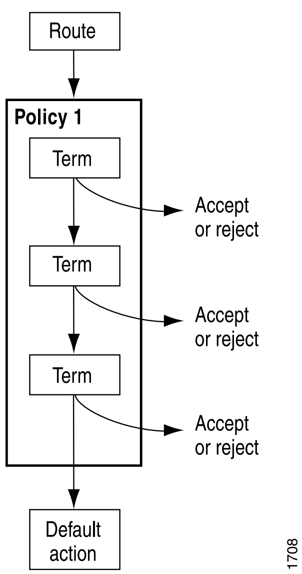
1. The route is evaluated against the first term. If it matches, the specified action is taken. If the action is to accept or reject the route, that action is taken and the evaluation of the route ends. If the next term action is specified, if no action is specified, or if the

route does not match, the evaluation continues as described in Step 2. If the next policy action is specified, any accept or reject action specified in this term is skipped, all remaining terms in this policy are skipped, all other actions are taken, and the evaluation continues as described in Step 3.

2. The route is evaluated against the second term. If it matches, the specified action is taken. If the action is to accept or reject the route, that action is taken and the evaluation of the route ends. If the next term action is specified, if no action is specified, or if the route does not match, the evaluation continues in a similar manner against the last term. If the next policy action is specified, any accept or reject action specified in this term is skipped, all remaining terms in this policy are skipped, all other actions are taken, and the evaluation continues as described in Step 3.

3. If the route matches no terms in the routing policy or the next policy action is specified, the accept or reject action specified by the default policy is taken. For more information about the default routing policies, see "Default Routing Policies" on page 30.

Figure 8: Routing Policy Evaluation



Categories of Routing Policy Match Conditions

A *match condition* defines the criteria that a route must match. You can define one or more match conditions. If a route matches all match conditions, one or more actions are applied to the route.

Match conditions fall into two categories: standard and extended. In general, the extended match conditions are more complex than standard match conditions. The extended match conditions provide many powerful capabilities. The standard match conditions include criteria that are defined within a routing policy and are less complex than the extended match conditions, also called named match conditions.

Extended match conditions are defined separately from the routing policy and are given names. You then reference the name of the match condition in the definition of the routing policy itself.

Named match conditions allow you to do the following:

- Reuse match conditions in other routing policies.
- Read configurations that include complex match conditions more easily.

Named match conditions include communities, prefix lists, and AS path regular expressions.

Table 7 on page 43 describes each match condition, including its category, when you typically use it, and any relevant notes about it. For more information about match conditions, see "Routing Policy Match Conditions" on page 44.

Table 7: Match Condition Concepts

Match Condition	Category	When to Use	Notes
AS path regular expression—A combination of AS numbers and regular expression operators.	Extended	(BGP only) Match a route based on its AS path. (An AS path consists of the AS numbers of all routers a packet must go through to reach a destination.) You can specify an exact match with a particular AS path or a less precise match.	You use regular expressions to match the AS path.
Community—A group of destinations that share a property. (Community information is included as a path attribute in BGP update messages.)	Extended	Match a group of destinations that share a property. Use a routing policy to define a community that specifies a group of destinations you want to match and one or more actions that you want taken on this community.	Actions can be performed on the entire group. You can create multiple communities associated with a particular destination. You can create match conditions using regular expressions.
Prefix list—A named list of IP addresses.	Extended	Match a route based on prefix information. You can specify an exact match of a particular route only.	You can specify a common action only for all prefixes in the list.
Route list—A list of destination prefixes.	Extended	Match a route based on prefix information. You can specify an exact match of a particular route or a less precise match.	You can specify an action for each prefix in the route list or a common action for all prefixes in the route list.

Table 7: Match Condition Concepts *(continued)*

Match Condition	Category	When to Use	Notes
Standard—A collection of criteria that can match a route.	Standard	Match a route based on one of the following criteria: area ID, color, external route, family, instance (routing), interface name, level number, local preference, metric, neighbor address, next-hop address, origin, preference, protocol, routing table name, or tag. You can specify a match condition for policies based on protocols by naming a protocol from which the route is learned or to which the route is being advertised.	None.
Subroutine—A routing policy that is called repeatedly from another routing policy.	Extended	Use an effective routing policy in other routing policies. You can create a subroutine that you can call over and over from other routing policies.	The subroutine action influences but does not necessarily determine the final action. For more information, see "How a Routing Policy Subroutine Is Evaluated" on page 201.

Each term can consist of two statements, **from** and **to**, that define match conditions:

- In the **from** statement, you define the criteria that an *incoming* route must match. You can specify one or more match conditions. If you specify more than one, all conditions must match the route for a match to occur.
- In the **to** statement, you define the criteria that an *outgoing* route must match. You can specify one or more match conditions. If you specify more than one, all conditions must match the route for a match to occur.

The order of match conditions in a term is not important, because a route must match all match conditions in a term for an action to be taken.

Related Documentation

- Routing Policy Match Conditions on page 44

Routing Policy Match Conditions

Each term in a routing policy can include two statements, **from** and **to**, to define the conditions that a route must match for the policy to apply:

```
from {
  family family-name;
  match-conditions;
  policy subroutine-policy-name;
  prefix-list name;
  route-filter destination-prefix match-type <actions>;
  source-address-filter source-prefix match-type <actions>;
}
to {
  match-conditions;
```



```
    policy subroutine-policy-name;
  }
```

In the **from** statement, you define the criteria that an incoming route must match. You can specify one or more match conditions. If you specify more than one, they all must match the route for a match to occur.

The **from** statement is optional. If you omit the **from**, all routes are considered to match. All routes then take the configured actions of the policy term.

In the **to** statement, you define the criteria that an outgoing route must match. You can specify one or more match conditions. If you specify more than one, they all must match the route for a match to occur. You can specify most of the same match conditions in the **to** statement that you can in the **from** statement. In most cases, specifying a match condition in the **to** statement produces the same result as specifying the same match condition in the **from** statement.

The **to** statement is optional. If you omit both the **to** and the **from** statements, all routes are considered to match.

Table 8 on page 45 summarizes key routing policy match conditions.

Table 8: Summary of Key Routing Policy Match Conditions

Match Condition	Description
aggregate-contributor	Matches routes that are contributing to a configured aggregate. This match condition can be used to suppress a contributor in an aggregate route.
area *area-id*	Matches a route learned from the specified OSPF area during the exporting of OSPF routes into other protocols.
as-path *name*	Matches the name of the path regular expression of an autonomous systems (AS). BGP routes whose AS path matches the regular expression are processed.
color *preference*	Matches a color value. You can specify preference values that are finer-grained than those specified in the ***preference*** match conditions. The **color** value can be a number from 0 through 4,294,967,295 (2^{32} – 1). A lower number indicates a more preferred route.
community	Matches the name of one or more communities. If you list more than one name, only one name needs to match for a match to occur. (The matching is effectively a logical OR operation.)
external [type *metric-type*]	Matches external OSPF routes, including routes exported from one level to another. In this match condition, **type** is an optional keyword. The **metric-type** value can be either 1 or 2. When you do not specify **type**, this condition matches all external routes.
interface *interface-name*	Matches the name or IP address of one or more router interfaces. Use this condition with protocols that are interface-specific. For example, do not use this condition with internal BGP (IBGP). Depending on where the policy is applied, this match condition matches routes learned from or advertised through the specified interface.

Table 8: Summary of Key Routing Policy Match Conditions *(continued)*

Match Condition	Description
internal	Matches a routing policy against the internal flag for simplified next-hop self policies.
level *level*	Matches the IS-IS level. Routes that are from the specified level or are being advertised to the specified level are processed.
local-preference *value*	Matches a BGP local preference attribute. The preference value can be from 0 through 4,294,967,295 (2^{32} – 1).
metric *metric* **metric2** *metric*	Matches a metric value. The **metric** value corresponds to the multiple exit discriminator (MED), and **metric2** corresponds to the IGP metric if the BGP next hop runs back through another route.
neighbor *address*	Matches the address of one or more neighbors (peers). For BGP export policies, the address can be for a directly connected or indirectly connected peer. For all other protocols, the address is for the neighbor from which the advertisement is received.
next-hop *address*	Matches the next-hop address or addresses specified in the routing information for a particular route. For BGP routes, matches are performed against each protocol next hop.
origin *value*	Matches the BGP origin attribute, which is the origin of the AS path information. The value can be one of the following: • **egp**—Path information originated from another AS. • **igp**—Path information originated from within the local AS. • **incomplete**—Path information was learned by some other means.
preference *preference* **preference2** *preference*	Matches the preference value. You can specify a primary preference value (**preference**) and a secondary preference value (**preference2**). The preference value can be a number from 0 through 4,294,967,295 (2^{32} – 1). A lower number indicates a more preferred route. NOTE: Do not set **preference2** for BGP route-policy.
protocol *protocol*	Matches the name of the protocol from which the route was learned or to which the route is being advertised. It can be one of the following: **aggregate**, **bgp**, **direct**, **dvmrp**, **isis**, **local**, **ospf**, **pim-dense**, **pim-sparse**, **rip**, **ripng**, or **static**.
route-type *value*	Matches the type of route. The value can be either **external** or **internal**.

All conditions in the **from** and **to** statements must match for the action to be taken. The match conditions defined in Table 9 on page 47 are effectively a logical AND operation. Matching in prefix lists and route lists is handled differently. They are effectively a logical OR operation. If you configure a policy that includes some combination of route filters, prefix lists, and source address filters, they are evaluated according to a logical OR operation or a longest-route match lookup.

Table 9 on page 47 describes the match conditions available for matching an incoming or outgoing route. The table indicates whether you can use the match condition in both **from** and **to** statements and whether the match condition functions the same or differently when used with both statements. If a match condition functions differently in a **from** statement than in a **to** statement, or if the condition cannot be used in one type of statement, there is a separate description for each type of statement. Otherwise, the same description applies to both types of statements.

Table 9 on page 47 also indicates whether the match condition is standard or extended. In general, the extended match conditions include criteria that are defined separately from the routing policy (autonomous system [AS] path regular expressions, communities, and prefix lists) and are more complex than standard match conditions. The extended match conditions provide many powerful capabilities. The standard match conditions include criteria that are defined within a routing policy and are less complex than the extended match conditions.

Table 9: Complete List of Routing Policy Match Conditions

Match Condition	Match Condition Category	from Statement Description	to Statement Description
aggregate-contributor	Standard	Match routes that are contributing to a configured aggregate. This match condition can be used to suppress a contributor in an aggregate route.	
area ***area-id***	Standard	(Open Shortest Path First [OSPF] only) Area identifier. In a **from** statement used with an **export** policy, match a route learned from the specified OSPF area when exporting OSPF routes into other protocols.	
as-path ***name***	Extended	(Border Gateway Protocol [BGP] only) Name of an AS path regular expression. For more information, see “Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions” on page 309.	
as-path-group ***group-name***	Extended	(BGP only) Name of an AS path group regular expression. For more information, see “Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions” on page 309.	
color ***preference*** **color2** ***preference***	Standard	Color value. You can specify preference values (**color** and **color2**) that are finer-grained than those specified in the **preference** and **preference2** match conditions. The color value can be a number in the range from 0 through 4,294,967,295 (2^{32} – 1). A lower number indicates a more preferred route.	

Table 9: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
community-count ***value*** **(equal | orhigher | orlower)**	Standard	(BGP only) Number of community entries required for a route to match. The count value can be a number in the range of 0 through 1,024. Specify one of the following options: • **equal**—The number of communities must equal this value to be considered a match. • **orhigher** —The number of communities must be greater than or equal to this value to be considered a match. • **orlower**—The number of communities must be less than or equal to this value to be considered a match. NOTE: If you configure multiple **community-count** statements, the matching is effectively a logical AND operation. NOTE: The **community-count** attribute only works with standard communities. It does not work with extended communities.	You cannot specify this match condition.
community [***names*** **]**	Extended	Name of one or more communities. If you list more than one name, only one name needs to match for a match to occur (the matching is effectively a logical OR operation). For more information, see "Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions" on page 359.	
external [type ***metric-type***]	Standard	(OSPF and IS-IS only) Match IGP external routes. For IS-IS routes, the **external** condition also matches routes that are exported from one IS-IS level to another. The **type** keyword is optional and is applicable only to OSPF external routes. When you do not specify **type**, the **external** condition matches all IGP external (OSPF and IS-IS) routes. When you specify **type**, the **external** condition matches only OSPF external routes with the specified OSPF metric type. The metric type can either be 1 or 2. To match BGP external routes, use the **route-type** match condition.	

Table 9: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
family ***family-name***	Standard	Name of an address family. Match the address family of the route. Depending on your device and configuration, ***family-name*** can be one of the following: • **inet**— IP version 4 (IPv4) traffic • **inet-mdt**—IPv4 multicast distribution tree (MDT) traffic • **inet-mvpn**—IPv4 multicast virtual private network (MVPN) traffic • **inet-vpn**—IPv4 VPN traffic • **inet6**—IP version 6 (IPv6) traffic • **inet6-mvpn**—IPv6 MVPN traffic • **inet6-vpn**—IPv6 VPN traffic • **iso**—IS-IS traffic • **route-target**—BGP route target filtering routes for VPN traffic Default setting is **inet**.	
instance ***instance-name***	Standard	Name of one or more routing instances. Match a route learned from one of the specified instances.	Name of one or more routing instances. Match a route to be advertised over one of the specified instances.
interface ***interface-name***	Standard	Name or IP address of one or more routing device interfaces. Do not use this qualifier with protocols that are not interface-specific, such as IBGP. Match a route learned from one of the specified interfaces. Direct routes match routes configured on the specified interface.	Name or IP address of one or more routing device interfaces. Do not use this qualifier with protocols that are not interface-specific, such as IBGP. Match a route to be advertised from one of the specified interfaces.
level ***level***	Standard	(Intermediate System-to-Intermediate System [IS-IS] only) IS-IS level. Match a route learned from a specified level.	(IS-IS only) IS-IS level. Match a route to be advertised to a specified level.
local-preference ***value***	Standard	(BGP only) BGP local preference (LOCAL_PREF**local-preference (add | subtract)** ***number***) attribute. The preference value can be a number in the range 0 through 4,294,967,295 (2^{32} – 1).	
metric ***metric*** **metric2** ***metric*** **metric3** ***metric*** **metric4** ***metric***	Standard	Metric value. You can specify up to four metric values, starting with **metric** (for the first metric value) and continuing with **metric2**, **metric3**, and **metric4**. (BGP only) **metric** corresponds to the multiple exit discriminator (MED), and **metric2** corresponds to the interior gateway protocol (IGP) metric if the BGP next hop runs back through another route.	

Table 9: Complete List of Routing Policy Match Conditions *(continued)*

<table>
<tr><th>Match Condition</th><th>Match Condition Category</th><th>from Statement Description</th><th>to Statement Description</th></tr>
<tr><td>multicast-scoping (scoping-name | number) < (orhigher | orlower) ></td><td>Standard</td><td colspan="2">Multicast scope value of IPv4 or IPv6 multicast group address. The multicast-scoping name corresponds to an IPv4 prefix. You can match on a specific multicast-scoping prefix or on a range of prefixes. Specify orhigher to match on a scope and numerically higher scopes, or orlower to match on a scope and numerically lower scopes. For more information, see the Multicast Protocols Feature Guide.

You can apply this scoping policy to the routing table by including the scope-policy statement at the [edit routing-options] hierarchy level.

The number value can be any hexadecimal number from 0 through F. The multicast-scope value is a number from 0 through 15, or one of the following keywords with the associated meanings:
• node-local (value=1)—No corresponding prefix
• link-local (value=2)—Corresponding prefix 224.0.0.0/24
• site-local (value=5)—No corresponding prefix
• global (value=14)—Corresponding prefix 224.0.1.0 through 238.255.255.255
• organization-local (value=8)—Corresponding prefix 239.192.0.0/14</td></tr>
<tr><td>neighbor address</td><td>Standard</td><td>Address of one or more neighbors (peers).

For BGP, the address can be a directly connected or indirectly connected peer.

For all other protocols, the address is the neighbor from which the advertisement is received.

NOTE: The neighbor address match condition is not valid for the Routing Information Protocol (RIP).</td><td>Address of one or more neighbors (peers).

For BGP import policies, specifying to neighbor produces the same result as specifying from neighbor.

For BGP export policies, specifying the neighbor match condition has no effect and is ignored.

For all other protocols, the to statement matches the neighbor to which the advertisement is sent.

NOTE: The neighbor address match condition is not valid for the Routing Information Protocol (RIP).</td></tr>
<tr><td>next-hop [addresses]</td><td>Standard</td><td colspan="2">One or more next-hop addresses specified in the routing information for a particular route. A next-hop address cannot include a netmask. For BGP routes, matches are performed against each protocol next hop.</td></tr>
<tr><td>next-hop-type merged</td><td>Standard</td><td>LDP generates a next hop based on RSVP and IP next hops available to use, combined with forwarding-class mapping.</td><td>You cannot specify this match condition.</td></tr>
<tr><td>nlri-route-type</td><td>Standard</td><td colspan="2">Route type from NLRI 1 through NLRI 10.

Multiple route types can be specified in a single policy.</td></tr>
</table>

Table 9: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
origin *value*	Standard	(BGP only) BGP origin attribute, which is the origin of the AS path information. The value can be one of the following: • **egp**—Path information originated in another AS. • **igp**—Path information originated within the local AS. • **incomplete**—Path information was learned by some other means.	
policy [*policy-name* **]**	Extended	Name of a policy to evaluate as a subroutine. For information about this extended match condition, see "Understanding Policy Subroutines in Routing Policy Match Conditions" on page 198.	
preference *preference* **preference2** *preference*	Standard	Preference value. You can specify a primary preference value (**preference**) and a secondary preference value (**preference2**). The preference value can be a number from 0 through 4,294,967,295 (2^{32} – 1). A lower number indicates a more preferred route. To specify even finer-grained preference values, see the **color** and **color2** match conditions in this table.	
prefix-list *prefix-list-name ip-addresses*	Extended	Named list of IP addresses. You can specify an exact match with incoming routes. For information about this extended match condition, see "Understanding Prefix Lists for Use in Routing Policy Match Conditions" on page 279.	You cannot specify this match condition.
prefix-list-filter *prefix-list-name match-type*	Extended	Named prefix list. You can specify prefix length qualifiers for the list of prefixes in the prefix list. For information about this extended match condition, see "Understanding Prefix Lists for Use in Routing Policy Match Conditions" on page 279.	You cannot specify this match condition.
protocol *protocol*	Standard	Name of the protocol from which the route was learned or to which the route is being advertised. It can be one of the following: **access**, **access-internal**, **aggregate**, **arp**, **bgp**, **direct**, **dvmrp**, **esis**, **frr**, **isis**, **l2circuit**, **l2vpn**, **ldp**, **local**, **msdp**, **ospf**, **ospf2**, **ospf3**, **pim**,**rip**, **ripng**, **route-target**, **rsvp**, or **static**. NOTE: The **ospf2** statement matches on OSPFv2 routes. The **ospf3** statement matches on OSPFv3 routes. The **ospf** statement matches on both OSPFv2 and OSPFv3 routes.	

Table 9: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
rib ***routing-table***	Standard	Name of a routing table. The value of ***routing-table*** can be one of the following: • inet.0—Unicast IPv4 routes • *instance-name* inet.0—Unicast IPv4 routes for a particular routing instance • inet.1—Multicast IPv4 routes • inet.2—Unicast IPv4 routes for multicast reverse-path forwarding (RPF) lookup • inet.3—MPLS routes • mpls.0—MPLS routes for label-switched path (LSP) next hops • inet6.0—Unicast IPv6 routes	
route-filter ***destination-prefix match-type <actions>***	Extended	List of destination prefixes. When specifying a destination prefix, you can specify an exact match with a specific route or a less precise match using match types. You can configure either a common action that applies to the entire list or an action associated with each prefix. For more information, see "Understanding Route Filters for Use in Routing Policy Match Conditions" on page 213.	You cannot specify this match condition.
route-type ***value***	Standard	Type of BGP route. The value can be one of the following: • **external**—External route. • **internal**—Internal route. To match IGP external routes, use the **external** match condition.	
rtf-prefix-list ***name route-targets***	Extended	(BGP only) Named list of route target prefixes for BGP route target filtering and proxy BGP route target filtering. For information about this extended match condition, see *Example: Configuring an Export Policy for BGP Route Target Filtering for VPNs*.	You cannot specify this match condition.
source-address-filter ***destination-prefix match-type <actions>***	Extended	List of multicast source addresses. When specifying a source address, you can specify an exact match with a specific route or a less precise match using match types. You can configure either a common action that applies to the entire list or an action associated with each prefix. For more information, see "Understanding Route Filters for Use in Routing Policy Match Conditions" on page 213.	You cannot specify this match condition.

Table 9: Complete List of Routing Policy Match Conditions *(continued)*

Match Condition	Match Condition Category	from Statement Description	to Statement Description
state (active | inactive)	Standard	(BGP export only) Match on the following types of advertised routes: • **active**—An active BGP route • **inactive**—A route advertised to internal BGP peers as the best external path even if the best path is an internal route • **inactive**—A route advertised by BGP as the best route even if the routing table did not select it to be an active route	
tag ***string*** **tag2** ***string***	Standard	Tag value. You can specify two tag strings: **tag** (for the first string) and **tag2**. These values are local to the router and can be set on configured routes or by using an import routing policy. You can specify multiple tags under one match condition by including the tags within a bracketed list. For example: **from tag [tag1 tag2 tag3];** For OSPF routes, the**tag** action sets the 32-bit tag field in OSPF external link-state advertisement (LSA) packets. For IS-IS routes, the **tag** action sets the 32-bit flag in the IS-IS IP prefix type length values. (TLV). OSPF stores the INTERNAL route's OSPF area ID in the**tag2** attribute. However, for EXTERNAL routes, OSPF does not store anything in the **tag2**attribute. You can configure a policy term to set the **tag2** value for a route. If the route, already has a **tag2** value (for example, an OSPF route that stores area id in tag2), then the original **tag2** value is overwritten by the new value. When the policy contains the "from area" match condition, for internal OSPF routes, where **tag2** is set, based on the OSPF area- ID, the evaluation is conducted to compare the **tag2** attribute with the area ID. For external OSPF routes that do not have the **tag2** attribute set, the match condition fails.	
validation-database	Standard	When BGP origin validation is configured, triggers a lookup in the route validation database to determine if the route prefix is valid, invalid, or unknown. The route validation database contains route origin authorization (ROA) records that map route prefixes to expected originating autonomous systems (ASs). This prevents the accidental advertisement of invalid routes. See *Example: Configuring Origin Validation for BGP*.	

Related Documentation

- Understanding Prefix Lists for Use in Routing Policy Match Conditions on page 279
- Understanding Route Filters for Use in Routing Policy Match Conditions on page 213

Route Filter Match Conditions

When specifying a destination prefix, you can specify an exact match with a specific route, or a less precise match by using match types. You can configure either a common reject action that applies to the entire list, or an action associated with each prefix.

You can specify known invalid ("bad") routes to ignore by specifying matches on destination prefixes. Additionally, you can specify that "good" routes be processed in a particular way. For instance, you can group traffic from specific source or destination addresses into forwarding classes to be processed using the class of service (CoS) feature.

Table 10 on page 54 lists route list match types.

Table 10: Route List Match Types

Match Type	Match Conditions
exact	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is equal to the route's prefix length.
longer	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is greater than the route's prefix length.
orlonger	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is equal to or greater than the route's prefix length.
prefix-length-range ***prefix-length2-prefix-length3***	The route shares the same most-significant bits (described by ***prefix-length***), and the route's prefix length falls between ***prefix-length2*** and ***prefix-length3***, inclusive.
through ***destination-prefix***	All the following are true: • The route shares the same most-significant bits (described by ***prefix-length***) of the first destination prefix. • The route shares the same most-significant bits (described by ***prefix-length***) of the second destination prefix for the number of bits in the prefix length. • The number of bits in the route's prefix length is less than or equal to the number of bits in the second prefix. You do not use the **through** match type in most routing policy configurations.
upto ***prefix-length2***	The route shares the same most-significant bits (described by ***prefix-length***) and the route's prefix length falls between ***prefix-length*** and ***prefix-length2***.

Related Documentation

- Categories of Routing Policy Match Conditions on page 42
- Summary of Routing Policy Actions on page 66
- Example: Rejecting Known Invalid Routes on page 96
- Example: Grouping Source and Destination Prefixes into a Forwarding Class on page 479

Actions in Routing Policy Terms

Each term in a routing policy can include a **then** statement, which defines the actions to take if a route matches all the conditions in the **from** and **to** statements in the term:

```
then {
  actions;
}
```

You can include this statement at the following hierarchy levels:

- **[edit policy-options** policy-statement ***policy-name*** **term *term-name*]**
- **[edit logical-systems *logical-system-name* policy-options** policy-statement ***policy-name*** **term *term-name*]**

If a term does not have **from** and **to** statements, all routes are considered to match, and the actions apply to them all. For information about the **from** and **to** statements, see "Routing Policy Match Conditions" on page 44.

You can specify one or more actions in the **then** statement. There are three types of actions:

- Flow control actions, which affect whether to accept or reject the route and whether to evaluate the next term or routing policy.
- Actions that manipulate route characteristics.
- Trace action, which logs route matches.

NOTE: When you specify an action that manipulates the route characteristics, the changes occur in a copy of the source route. The source route itself does not change. The effect of the action is visible only after the route is imported into or exported from the routing table. To view the source route before the routing policy has been applied, use the **show route receive-protocol** command. To view a route after an export policy has been applied, use the **show route advertised-protocol** command.

During policy evaluation, the characteristics in the copy of the source route always change immediately after the action is evaluated. However, the route is not copied to the routing table or a routing protocol until the policy evaluation is complete.

The **then** statement is optional. If you omit it, one of the following occurs:

- The next term in the routing policy, if one is present, is evaluated.
- If there are no more terms in the routing policy, the next routing policy, if one is present, is evaluated.

- If there are no more terms or routing policies, the accept or reject action specified by the default policy is taken. For more information, see “Default Routing Policies” on page 30.

The following sections discuss these actions:

Configuring Flow Control Actions

Table 11 on page 56 lists the flow control actions. You can specify one of these actions along with the trace action or one or more of the actions that manipulate route characteristics (see “Configuring Actions That Manipulate Route Characteristics” on page 57).

Table 11: Flow Control Actions

Flow Control Action	Description
accept	Accept the route and propagate it. After a route is accepted, no other terms in the routing policy and no other routing policies are evaluated.
default-action accept	Accept and override any action intrinsic to the protocol. This is a nonterminating policy action.
reject	Reject the route and do not propagate it. After a route is rejected, no other terms in the routing policy and no other routing policies are evaluated.
default-action reject	Reject and override any action intrinsic to the protocol. This is a nonterminating policy action.
next term	Skip to and evaluate the next term in the same routing policy. Any accept or reject action specified in the **then** statement is skipped. Any actions in the **then** statement that manipulate route characteristics are applied to the route. **next term** is the default control action if a match occurs and you do not specify a flow control action.
next policy	Skip to and evaluate the next routing policy. Any accept or reject action specified in the **then** statement is skipped. Any actions in the **then** statement that manipulate route characteristics are applied to the route. **next policy** is the default control action if a match occurs, you do not specify a flow control action, and there are no further terms in the current routing policy.

Configuring Actions That Manipulate Route Characteristics

You can specify one or more of the actions listed in Table 12 on page 57 to manipulate route characteristics.

Table 12: Actions That Manipulate Route Characteristics

Action	Description
add-path send-count *path-count*	(BGP only) Enable sending up to 20 BGP paths to a destination for a subset of **add-path** advertised prefixes.
as-path-prepend *as-path*	(BGP only) Affix one or more AS numbers at the beginning of the AS path. If specifying more than one AS number, enclose the numbers in quotation marks (" "). The AS numbers are added after the local AS number has been added to the path. This action adds AS numbers to AS sequences only, not to AS sets. If the existing AS path begins with a confederation sequence or set, the affixed AS numbers are placed within a confederation sequence. Otherwise, the affixed AS numbers are placed within a nonconfederation sequence. For more information, see "Understanding Prepending AS Numbers to BGP AS Paths" on page 326. In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS.
as-path-expand last-as count *n*	(BGP only) Extract the last AS number in the existing AS path and affix that AS number to the beginning of the AS path ***n*** times, where ***n*** is a number from 1 through 32. The AS number is added before the local AS number has been added to the path. This action adds AS numbers to AS sequences only, not to AS sets. If the existing AS path begins with a confederation sequence or set, the affixed AS numbers are placed within a confederation sequence. Otherwise, the affixed AS numbers are placed within a nonconfederation sequence. This option is typically used in non-IBGP export policies.
bgp-output-queue-priority	(BGP only) Set the output priority queue used for this route. There are 17 prioritized output queues: an expedited queue that is the highest priority, and 16 numbered queues where 1 is the lowest priority and 16 is the highest.
class *class-name*	(Class of service [CoS] only) Apply the specified class-of-service parameters to routes installed into the routing table. For more information, see the *Class of Service Feature Guide for Routing Devices*.
color *preference* color2 *preference*	Set the preference value to the specified value. The **color** and **color2** preference values are even more fine-grained than those specified in the **preference** and **preference2** actions. The color value can be a number in the range from 0 through 4,294,967,295 (2^{32} – 1). A lower number indicates a more preferred route. If you set the preference with the **color** action, the value is internal to Junos OS and is not transitive.
color (add | subtract) *number* **color2 (add | subtract) *number***	Change the color preference value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 (2^{32} – 1), the value is set to 2^{32} – 1. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value.

Table 12: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
community (+ | add) [*names*]	(BGP only) Add the specified communities to the set of communities in the route. For more information, see "Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions" on page 359.
community (– | delete) [*names*]	(BGP only) Delete the specified communities from the set of communities in the route. For more information, see "Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions" on page 359.
community (= | set) [*names*]	(BGP only) Replace any communities that were in the route in with the specified communities. For more information, see "Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions" on page 359.
cos-next-hop-map *map-name*	Set CoS-based next-hop map in forwarding table.
damping *name*	(BGP only) Apply the specified route-damping parameters to the route. These parameters override the default damping parameters. This action is useful only in an import policy, because the damping parameters affect the state of routes in the routing table. To apply damping parameters, you must enable BGP flap damping as described in the *Junos OS Routing Protocols Library*, and you must create a named list of parameters as described in "Using Routing Policies to Damp BGP Route Flapping" on page 424.
destination-class *destination-class-name*	Maintain packet counts for a route passing through your network, based on the destination address in the packet. You can do the following: • Configure group destination prefixes by configuring a routing policy. • Apply that routing policy to the forwarding table with the corresponding destination class. • Enable packet counting on one or more interfaces by including the **destination-class-usage** statement at the **[edit interfaces *interface-name* unit *logical-unit-number* family inet accounting]** hierarchy level (see the *Class of Service Feature Guide for Routing Devices*). • View the output by using one of the following commands: **show interfaces destination-class (all | *destination-class-name logical-interface-name*)**, **show interfaces *interface-name* extensive**, or **show interfaces *interface-name* statistics** (see the CLI Explorer). • To configure a packet count based on the source address, use the **source-class** statement described in this table.
external type *metric*	Set the external metric type for routes exported by OSPF. You must specify the keyword **type**.
forwarding-class *forwarding-class-name*	Create the forwarding class that includes packets based on both the destination address and the source address in the packet. You can do the following: • Configure group prefixes by configuring a routing policy. • Apply that routing policy to the forwarding table with the corresponding forwarding class. • Enable packet counting on one or more interfaces by using the procedure described in either the **destination-class** or **source-class** actions defined in this table.

Table 12: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
install-nexthop <strict> lsp *lsp-name*	Choose which next hops, among a set of equal LSP next hops, are installed in the forwarding table. Use the export policy for the forwarding table to specify the LSP next hop to be used for the desired routes. Specify the **strict** option to enable strict mode, which checks to see if any of the LSP next hops specified in the policy are up. If none of the specified LSP next hops are up, the policy installs the discard next hop.
install-to-fib	For PTX Series routers only, override the default BGP routing policy. For more information, see *Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers*.
load-balance consistent-hash	(BGP only) For MX Series routers with modular port concentrators (MPCs) and for QFX10000 switches only, specify consistent load balancing for one or more IP addresses. This feature preserves the affinity of a flow to a path in an equal-cost multipath (ECMP) group when one or more next-hop paths fail. Only flows for paths that are inactive are redirected. Flows mapped to servers that remain active are maintained.
load-balance destination-ip-only	Calculate load balancing hash based solely on destination IP address. This allows a service provider to direct traffic toward a specific content server in per-subscriber aware environments.
load-balance per-packet	(For export to the forwarding table only) Install all next-hop addresses in the forwarding table and have the forwarding table perform per-packet load balancing. This policy action allows you to optimize VPLS traffic flows across multiple paths. For more information, see *Configuring Per-Packet Load Balancing*.
load-balance per-prefix	For PTX Series routers only, override the default per-packet load balancing routing policy for BGP. For more information, see *Example: Overriding the Default BGP Routing Policy on PTX Series Packet Transport Routers*.
load-balance source-ip-only	Calculate load balancing hash based solely on source IP address. This allows a service provider to direct traffic toward a specific content server in per-subscriber aware environments.
local-preference *value*	(BGP only) Set the BGP local preference (LOCAL_PREF) attribute. The preference value can be a number in the range from 0 through 4,294,967,295 (2^{32} – 1).
local-preference (add | subtract) *number*	Change the local preference value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 (2^{32} – 1), the value is set to 2^{32} – 1. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value. For BGP, if the attribute value is not known, it is initialized to 100 before the routing policy is applied.

Table 12: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
map-to-interface (*interface-name* | self)	Sets the **map-to-interface** value which is similar to existing metric or tag actions. The **map-to-interface** action requires you to specify one of the following: • A logical interface (for example, ge-0/0/0.0). The logical interface can be any interface that multicast currently supports, including VLAN and aggregated Ethernet interfaces. NOTE: If you specify a physical interface as the **map-to-interface** (for example, ge-0/0/0), a value of .0 is appended to physical interface to create a logical interface. • The keyword **self**. The **self** keyword specifies that multicast data packets are sent on the same interface as the control packets and no mapping occurs. If no term matches, then no multicast data packets are sent.
metric *metric* metric2 *metric* metric3 *metric* metric4 *metric*	Set the metric. You can specify up to four metric values, starting with **metric** (for the first metric value) and continuing with **metric2**, **metric3**, and **metric4**. (BGP only) **metric** corresponds to the MED, and **metric2** corresponds to the IGP metric if the BGP next hop loops through another router.
metric (add | subtract) *number* metric2 (add | subtract) *number* metric3 (add | subtract) *number* metric4 (add | subtract) *number*	Change the metric value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 ($2^{32} – 1$), the value is set to $2^{32} – 1$. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value.
metric expression (metric multiplier *x* offset *a* | metric2 multiplier *y* offset *b*)	Calculate a metric based on the current values of **metric** and **metric2**. This policy action overrides the current value of the metric attribute with the result of the expression ((x * metric) + a) + ((y * metric2) + b) where **metric** and **metric2** are the current input values. Metric multipliers are limited in range to eight significant digits.
metric (igp | minimum-igp) *site-offset*	(BGP only) Change the metric (MED) value by the specified negative or positive offset. This action is useful only in an external BGP (EBGP) export policy.

Table 12: Actions That Manipulate Route Characteristics *(continued)*

Action	Description

next-hop (*address* | discard | next-table *table-name* | peer-address | reject | self)	Set the next-hop address. When the advertising protocol is BGP, you can set the next hop only when any third-party next hop can be advertised; that is, when you are using IBGP or EBGP confederations.
	
	If you specify **self**, the next-hop address is replaced by one of the local routing device's addresses. The advertising protocol determines which address to use. When the advertising protocol is BGP, this address is set to the local IP address used for the BGP adjacency. A routing device cannot install routes with itself as the next hop.
	
	If you specify **peer-address**, the next-hop address is replaced by the peer's IP address. This option is valid only in import policies. Primarily used by BGP to enforce using the peer's IP address for advertised routes, this option is meaningful only when the next hop is the advertising routing device or another directly connected routing device.
	
	If you specify **discard**, the next-hop address is replaced by a discard next hop.
	
	If you specify **next-table**, the routing device performs a forwarding lookup in the specified table.
	
	If you use the **next-table** action, the configuration must include a term qualifier that specifies a different table than the one specified in the **next-table** action. In other words, the term qualifier in the **from** statement must exclude the table in the **next-table** action. In the following example, the first term contains **rib vrf-customer2.inet.0** as a matching condition. The action specifies a next-hop in a different routing table, **vrf-customer1.inet.0**. The second term does the opposite by using **rib vrf-customer1.inet.0** in the match condition and **vrf-customer2.inet.0** In the **next-table** action.
	
	term 1 {
	  from {
	    protocol bgp;
	    rib vrf-customer2.inet.0;
	    community customer;
	  }
	  then {
	    next-hop next-table vrf-customer1.inet.0;
	  }
	}
	term 2 {
	  from {
	    protocol bgp;
	    rib vrf-customer1.inet.0;
	    community customer;
	  }
	  then {
	    next-hop next-table vrf-customer2.inet.0;
	  }
	}
	
	If you specify **reject**, the next-hop address is replaced by a reject next hop.

origin *value*	(BGP only) Set the BGP origin attribute to one of the following values: • **igp**—Path information originated within the local AS. • **egp**—Path information originated in another AS. • **incomplete**—Path information learned by some other means.

Table 12: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
p2mp-lsp-root	Set the ingress root node for a multipoint LDP (M-LDP)-based point-to-multipoint label-switched path (LSP). For more information, see *Example: Configuring Multipoint LDP In-Band Signaling for Point-to-Multipoint LSPs*.
preference ***preference*** **preference2** ***preference***	Set the preference value. You can specify a primary preference value (**preference**) and a secondary preference value (**preference2**). The preference value can be a number in the range from 0 through 4,294,967,295 ($2^{32} – 1$). A lower number indicates a more preferred route. When you use an import policy to set the value of **preference2** to the highest allowed value of 4,294,967,295, Junos OS resets this value to -1. If you set **preference2** to a number greater than ($2^{31} – 1$), it is reset to a negative value. To specify even finer-grained preference values, see the **color** and **color2** actions in this table. If you set the preference with the **preference** action, the new preference remains associated with the route. The new preference is internal to the Junos OS and is not transitive.
preference (add | subtract) ***number*** **preference2 (add | subtract)** ***number***	Change the preference value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 ($2^{32} – 1$), the value is set to $2^{32} – 1$. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value.
priority (low | medium | high)	(OSPF import only) Specify a priority for prefixes included in an OSPF import policy. Prefixes learned through OSPF are installed in the routing table based on the priority assigned to the prefixes. Prefixes assigned a priority of **high** are installed first, while prefixes assigned a priority of **low** are installed last. NOTE: An OSPF import policy can only be used to set priority or to filter OSPF external routes. If an OSPF import policy is applied that results in a **reject** terminating action for a nonexternal route, then the **reject** action is ignored and the route is accepted anyway.

Table 12: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
source-class ***source-class-name***	Maintain packet counts for a route passing through your network, based on the source address. You can do the following: • Configure group source prefixes by configuring a routing policy. • Apply that routing policy to the forwarding table with the corresponding source class. • Enable packet counting on one or more interfaces by including the **source-class-usage** ***interface-name*** statement at the **[edit interfaces** ***logical-unit-number*** **unit family inet accounting]** hierarchy level. Also, follow the **source-class-usage** statement with the **input** or **output** statement to define the inbound and outbound interfaces on which traffic monitored for source-class usage (SCU) is arriving and departing (or define one interface for both). The complete syntax is **[edit interfaces** ***interface-name*** **unit family inet accounting source-class-usage (input | output | input output)** ***unit-number*****]**. • View the output by using one of the following commands: **show interfaces** ***interface-name*** **source-class** ***source-class-name***, **show interfaces** ***interface-name*** **extensive**, or **show interfaces** ***interface-name*** **statistics** (see the CLI Explorer). • To configure a packet count based on the destination address, use the **destination-class** statement described in this table. • For a detailed source-class usage example configuration, see the "Example: Grouping Source and Destination Prefixes into a Forwarding Class" on page 479. NOTE: When configuring policy action statements, you can configure only one source class for each matching route. In other words, more than one source class cannot be applied to the same route.
ssm-source [***addresses*** **];**	Specify one or more IPv4 or IPv6 source addresses for the source-specific multicast (SSM) policy
ssm-source [***addresses*** **];**	Specify one or more IPv4 or IPv6 source addresses for the source-specific multicast (SSM) policy.
tag ***tag*** **tag2** ***tag***	Set the tag value. You can specify two tag strings: **tag** (for the first string) and **tag2** (a second string). These values are local to the router. • For OSPF routes the **tag** action sets the 32-bit tag field in OSPF external link-state advertisement (LSA) packets. • For IS-IS routes, the **tag** action sets the 32-bit flag in the IS-IS IP prefix type length values (TLV). • For RIPv2 routes, the **tag** action sets the route-tag community. The **tag2** option is not supported.
tag (add | subtract) ***number*** **tag2 (add | subtract)** ***number***	Change the tag value by the specified amount. If an addition operation results in a value that is greater than 4,294,967,295 ($2^{32} – 1$), the value is set to $2^{32} – 1$. If a subtraction operation results in a value less than 0, the value is set to 0. If an attribute value is not already set at the time of the addition or subtraction operation, the attribute value defaults to a value of 0 regardless of the amount specified. If you perform an addition to an attribute with a value of 0, the number you add becomes the resulting attribute value.

Table 12: Actions That Manipulate Route Characteristics *(continued)*

Action	Description
validation-state	When BGP origin validation is configured, set the validation state of a route prefix to valid, invalid, or unknown. The route validation database contains route origin authorization (ROA) records that map route prefixes to expected originating autonomous systems (ASs). This prevents the accidental advertisement of invalid routes. See *Understanding Origin Validation for BGP*.

Configuring the Default Action in Routing Policies

The **default-action** statement overrides any action intrinsic to the protocol. This action is also nonterminating, so that various policy terms can be evaluated before the policy is terminated. You can specify a default action, either **accept** or **reject**, as follows:

```
[edit]
policy-options {
  policy-statement policy-name {
    term term-name {
      from {
        family family-name;
        match-conditions;
        policy subroutine-policy-name;
        prefix-list name;
        route-filter destination-prefix match-type <actions>;
        source-address-filter source-prefix match-type <actions>;
      }
      to {
        match-conditions;
        policy subroutine-policy-name;
      }
      then {
        actions;
        default-action (accept | reject);
      }
    }
  }
}
```

The resulting action is set either by the protocol or by the last policy term that is matched.

Example: Configuring the Default Action in a Routing Policy

Configure a routing policy that matches routes based on three policy terms. If the route matches the first term, a certain community tag is attached. If the route matches two separate terms, then both community tags are attached. If the route does not match any terms, it is rejected (protocol's default action). Note that the terms **hub** and **spoke** are mutually exclusive.

```
[edit]
policy-options {
  policy-statement test {
```



```
        term set-default {
            then default-action reject;
        }
        term hub {
            from interface ge-2/1/0.5;
            then {
                community add test-01-hub;
                default-action accept;
            }
        }
        term spoke {
            from interface [ ge-2/1/0.1 ge-2/1/0.2 ];
            then {
                community add test-01-spoke;
                default-action accept;
            }
        }
        term management {
            from protocol direct;
            then {
                community add management;
                default-action accept;
            }
        }
    }
}
```

Configuring a Final Action in Routing Policies

In addition to specifying an action using the **then** statement in a named term, you can also specify an action using the **then** statement in an unnamed term, as follows:

```
[edit]
policy-options {
    policy-statement policy-name {
        term term-name {
            from {
                family family-name;
                match-conditions;
                policy subroutine-policy-name;
                prefix-list name;
                route-filter destination-prefix match-type <actions>;
                source-address-filter source-prefix match-type <actions>;
            }
            to {
                match-conditions;
                policy subroutine-policy-name;
            }
            then {
                actions;
            }
        }
        then action;
    }
}
```

Logging Matches to a Routing Policy Term

If you specify the trace action, the match is logged to a trace file. To set up a trace file, you must specify the following elements in the global **traceoptions** statement:

- Trace filename
- **policy** option in the **flag** statement

The following example uses the trace filename of **policy-log**:

```
[edit]
routing-options {
   traceoptions {
      file "policy-log";
      flag policy;
   }
}
```

This action does not affect the flow control during routing policy evaluation.

If a term that specifies a trace action also specifies a flow control action, the name of the term is logged in the trace file. If a term specifies a trace action only, the word <**default**> is logged.

Configuring Separate Actions for Routes in Route Lists

If you specify route lists in the **from** statement, for each route in the list, you can specify an action to take on that individual route directly, without including a **then** statement. For more information, see "Understanding Route Filters for Use in Routing Policy Match Conditions" on page 213.

Related Documentation

- Route Filter Match Conditions on page 53
- Routing Policy Match Conditions on page 44

Summary of Routing Policy Actions

An *action* is what the policy framework software does if a route matches all criteria defined in a match condition. You can configure one or more actions in a term.

The policy framework software supports the following types of actions:

- Flow control actions, which affect whether to accept or reject the route or whether to evaluate the next term or routing policy
- Actions that manipulate route characteristics
- Trace action, which logs route matches

Manipulating the route characteristics allows you to control which route is selected as the active route to reach a destination. In general, the active route is also advertised to a routing platform's neighbors. You can manipulate the following route characteristics:

AS path, class, color, community, damping parameters, destination class, external type, next hop, load balance, local preference, metric, origin, preference, and tag.

For the numeric information (color, local preference, metric, preference, and tag), you can set a specific value or change the value by adding or subtracting a specified amount. The addition and subtraction operations do not allow the value to exceed a maximum value and drop below a minimum value.

All policies have default actions in case one of the following situations arises during policy evaluation:

- A policy does not specify a match condition.
- A match occurs, but a policy does not specify an action.
- A match does not occur with a term in a policy and subsequent terms in the same policy exist.
- A match does not occur by the end of a policy.

An action defines what the router does with the route when the route matches all the match conditions in the **from** and **to** statements for a particular term. If a term does not have **from** and **to** statements, all routes are considered to match and the actions apply to all routes.

Each term can have one or more of the following types of actions. The actions are configured under the **then** statement.

- Flow control actions, which affect whether to accept or reject the route and whether to evaluate the next term or routing policy
- Actions that manipulate route characteristics
- Trace action, which logs route matches

If you do not specify an action, one of the following results occurs:

- The next term in the routing policy, if one exists, is evaluated.
- If the routing policy has no more terms, the next routing policy, if one exists, is evaluated.
- If there are no more terms or routing policies, the accept or reject action specified by the default policy is executed.

Table 13 on page 67 summarizes the routing policy actions.

Table 13: Summary of Key Routing Policy Actions

Action	Description
Flow Control Actions	These actions control the flow of routing information into and out of the routing table.
accept	Accepts the route and propagates it. After a route is accepted, no other terms in the routing policy and no other routing policies are evaluated.

Table 13: Summary of Key Routing Policy Actions *(continued)*

Action	Description
reject	Rejects the route and does not propagate it. After a route is rejected, no other terms in the routing policy and no other routing policies are evaluated.
next term	Skips to and evaluates the next term in the same routing policy. Any **accept** or **reject** action specified in the **then** statement is ignored. Any actions specified in the **then** statement that manipulate route characteristics are applied to the route.
next policy	Skips to and evaluates the next routing policy. Any **accept** or **reject** action specified in the **then** statement is ignored. Any actions specified in the **then** statement that manipulate route characteristics are applied to the route.
Route Manipulation Actions	These actions manipulate the route characteristics.
as-path-prepend ***as-path***	Appends one or more AS numbers at the beginning of the AS path. If you are specifying more than one AS number, include the numbers in quotation marks. The AS numbers are added after the local AS number has been added to the path. This action adds AS numbers to AS sequences only, not to AS sets. If the existing AS path begins with a confederation sequence or set, the appended AS numbers are placed within a confederation sequence. Otherwise, the appended AS numbers are placed with a nonconfederation sequence.
as-path-expand last-as count ***n***	Extracts the last AS number in the existing AS path and appends that AS number to the beginning of the AS path *n* times. Replace *n* with a number from **1** through **32**. The AS numbers are added after the local AS number has been added to the path. This action adds AS numbers to AS sequences only, not to AS sets. If the existing AS path begins with a confederation sequence or set, the appended AS numbers are placed within a confederation sequence. Otherwise, the appended AS numbers are placed with a nonconfederation sequence.
class ***class-name***	Applies the specified class-of-service (CoS) parameters to routes installed into the routing table.
color ***preference*** **color2** ***preference***	Sets the preference value to the specified value. The **color** and **color2** preference values can be a number from **0** through **4,294,967,295** ($2^{32} – 1$). A lower number indicates a more preferred route.
damping ***name***	Applies the specified route-damping parameters to the route. These parameters override BGP's default damping parameters. This action is useful only in import policies.
local-preference ***value***	Sets the BGP local preference attribute. The preference can be a number from **0** through **4,294,967,295** ($2^{32} – 1$).
metric ***metric*** **metric2** ***metric*** **metric3** ***metric*** **metric4** ***metric***	Sets the metric. You can specify up to four metric values, starting with **metric** (for the first metric value) and continuing with **metric2**, **metric3**, and **metric4**. For BGP routes, **metric** corresponds to the MED, and **metric2** corresponds to the IGP metric if the BGP next hop loops through another router.

Table 13: Summary of Key Routing Policy Actions *(continued)*

Action	Description
next-hop ***address***	Sets the next hop. If you specify ***address*** as **self**, the next-hop address is replaced by one of the local router's addresses. The advertising protocol determines which address to use.

Related Documentation

- *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*

Example: Configuring a Routing Policy to Advertise the Best External Route to Internal Peers

The BGP protocol specification, as defined in RFC 1771, specifies that a BGP peer shall advertise to its internal peers the higher preference external path, even if this path is not the overall best (in other words, even if the best path is an internal path). In practice, deployed BGP implementations do not follow this rule. The reasons for deviating from the specification are as follows:

- Minimizing the amount of advertised information. BGP scales according to the number of available paths.
- Avoiding routing and forwarding loops.

There are, however, several scenarios in which the behavior, specified in RFC 1771, of advertising the best external route might be beneficial. Limiting path information is not always desirable as path diversity might help reduce restoration times. Advertising the best external path can also address internal BGP (IBGP) route oscillation issues as described in RFC 3345, *Border Gateway Protocol (BGP) Persistent Route Oscillation Condition*.

The **advertise-external** statement modifies the behavior of a BGP speaker to advertise the best external path to IBGP peers, even when the best overall path is an internal path.

NOTE: **The advertise-external statement is supported at both the group and neighbor level. If you configure the statement at the neighbor level, you must configure it for all neighbors in a group. Otherwise, the group is automatically split into different groups.**

The **conditional** option limits the behavior of the **advertise-external** setting, such that the external route is advertised only if the route selection process reaches the point where the multiple exit discriminator (MED) metric is evaluated. Thus, an external route is not advertised if it has, for instance, an AS path that is worse (longer) than that of the active path. The **conditional** option restricts external path advertisement to when the best external path and the active path are equal until the MED step of the route selection process. Note that the criteria used for selecting the best external path is the same whether or not the **conditional** option is configured.

Junos OS also provides support for configuring a BGP export policy that matches the state of an advertised route. You can match either active or inactive routes, as follows:

```
policy-options {
  policy-statement name{
    from state (active|inactive);
  }
}
```

This qualifier only matches when used in the context of an export policy. When a route is being advertised by a protocol that can advertise inactive routes (such as BGP), **state inactive** matches routes advertised as a result of the **advertise-inactive** and **advertise-external** statements.

For example, the following configuration can be used as a BGP export policy toward internal peers to mark routes advertised due to the **advertise-external** setting with a user-defined community. That community can be later used by the receiving routers to filter out such routes from the forwarding table. Such a mechanism can be used to address concerns that advertising paths not used for forwarding by the sender might lead to forwarding loops.

```
user@host# show policy-options
policy-statement mark-inactive {
  term inactive {
    from state inactive;
    then {
      community set comm-inactive;
    }
  }
  term default {
    from protocol bgp;
    then accept;
  }
  then reject;
}
community comm-inactive members 65536:65284;
```


Requirements

Junos OS 9.3 or later is required.

Overview

This example shows three routing devices. Device R2 has an external BGP (EBGP) connection to Device R1. Device R2 has an IBGP connection to Device R3.

Device R1 advertises 172.16.6.0/24. Device R2 does not set the local preference in an import policy for Device R1's routes, and thus 172.16.6.0/24 has the default local preference of 100.

Device R3 advertises 172.16.6.0/24 with a local preference of 200.

When the **advertise-external** statement is not configured on Device R2, 172.16.6.0/24 is not advertised by Device R2 toward Device R3.

When the **advertise-external** statement is configured on Device R2 on the session toward Device R3, 172.16.6.0/24 is advertised by Device R2 toward Device R3.

When **advertise-external conditional** is configured on Device R2 on the session toward Device R3, 172.16.6.0/24 is not advertised by Device R2 toward Device R3. If you remove the **then local-preference 200** setting on Device R3 and add the **path-selection as-path-ignore** setting on Device R2 (thus making the path selection criteria equal until the MED step of the route selection process), 172.16.6.0/24 is advertised by Device R2 toward Device R3.

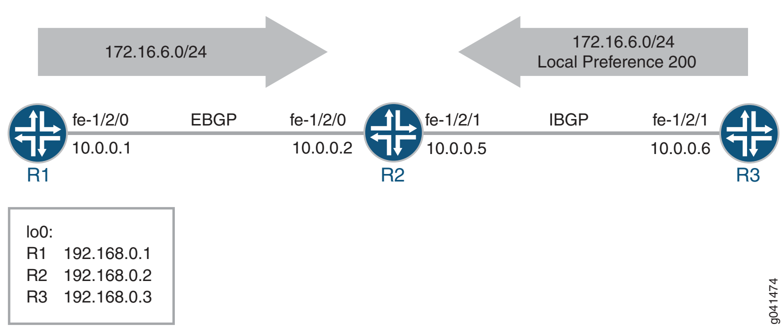
NOTE: To configure the **advertise-external** statement on a route reflector, you must disable intracluster reflection with the **no-client-reflect** statement, and the client cluster must be fully meshed to prevent the sending of redundant route advertisements.

When a routing device is configured as a route reflector for a cluster, a route advertised by the route reflector is considered internal if it is received from an internal peer with the same cluster identifier or if both peers have no cluster identifier configured. A route received from an internal peer that belongs to another cluster, that is, with a different cluster identifier, is considered external.

Topology

Figure 9 on page 71 shows the sample network.

Figure 9: BGP Topology for advertise-external



"CLI Quick Configuration" on page 72 shows the configuration for all of the devices in Figure 9 on page 71.

The section "Step-by-Step Procedure" on page 73 describes the steps on Device R2.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 description to-R2
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 from route-filter 172.16.6.0/24
  exact
set policy-options policy-statement send-static term 1 then accept
set policy-options policy-statement send-static term 2 then reject
set routing-options static route 172.16.6.0/24 reject
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 description to-R1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 description to-R3
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 100
set protocols bgp group ext neighbor 10.0.0.1
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int advertise-external
set protocols bgp group int neighbor 192.168.0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/0 unit 6 family inet address 10.0.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.3
set protocols bgp group int export send-static
set protocols bgp group int neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/0.6
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then local-preference 200
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.16.6.0/24 reject
set routing-options static route 0.0.0.0/0 next-hop 10.0.0.5
```

set routing-options autonomous-system 200

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 description to-R1
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

user@R2# set fe-1/2/1 unit 0 description to-R3
user@R2# set fe-1/2/1 unit 0 family inet address 10.0.0.5/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure OSPF or another interior gateway protocol (IGP).

```
[edit protocols ospf area 0.0.0.0]
user@R2# set interface fe-1/2/1.0
user@R2# set interface lo0.0 passive
```

3. Configure the EBGP connection to Device R1.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set peer-as 100
user@R2# set neighbor 10.0.0.1
```

4. Configure the IBGP connection to Device R3.

```
[edit protocols bgp group int]
user@R2# set type internal
user@R2# set local-address 192.168.0.2
user@R2# set neighbor 192.168.0.3
```

5. Add the **advertise-external** statement to the IBGP group peering session.

```
[edit protocols bgp group int]
user@R2# set advertise-external
```

6. Configure the autonomous system (AS) number and the router ID.

```
[edit routing-options ]
user@R2# set router-id 192.168.0.2
user@R2# set autonomous-system 200
```


Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
fe-1/2/0 {
   unit 0{
      description to-R1;
      family inet {
         address 10.0.0.2/30;
      }
   }
}
fe-1/2/1 {
   unit 0 {
      description to-R3;
      family inet {
         address 10.0.0.5/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.2/32;
      }
   }
}
```

```
user@R2# show protocols
bgp {
   group ext {
      type external;
      peer-as 100;
      neighbor 10.0.0.1;
   }
   group int {
      type internal;
      local-address 192.168.0.2;
      advertise-external;
      neighbor 192.168.0.3;
   }
}
ospf {
   area 0.0.0.0 {
      interface fe-1/2/1.0;
      interface lo0.0 {
         passive;
      }
   }
}
```

```
user@R2# show routing-options
router-id 192.168.0.2;
autonomous-system 200;
```


If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Verifying the BGP Active Path on page 75
- Verifying the External Route Advertisement on page 75
- Verifying the Route on Device R3 on page 76
- Experimenting with the conditional Option on page 76

Verifying the BGP Active Path

Purpose On Device R2, make sure that the 172.16.6.0/24 prefix is in the routing table and has the expected active path.

Action user@R2> **show route 172.16.6**

```
inet.0: 8 destinations, 9 routes (8 active, 1 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.6.0/24      *[BGP/170] 00:00:07, localpref 200, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/1.0
                    [BGP/170] 03:23:03, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
```

Meaning Device R2 receives the 172.16.6.0/24 route from both Device R1 and Device R3. The route from Device R3 is the active path, as designated by the asterisk (*). The active path has the highest local preference. Even if the local preferences of the two routes were equal, the route from Device R3 would remain active because it has the shortest AS path.

Verifying the External Route Advertisement

Purpose On Device R2, make sure that the 172.16.6.0/24 route is advertised toward Device R3.

Action user@R2> **show route advertising-protocol bgp 192.168.0.3**

```
inet.0: 8 destinations, 9 routes (8 active, 1 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.6.0/24           10.0.0.1                     100        100 I
```

Meaning Device R2 is advertising the 172.16.6.0/24 route toward Device R3.

Verifying the Route on Device R3

Purpose Make sure that the 172.16.6.0/24 prefix is in Device R3's routing table.

Action

```
user@R3> show route 172.16.6.0/24

inet.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.6.0/24      *[Static/5] 03:34:14
                      Reject
                    [BGP/170] 06:34:43, localpref 100, from 192.168.0.2
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/0.6
```

Meaning Device R3 has the static route and the BGP route for 172.16.6.0/24.

Note that the BGP route is hidden on Device R3 if the route is not reachable or if the next hop cannot be resolved. To fulfill this requirement, this example includes a static default route on Device R3 (**static route 0.0.0.0/0 next-hop 10.0.0.5**).

Experimenting with the conditional Option

Purpose See how the **conditional** option works in the context of the BGP path selection algorithm.

Action

1. On Device R2, add the **conditional** option.

   ```
   [edit protocols bgp group int]
   user@R2# set advertise-external conditional
   user@R2# commit
   ```

2. On Device R2, check to see if the 172.16.6.0/24 route is advertised toward Device R3.

   ```
   user@R2> show route advertising-protocol bgp 192.168.0.3
   ```

   As expected, the route is no longer advertised. You might need to wait a few seconds to see this result.

3. On Device R3, deactivate the **then local-preference** policy action.

   ```
   [edit policy-options policy-statement send-static term 1]
   user@R3# deactivate logical-systems R3 then local-preference
   user@R3# commit
   ```

4. On Device R2, ensure that the local preferences of the two paths are equal.

   ```
   user@R2> show route 172.16.6.0/24

   inet.0: 8 destinations, 9 routes (8 active, 0 holddown, 0 hidden)
   + = Active Route, - = Last Active, * = Both
   ```



```
172.16.6.0/24       *[BGP/170] 08:02:59, localpref 100
                       AS path: 100 I, validation-state: unverified
                     > to 10.0.0.1 via fe-1/2/0.0
                     [BGP/170] 00:07:51, localpref 100, from 192.168.0.3
                       AS path: I, validation-state: unverified
                     > to 10.0.0.6 via fe-1/2/1.0
```

5. On Device R2, add the **as-path-ignore** statement.

```
[edit protocols bgp]
user@R2# set path-selection as-path-ignore
user@R2# commit
```

6. On Device R2, check to see if the 172.16.6.0/24 route is advertised toward Device R3.

```
user@R2> show route advertising-protocol bgp 192.168.0.3

inet.0: 8 destinations, 9 routes (8 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.6.0/24           10.0.0.1                     100        100 I
```

As expected, the route is now advertised because the AS path length is ignored and because the local preferences are equal.

Related Documentation

- Example: Configuring BGP to Advertise Inactive Routes on page 77
- *Understanding BGP Path Selection*

Example: Configuring BGP to Advertise Inactive Routes

By default, BGP readvertises only active routes. To have the routing table export to BGP the best route learned by BGP even if Junos OS did not select it to be an active route, include the **advertise-inactive** statement:

```
advertise-inactive;
```

In Junos OS, BGP advertises BGP routes that are installed or active, which are routes selected as the best based on the BGP path selection rules. The **advertise-inactive** statement allows nonactive BGP routes to be advertised to other peers.

NOTE: If the routing table has two BGP routes where one is active and the other is inactive, the **advertise-inactive** statement does not advertise the inactive BGP prefix. This statement does not advertise an inactive BGP route in the presence of another active BGP route. However, if the active route is a static route, the **advertise-inactive** statement advertises the inactive BGP route.

Junos OS also provides support for configuring a BGP export policy that matches the state of an advertised route. You can match either active or inactive routes, as follows:

```
policy-options {
  policy-statement name{
```

```
        from state (active|inactive);
    }
}
```

This qualifier only matches when used in the context of an export policy. When a route is being advertised by a protocol that can advertise inactive routes (such as BGP), **state inactive** matches routes advertised as a result of the **advertise-inactive** (or **advertise-external**) statement.

For example, the following configuration can be used as a BGP export policy to mark routes advertised due to the **advertise-inactive** setting with a user-defined community. That community can be later used by the receiving routers to filter out such routes from the forwarding table. Such a mechanism can be used to address concerns that advertising paths not used for forwarding by the sender might lead to forwarding loops.

```
user@host# show policy-options
policy-statement mark-inactive {
    term inactive {
        from state inactive;
        then {
            community set comm-inactive;
        }
    }
    term default {
        from protocol bgp;
        then accept;
    }
    then reject;
}
community comm-inactive members 65536:65284;
```


Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

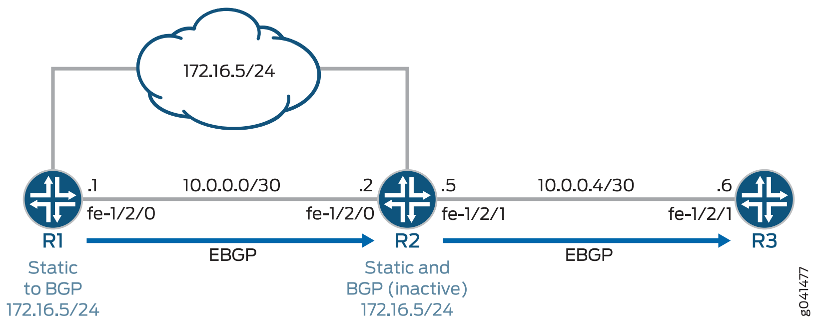
In this example, Device R2 has two external BGP (EBGP) peers, Device R1 and Device R3.

Device R1 has a static route to 172.16.5/24. Likewise, Device R2 also has a static route to 172.16.5/24. Through BGP, Device R1 sends information about its static route to Device R2. Device R2 now has information about 172.16.5/24 from two sources—its own static route and the BGP-learned route received from Device R1. Static routes are preferred over BGP-learned routes, so the BGP route is inactive on Device R2. Normally Device R2 would send the BGP-learned information to Device R3, but Device R2 does not do this because the BGP route is inactive. Device R3, therefore, has no information about 172.16.5/24 unless you enable the **advertise-inactive** command on Device R2, which causes Device R2 to send the BGP-learned to Device R3.

Topology

Figure 10 on page 79 shows the sample network.

Figure 10: BGP Topology for advertise-inactive



“CLI Quick Configuration” on page 79 shows the configuration for all of the devices in Figure 10 on page 79.

The section “Step-by-Step Procedure” on page 80 describes the steps on Device R2.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group to_R2 type external
set protocols bgp group to_R2 export send-static
set protocols bgp group to_R2 neighbor 10.0.0.2 peer-as 200
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.16.5.0/24 discard
set routing-options static route 172.16.5.0/24 install
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.5/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group to_R1 type external
set protocols bgp group to_R1 neighbor 10.0.0.1 peer-as 100
set protocols bgp group to_R3 type external
set protocols bgp group to_R3 advertise-inactive
set protocols bgp group to_R3 neighbor 10.0.0.6 peer-as 300
set routing-options static route 172.16.5.0/24 discard
set routing-options static route 172.16.5.0/24 install
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.6/30
```

```
set interfaces fe-1/2/0 unit 9 family inet address 10.0.0.9/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.5
set routing-options autonomous-system 300
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

   user@R2# set fe-1/2/1 unit 0 family inet address 10.0.0.5/30

   user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
   ```

2. Configure the EBGP connection to Device R1.

   ```
   [edit protocols bgp group to_R1]
   user@R2# set type external
   user@R2# set neighbor 10.0.0.1 peer-as 100
   ```

3. Configure the EBGP connection to Device R3.

   ```
   [edit protocols bgp group to_R3]
   user@R2# set type external
   user@R2# set neighbor 10.0.0.6 peer-as 300
   ```

4. Add the **advertise-inactive** statement to the EBGP group peering session with Device R3.

   ```
   [edit protocols bgp group to_R3]
   user@R2# set advertise-inactive
   ```

5. Configure the static route to the 172.16.5.0/24 network.

   ```
   [edit routing-options static]
   user@R2# set route 172.16.5.0/24 discard
   user@R2# set route 172.16.5.0/24 install
   ```

6. Configure the autonomous system (AS) number.

   ```
   [edit routing-options]
   user@R2# set autonomous-system 200
   ```


Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      address 10.0.0.2/30;
    }
  }
}
fe-1/2/1 {
  unit 0 {
    family inet {
      address 10.0.0.5/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.2/32;
    }
  }
}
```

```
user@R2# show protocols
bgp {
  group to_R1 {
    type external;
    neighbor 10.0.0.1 {
      peer-as 100;
    }
  }
  group to_R3 {
    type external;
    advertise-inactive;
    neighbor 10.0.0.6 {
      peer-as 300;
    }
  }
}
```

```
user@R2# show routing-options
static {
  route 172.16.5.0/24 {
    discard;
    install;
  }
}
autonomous-system 200;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the BGP Active Path

Purpose On Device R2, make sure that the 172.16.5.0/24 prefix is in the routing table and has the expected active path.

Action user@R2> **show route 172.16.5**

```
inet.0: 7 destinations, 8 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.5.0/24      *[Static/5] 21:24:38
                      Discard
                    [BGP/170] 21:21:41, localpref 100
                      AS path: 100 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
```

Meaning Device R2 receives the 172.16.5.0/24 route from both Device R1 and from its own statically configured route. The static route is the active path, as designated by the asterisk (*). The static route path has the lowest route preference (5) as compared to the BGP preference (170). Therefore, the static route becomes active.

Verifying the External Route Advertisement

Purpose On Device R2, make sure that the 172.16.5.0/24 route is advertised toward Device R3.

Action user@R2> **show route advertising-protocol bgp 10.0.0.6**

```
inet.0: 6 destinations, 7 routes (6 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.5.0/24           Self                                    100 I
```

Meaning Device R2 is advertising the 172.16.5.0/24 route toward Device R3

Verifying the Route on Device R3

Purpose Make sure that the 172.16.6.0/24 prefix is in Device R3's routing table.

Action

user@R3> **show route 172.16.5.0/24**

```
inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.5.0/24      *[BGP/170] 00:01:19, localpref 100
                      AS path: 200 100 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/1.0
```

Meaning

Device R3 has the BGP-learned route for 172.16.5.0/24.

Experimenting with the advertise-inactive Statement

Purpose

See what happens when the **advertise-inactive** statement is removed from the BGP configuration on Device R2.

Action

1. On Device R2, deactivate the **advertise-inactive** statement.

   [edit protocols bgp group to_R3]
   user@R2# **deactivate advertise-inactive**
   user@R2# **commit**

2. On Device R2, check to see if the 172.16.5.0/24 route is advertised toward Device R3.

   user@R2> **show route advertising-protocol bgp 10.0.0.6**

   As expected, the route is no longer advertised.

3. On Device R3, ensure that the 172.16.5/24 route is absent from the routing table.

   user@R3> **show route 172.16.5/24**

Meaning

Device R1 advertises route 172.16.5/24 to Device R2, but Device R2 has a manually configured static route for this prefix. Static routes are preferred over BGP routes, so Device R2 installs the BGP route as an inactive route. Because the BGP route is not active, Device R2 does not readvertise the BGP route to Device R3. This is the default behavior in Junos OS. If you add the **advertise-inactive** statement to the BGP configuration on Device R2, Device R2 readvertises nonactive routes.

Related Documentation

- Example: Configuring a Routing Policy to Advertise the Best External Route to Internal Peers on page 69
- *Understanding BGP Path Selection*

Example: Using Routing Policy to Set a Preference Value for BGP Routes

This example shows how to use routing policy to set the preference for routes learned from BGP. Routing information can be learned from multiple sources. To break ties among equally specific routes learned from multiple sources, each source has a preference value. Routes that are learned through explicit administrative action, such as static routes, are preferred over routes learned from a routing protocol, such as BGP or OSPF. This concept is called *administrative distance* by some vendors.

Requirements

No special configuration beyond device initialization is required before you configure this example.

Overview

Routing information can be learned from multiple sources, such as through static configuration, BGP, or an interior gateway protocol (IGP). When Junos OS determines a route's preference to become the active route, it selects the route with the lowest preference as the active route and installs this route into the forwarding table. By default, the routing software assigns a preference of 170 to routes that originated from BGP. Of all the routing protocols, BGP has the highest default preference value, which means that routes learned by BGP are the least likely to become the active route.

Some vendors have a preference (distance) of 20 for external BGP (EBGP) and a distance of 200 for internal BGP (IGBP). Junos OS uses the same value (170) for both EBGP and IBGP. However, this difference between vendors has no operational impact because Junos OS always prefers EBGP routes over IBGP routes.

Another area in which vendors differ is in regard to IGP distance compared to BGP distance. For example, some vendors assign a distance of 110 to OSPF routes. This is higher than the EBGP distance of 20 , and results in the selection of an EBGP route over an equivalent OSPF route. In the same scenario, Junos OS chooses the OSPF route, because of the default preference 10 for an internal OSPF route and 150 for an external OSPF route, which are both lower than the 170 preference assigned to all BGP routes.

This example shows a routing policy that matches routes from specific next hops and sets a preference. If a route does not match the first term, it is evaluated by the second term.

Topology

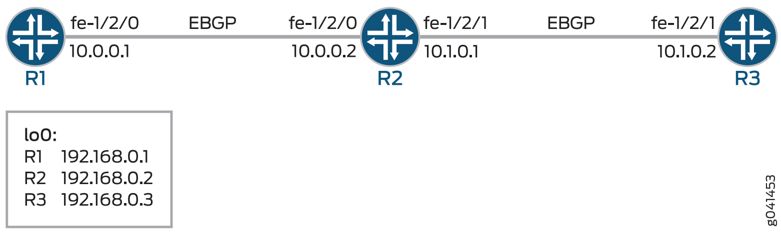
In the sample network, Device R1 and Device R3 have EBGP sessions with Device R2.

On Device R2, an import policy takes the following actions:

- For routes received through BGP from next-hop 10.0.0.1 (Device R1), set the route preference to 10.
- For routes received through BGP from next-hop 10.1.0.2 (Device R3), set the route preference to 15.

Figure 11 on page 85 shows the sample network.

Figure 11: BGP Preference Value Topology



"CLI Quick Configuration" on page 85 shows the configuration for all of the devices in Figure 11 on page 85.

The section "Step-by-Step Procedure" on page 86 describes the steps on Device R2.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext import set-preference
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set protocols bgp group ext neighbor 10.1.0.2 peer-as 300
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement set-preference term term1 from protocol bgp
set policy-options policy-statement set-preference term term1 from next-hop 10.0.0.1
```

```
set policy-options policy-statement set-preference term term1 then preference 10
set policy-options policy-statement set-preference term term2 from protocol bgp
set policy-options policy-statement set-preference term term2 from next-hop 10.1.0.2
set policy-options policy-statement set-preference term term2 then preference 15
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.1.0.1
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 300
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

user@R2# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the local autonomous system.

```
[edit routing-options]
user@R2# set autonomous-system 200
```

3. Configure the routing policy that sends direct routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

4. Configure the routing policy that changes the preference of received routes.

```
[edit policy-options policy-statement set-preference]
user@R2# set term term1 from protocol bgp
user@R2# set term term1 from next-hop 10.0.0.1
user@R2# set term term1 then preference 10

user@R2# set term term2 from protocol bgp
user@R2# set term term2 from next-hop 10.1.0.2
user@R2# set term term2 then preference 15
```

5. Configure the external peering with Device R2.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set export send-direct
user@R2# set neighbor 10.0.0.1 peer-as 100
user@R2# set neighbor 10.1.0.2 peer-as 300
```

6. Apply the **set-preference** policy as an import policy.

   This affects Device R2's routing table and has no impact on Device R1 and Device R3.

```
[edit protocols bgp group ext]
user@R2# set import set-preference
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
fe-1/2/0 {
   unit 0 {
      family inet {
         address 10.0.0.2/30;
      }
   }
}
fe-1/2/1 {
   unit 0 {
      family inet {
         address 10.1.0.1/30;
      }
   }
}
lo0 {
   unit 0{
      family inet {
         address 192.168.0.2/32;
      }
   }
}
```

```
user@R2# show protocols
bgp {
   group ext {
      type external;
      import set-preference;
      export send-direct;
      neighbor 10.0.0.1 {
         peer-as 100;
      }
      neighbor 10.1.0.2 {
```

```
            peer-as 300;
        }
    }
}
user@R2# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
policy-statement set-preference {
    term term1 {
        from {
            protocol bgp;
            next-hop 10.0.0.1;
        }
        then {
            preference 10;
        }
    }
    term term2 {
        from {
            protocol bgp;
            next-hop 10.1.0.2;
        }
        then {
            preference 15;
        }
    }
}
user@R2# show routing-options
autonomous-system 200;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the Preference

Purpose Make sure that the routing tables on Device R1 and Device R2 reflect the fact that Device R1 is using the configured EBGP preference of 8, and Device R2 is using the default EBGP preference of 170.

Action From operational mode, enter the **show route protocols bgp** command.

```
user@R2> show route protocols bgp
inet.0: 7 destinations, 9 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/30         [BGP/10] 04:42:23, localpref 100
```



```
                        AS path: 100 I, validation-state: unverified
                      > to 10.0.0.1 via fe-1/2/0.0
10.1.0.0/30           [BGP/15] 04:42:23, localpref 100
                        AS path: 300 I, validation-state: unverified
                      > to 10.1.0.2 via fe-1/2/1.0
192.168.0.1/32       *[BGP/10] 04:42:23, localpref 100
                        AS path: 100 I, validation-state: unverified
                      > to 10.0.0.1 via fe-1/2/0.0
192.168.0.3/32       *[BGP/15] 04:42:23, localpref 100
                        AS path: 300 I, validation-state: unverified
                      > to 10.1.0.2 via fe-1/2/1.0
```

Meaning The output shows that on Device R2, the preference values have been changed to 15 for routes learned from Device R3, and the preference values have been changed to 10 for routes learned from Device R1.

Related Documentation

- *Route Preferences Overview*
- *Understanding External BGP Peering Sessions*

Example: Enabling BGP Route Advertisements

Junos OS does not advertise the routes learned from one EBGP peer back to the same external BGP (EBGP) peer. In addition, the software does not advertise those routes back to any EBGP peers that are in the same autonomous system (AS) as the originating peer, regardless of the routing instance. You can modify this behavior by including the **advertise-peer-as** statement in the configuration.

If you include the **advertise-peer-as** statement in the configuration, BGP advertises the route regardless of this check.

To restore the default behavior, include the **no-advertise-peer-as** statement in the configuration:

```
no-advertise-peer-as;
```

The route suppression default behavior is disabled if the **as-override** statement is included in the configuration. If you include both the **as-override** and **no-advertise-peer-as** statements in the configuration, the **no-advertise-peer-as** statement is ignored.

Requirements

No special configuration beyond device initialization is required before you configure this example.

Overview

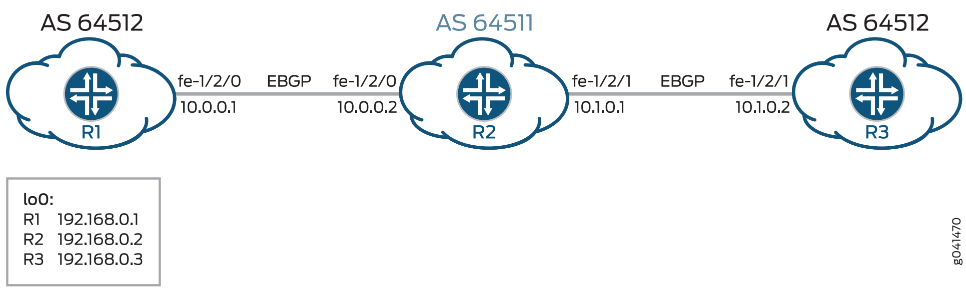
This example shows three routing devices with external BGP (EBGP) connections. Device R2 has an EBGP connection to Device R1 and another EBGP connection to Device R3. Although separated by Device R2 which is in AS 64511, Device R1 and Device R3 are in the same AS (AS 64512). Device R1 and Device R3 advertise into BGP direct routes to their own loopback interface addresses.

Device R2 receives these loopback interface routes, and the **advertise peer-as** statement allows Device R2 to advertise them. Specifically, Device R1 sends the 192.168.0.1 route to Device R2, and because Device R2 has the **advertise peer-as** configured, Device R2 can send the 192.168.0.1 route to Device R3. Likewise, Device R3 sends the 192.168.0.3 route to Device R2, and **advertise peer-as** enables Device R2 to forward the route to Device R1.

To enable Device R1 and Device R3 to accept routes that contain their own AS number in the AS path, the **loops 2** statement is required on Device R1 and Device R3.

Topology

Figure 12: BGP Topology for advertise-peer-as



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp family inet unicast loops 2
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64512
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
```

```
set protocols bgp group ext type external
set protocols bgp group ext advertise-peer-as
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 64512
set protocols bgp group ext neighbor 10.1.0.2 peer-as 64512
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64511
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp family inet unicast loops 2
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.1.0.1
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 64512
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30

user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure BGP.

```
[edit protocols bgp group ext]
user@R1# set type external
user@R1# set peer-as 64511
user@R1# set neighbor 10.0.0.2
```

3. Prevent routes from Device R3 from being hidden on Device R1 by including the **loops 2** statement.

   The **loops 2** statement means that the local device's own AS number can appear in the AS path up to one time without causing the route to be hidden. The route is hidden if the local device's AS number is detected in the path two or more times.

```
[edit protocols bgp family inet unicast]
user@R1# set loops 2
```

4. Configure the routing policy that sends direct routes.

```
[edit policy-options policy-statement send-direct term 1]
```

```
user@R1# set from protocol direct
user@R1# set then accept
```

5. Apply the export policy to the BGP peering session with Device R2.

```
[edit protocols bgp group ext]
user@R1# set export send-direct
```

6. Configure the autonomous system (AS) number.

```
[edit routing-options ]
user@R1# set autonomous-system 64512
```

Step-by-Step Procedure

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

user@R2# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure BGP.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set neighbor 10.0.0.1 peer-as 64512
user@R2# set neighbor 10.1.0.2 peer-as 64512
```

3. Configure Device R2 to advertise routes learned from one EBGP peer to another EBGP peer in the same AS.

   In other words, advertise to Device R1 routes learned from Device R3 (and the reverse), even though Device R1 and Device R3 are in the same AS.

```
[edit protocols bgp group ext]
user@R2# set advertise-peer-as
```

4. Configure a routing policy that sends direct routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

5. Apply the export policy.

```
[edit protocols bgp group ext]
user@R2# set export send-direct
```

6. Configure the AS number.

```
[edit routing-options]
user@R2# set autonomous-system 64511
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device R1

```
user@R1# show interfaces
fe-1/2/0 {
   unit 0 {
      family inet {
         address 10.0.0.1/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.1/32;
      }
   }
}

user@R1# show protocols
bgp {
   family inet {
      unicast {
         loops 2;
      }
   }
   group ext {
      type external;
      export send-direct;
      peer-as 64511;
      neighbor 10.0.0.2;
   }
}

user@R1# show policy-options
policy-statement send-direct {
   term 1 {
      from protocol direct;
      then accept;
   }
}

user@R1# show routing-options
autonomous-system 64512;
```

Device R2

```
user@R2# show interfaces
fe-1/2/0 {
   unit 0 {
      family inet {
```

```
            address 10.0.0.2/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        family inet {
            address 10.1.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}

user@R2# show protocols
bgp {
    group ext {
        type external;
        advertise-peer-as;
        export send-direct;
        neighbor 10.0.0.1 {
            peer-as 64512;
        }
        neighbor 10.1.0.2 {
            peer-as 64512;
        }
    }
}

user@R2# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}

user@R2# show routing-options
autonomous-system 64511;
```

If you are done configuring the devices, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the BGP Routes

Purpose Make sure that the routing tables on Device R1 and Device R3 contain the expected routes.

Action

1. On Device R2, deactivate the **advertise-peer-as** statement in the BGP configuration.

   [edit protocols bgp group ext]
   user@R2# **deactivate advertise-peer-as**
   user@R2# **commit**

2. On Device R3, deactivate the **loops** statement in the BGP configuration.

   [edit protocols bgp family inet unicast]
   user@R3# **deactivate unicast loops**
   user@R3# **commit**

3. On Device R1, check to see what routes are advertised to Device R2.

```
user@R1> show route advertising-protocol bgp 10.0.0.2
inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                                    I
* 192.168.0.1/32          Self                                    I
```

4. On Device R2, check to see what routes are received from Device R1.

```
user@R2> show route receive-protocol bgp 10.0.0.1
inet.0: 7 destinations, 9 routes (7 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  10.0.0.0/30             10.0.0.1                                64512 I
* 192.168.0.1/32          10.0.0.1                                64512 I
```

5. On Device R2, check to see what routes are advertised to Device R3.

```
user@R2> show route advertising-protocol bgp 10.1.0.2
inet.0: 7 destinations, 9 routes (7 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                                    I
* 10.1.0.0/30             Self                                    I
* 192.168.0.2/32          Self                                    I
```

6. On Device R2, activate the **advertise-peer-as** statement in the BGP configuration.

   [edit protocols bgp group ext]
   user@R2# **activate advertise-peer-as**
   user@R2# **commit**

7. On Device R2, recheck the routes that are advertised to Device R3.

```
user@R2> show route advertising-protocol bgp 10.1.0.2
inet.0: 7 destinations, 9 routes (7 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             Self                                    I
* 10.1.0.0/30             Self                                    I
* 192.168.0.1/32          Self             64512 I
* 192.168.0.2/32          Self                                    I
* 192.168.0.3/32          10.1.0.2         64512 I
```

8. On Device R3, check the routes that are received from Device R2.

```
user@R3> show route receive-protocol bgp 10.1.0.1
inet.0: 5 destinations, 6 routes (5 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             10.1.0.1                                64511 I
```



```
  10.1.0.0/30             10.1.0.1                                    64511 I
* 192.168.0.2/32          10.1.0.1                                    64511 I
```

9. On Device R3, activate the **loops** statement in the BGP configuration.

   [edit protocols bgp family inet unicast]
   user@R3# **activate unicast loops**
   user@R3# **commit**

10. On Device R3, recheck the routes that are received from Device R2.

```
user@R3> show route receive-protocol bgp 10.1.0.1
inet.0: 6 destinations, 8 routes (6 active, 0 holddown, 1 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.0/30             10.1.0.1                                64511 I
  10.1.0.0/30             10.1.0.1                                64511 I
* 192.168.0.1/32          10.1.0.1                                64511 64512
 I
* 192.168.0.2/32          10.1.0.1                                64511 I
```

Meaning First the **advertise-peer-as** statement and the **loops** statement are deactivated so that the default behavior can be examined. Device R1 sends to Device R2 a route to Device R1's loopback interface address, 192.168.0.1/32. Device R2 does not advertise this route to Device R3. After activating the **advertise-peer-as** statement, Device R2 does advertise the 192.168.0.1/32 route to Device R3. Device R3 does not accept this route until after the **loops** statement is activated.

Related Documentation

- *Example: Configuring a Layer 3 VPN with Route Reflection and AS Override*

Example: Rejecting Known Invalid Routes

This example shows how to create route-based match conditions for a routing policy.

Requirements

Before you begin, be sure your router interfaces and protocols are correctly configured.

Overview

In this example, you create a policy called rejectpolicy1 that rejects routes with a mask of /8 and greater (/8, /9, /10, and so on) that have the first 8 bits set to 0. This policy also accepts routes less than 8 bits in length by creating a mask of 0/0 up to /7.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set policy-options policy-statement rejectpolicy1 term rejectterm1 from route-filter
   0.0.0.0/0 upto /7 accept
set policy-options policy-statement rejectpolicy1 term rejectterm1 from route-filter
   0.0.0.0/8 orlonger reject
set policy-options policy-statement test term 1 from protocol direct
```

Step-by-Step Procedure

To create a policy that rejects known invalid routes:

1. Create the routing policy.

```
[edit]
user@host# edit policy-options policy-statement rejectpolicy1
```

2. Create the policy term.

```
[edit policy-options policy-statement rejectpolicy1]
user@host# edit term rejectterm1
```

3. Create a mask that specifies which routes to accept.

```
[edit policy-options policy-statement rejectpolicy1 term rejectterm1]
user@host# set from route-filter 0/0 upto /7 accept
```

4. Create a mask that specifies which routes to reject.

```
[edit policy-options policy-statement rejectpolicy1 term rejectterm1]
user@host# set from route-filter 0/8 orlonger reject
```

Results

Confirm your configuration by entering the **show policy-options** command from configuration mode. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
user@host# show policy-options
policy-statement rejectpolicy1 {
   term rejectterm1 {
      from {
         route-filter 0.0.0.0/0 upto /7 accept;
         route-filter 0.0.0.0/8 orlonger reject;
      }
   }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

To confirm that the configuration is working properly, perform these tasks:

Verifying the Route-Based Match Conditions

Purpose Verify that the policy and term are configured on the device with the appropriate route-based match conditions.

Action From operational mode, enter the **show policy-options** command.

Related Documentation

- *Junos OS Feature Support Reference for SRX Series and J Series Devices*
- Route Filter Match Conditions on page 53
- Example: Grouping Source and Destination Prefixes into a Forwarding Class on page 479

Example: Using Routing Policy in an ISP Network

This example is a case study in how routing policies might be used in a typical Internet service provider (ISP) network.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this network example, the ISP's AS number is 64510. The ISP has two transit peers (AS 64514 and AS 64515) to which it connects at an exchange point. The ISP is also connected to two private peers (AS 64513 and AS 64516) with which it exchanges specific customer routes. The ISP has two customers (AS 64511 and AS 64512).

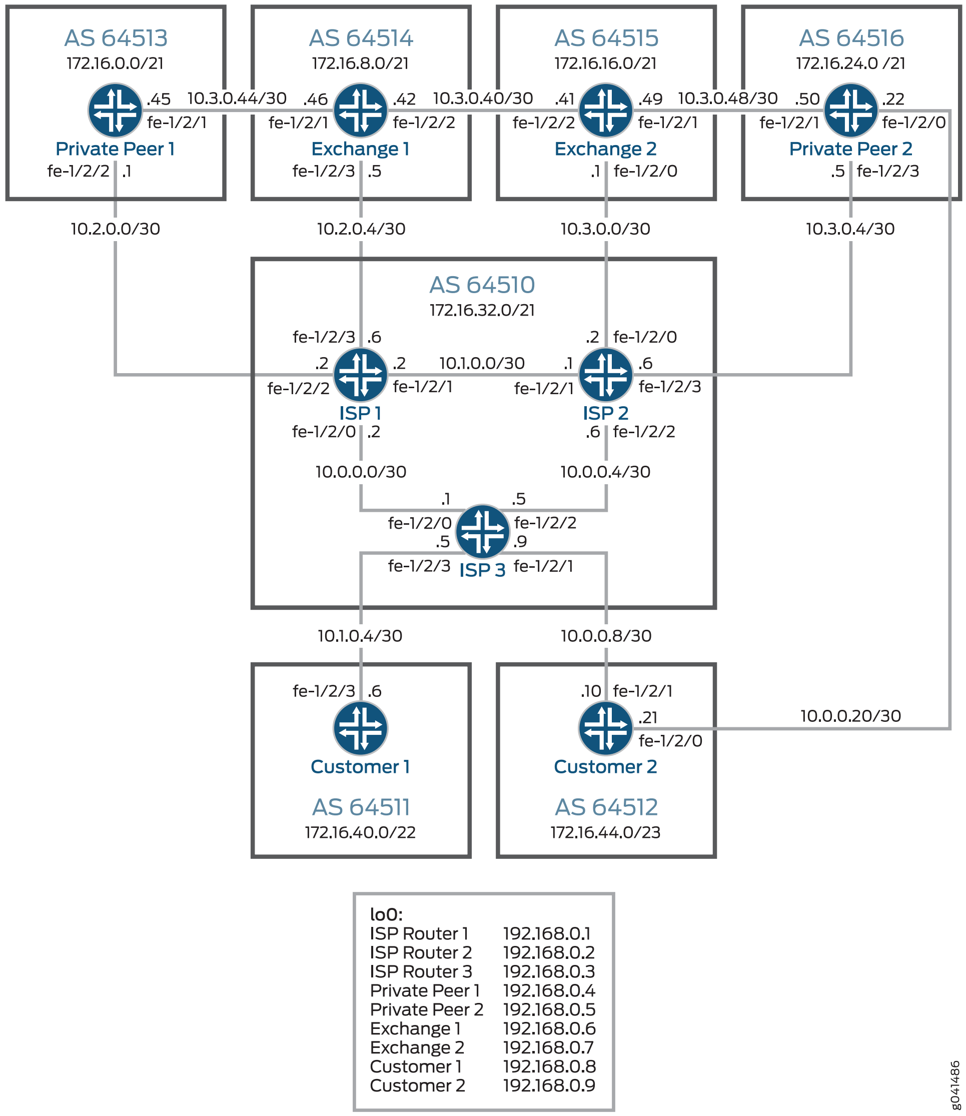
The ISP policies are configured in an outbound direction. That is, the example focuses on the routes that the ISP announces to its peers and customers, and includes the following:

1. The ISP has been assigned AS 64510 and the routing space of 172.16.32.0/21. With the exception of the two customer networks, all other customer routes are simulated with static routes.
2. The exchange peers are used for transit service to other portions of the Internet. This means that the ISP is accepting all routes (the full Internet routing table) from those BGP peers. To help maintain an optimized Internet routing table, the ISP is configured to advertise only two aggregate routes to the transit peers.
3. The ISP administrators want all data to the private peers to use the direct links. As a result, all the customer routes from the ISP are advertised to those private peers. These peers then advertise all their customer routes to the ISP.
4. Finally, each customer has a different set of requirements. Customer-1 requires a singe default route. Customer-2 requires specific routes.

Topology

Figure 13 on page 100 shows the sample network.

Figure 13: ISP Network Example



Set Commands for All Devices in the Topology

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device Customer-1

```
set interfaces fe-1/2/3 unit 0 description to_ISP-3
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.8/32
set protocols bgp group ext type external
set protocols bgp group ext export send-statics
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.1.0.5
```



```
set policy-options policy-statement send-statics term static-routes from protocol static
set policy-options policy-statement send-statics term static-routes then accept
set routing-options static route 172.16.40.0/25 reject
set routing-options static route 172.16.40.128/25 reject
set routing-options static route 172.16.41.0/25 reject
set routing-options static route 172.16.41.128/25 reject
set routing-options autonomous-system 64511
```

Device Customer-2

```
set interfaces fe-1/2/1 unit 0 description to_ISP-3
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.10/30
set interfaces fe-1/2/0 unit 0 description to-Private-Peer-2
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.21/30
set interfaces lo0 unit 0 family inet address 192.168.0.9/32
set protocols bgp group ext type external
set protocols bgp group ext import inbound-routes
set protocols bgp group ext export outbound-routes
set protocols bgp group ext neighbor 10.0.0.9 peer-as 64510
set protocols bgp group ext neighbor 10.0.0.22 peer-as 64516
set policy-options policy-statement inbound-routes term AS64510-primary from protocol
  bgp
set policy-options policy-statement inbound-routes term AS64510-primary from as-path
  AS64510-routes
set policy-options policy-statement inbound-routes term AS64510-primary then
  local-preference 200
set policy-options policy-statement inbound-routes term AS64510-primary then accept
set policy-options policy-statement inbound-routes term AS64516-backup from protocol
  bgp
set policy-options policy-statement inbound-routes term AS64516-backup from as-path
  AS64516-routes
set policy-options policy-statement inbound-routes term AS64516-backup then
  local-preference 50
set policy-options policy-statement inbound-routes term AS64516-backup then accept
set policy-options policy-statement outbound-routes term statics from protocol static
set policy-options policy-statement outbound-routes term statics then accept
set policy-options policy-statement outbound-routes term internal-bgp-routes from
  protocol bgp
set policy-options policy-statement outbound-routes term internal-bgp-routes from
  as-path my-own-routes
set policy-options policy-statement outbound-routes term internal-bgp-routes then
  accept
set policy-options policy-statement outbound-routes term no-transit then reject
set policy-options as-path my-own-routes "()"
set policy-options as-path AS64510-routes "64510 .*"
set policy-options as-path AS64516-routes "64516 .*"
set routing-options static route 172.16.44.0/26 reject
set routing-options static route 172.16.44.64/26 reject
set routing-options static route 172.16.44.128/26 reject
set routing-options static route 172.16.44.192/26 reject
set routing-options autonomous-system 64512
```

Device ISP-1

```
set interfaces fe-1/2/0 unit 0 description to_ISP-3
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 description to_ISP-2
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
```

```
set interfaces fe-1/2/2 unit 0 description to_Private-Peer-1
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.2/30
set interfaces fe-1/2/3 unit 0 description to_Exchange-1
set interfaces fe-1/2/3 unit 0 family inet address 10.2.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.1
set protocols bgp group int export internal-peers
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group to_64513 type external
set protocols bgp group to_64513 export private-peer
set protocols bgp group to_64513 peer-as 64513
set protocols bgp group to_64513 neighbor 10.2.0.1
set protocols bgp group to_64514 type external
set protocols bgp group to_64514 export exchange-peer
set protocols bgp group to_64514 peer-as 64514
set protocols bgp group to_64514 neighbor 10.2.0.5
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement exchange-peer term AS64510-Aggregate from
  protocol aggregate
set policy-options policy-statement exchange-peer term AS64510-Aggregate from
  route-filter 172.16.32.0/21 exact
set policy-options policy-statement exchange-peer term AS64510-Aggregate then accept
set policy-options policy-statement exchange-peer term Customer-2-Aggregate from
  protocol aggregate
set policy-options policy-statement exchange-peer term Customer-2-Aggregate from
  route-filter 172.16.40.0/22 exact
set policy-options policy-statement exchange-peer term Customer-2-Aggregate then
  accept
set policy-options policy-statement exchange-peer term reject-all-other-routes then
  reject
set policy-options policy-statement internal-peers term statics from protocol static
set policy-options policy-statement internal-peers term statics then accept
set policy-options policy-statement internal-peers term next-hop-self then next-hop self
set policy-options policy-statement private-peer term statics from protocol static
set policy-options policy-statement private-peer term statics then accept
set policy-options policy-statement private-peer term isp-and-customer-routes from
  protocol bgp
set policy-options policy-statement private-peer term isp-and-customer-routes from
  route-filter 172.16.32.0/21 orlonger
set policy-options policy-statement private-peer term isp-and-customer-routes then
  accept
set policy-options policy-statement private-peer term reject-all then reject
set routing-options static route 172.16.32.0/24 reject
set routing-options static route 172.16.33.0/24 reject
set routing-options aggregate route 172.16.32.0/21
set routing-options aggregate route 172.16.40.0/22
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device ISP-2

```
set interfaces fe-1/2/1 unit 0 description to_ISP-1
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
```

```
set interfaces fe-1/2/2 unit 0 description to_ISP-3
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.6/30
set interfaces fe-1/2/3 unit 0 description to_Private-Peer-2
set interfaces fe-1/2/3 unit 0 family inet address 10.3.0.6/30
set interfaces fe-1/2/0 unit 0 description to_Exchange-2
set interfaces fe-1/2/0 unit 0 family inet address 10.3.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int export internal-peers
set protocols bgp group int neighbor 192.168.0.1
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group AS-64516 type external
set protocols bgp group AS-64516 export private-peer
set protocols bgp group AS-64516 peer-as 64516
set protocols bgp group AS-64516 neighbor 10.3.0.5
set protocols bgp group AS-64515 type external
set protocols bgp group AS-64515 export exchange-peer
set protocols bgp group AS-64515 peer-as 64515
set protocols bgp group AS-64515 neighbor 10.3.0.1
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement exchange-peer term AS64510-Aggregate from
  protocol aggregate
set policy-options policy-statement exchange-peer term AS64510-Aggregate from
  route-filter 172.16.32.0/21 exact
set policy-options policy-statement exchange-peer term AS64510-Aggregate then accept
set policy-options policy-statement exchange-peer term Customer-2-Aggregate from
  protocol aggregate
set policy-options policy-statement exchange-peer term Customer-2-Aggregate from
  route-filter 172.16.44.0/23 exact
set policy-options policy-statement exchange-peer term Customer-2-Aggregate then
  accept
set policy-options policy-statement exchange-peer term reject-all-other-routes then
  reject
set policy-options policy-statement internal-peers term statics from protocol static
set policy-options policy-statement internal-peers term statics then accept
set policy-options policy-statement internal-peers term next-hop-self then next-hop self
set policy-options policy-statement private-peer term statics from protocol static
set policy-options policy-statement private-peer term statics then accept
set policy-options policy-statement private-peer term isp-and-customer-routes from
  protocol bgp
set policy-options policy-statement private-peer term isp-and-customer-routes from
  route-filter 172.16.32.0/21 orlonger
set policy-options policy-statement private-peer term isp-and-customer-routes then
  accept
set policy-options policy-statement private-peer term reject-all then reject
set routing-options static route 172.16.34.0/24 reject
set routing-options static route 172.16.35.0/24 reject
set routing-options aggregate route 172.16.44.0/23
set routing-options aggregate route 172.16.32.0/21
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device ISP-3

```
set interfaces fe-1/2/0 unit 0 description to_ISP-1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/2 unit 0 description to_ISP-2
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30
set interfaces fe-1/2/3 unit 0 description to_Customer-1
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30
set interfaces fe-1/2/1 unit 0 description to_Customer-2
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.9/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.3
set protocols bgp group int export internal-peers
set protocols bgp group int neighbor 192.168.0.1
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group to_64511 type external
set protocols bgp group to_64511 export customer-1-peer
set protocols bgp group to_64511 neighbor 10.1.0.6 peer-as 64511
set protocols bgp group to_64512 type external
set protocols bgp group to_64512 export customer-2-peer
set protocols bgp group to_64512 neighbor 10.0.0.10 peer-as 64512
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement customer-1-peer term defaut-route from route-filter
  0.0.0.0/0 exact
set policy-options policy-statement customer-1-peer term defaut-route then accept
set policy-options policy-statement customer-1-peer term reject-all-other-routes then
  reject
set policy-options policy-statement customer-2-peer term statics from protocol static
set policy-options policy-statement customer-2-peer term statics then accept
set policy-options policy-statement customer-2-peer term isp-and-customer-routes
  from protocol bgp
set policy-options policy-statement customer-2-peer term isp-and-customer-routes
  from route-filter 172.16.32.0/21 orlonger
set policy-options policy-statement customer-2-peer term isp-and-customer-routes then
  accept
set policy-options policy-statement customer-2-peer term default-route from route-filter
  0.0.0.0/0 exact
set policy-options policy-statement customer-2-peer term default-route then accept
set policy-options policy-statement customer-2-peer term reject-all-other-routes then
  reject
set policy-options policy-statement if-upstream-routes-exist term
  only-certain-contributing-routes from route-filter 172.16.8.0/21 exact
set policy-options policy-statement if-upstream-routes-exist term
  only-certain-contributing-routes then accept
set policy-options policy-statement if-upstream-routes-exist term reject-all-other-routes
  then reject
set policy-options policy-statement internal-peers term statics from protocol static
set policy-options policy-statement internal-peers term statics then accept
set policy-options policy-statement internal-peers term next then next-hop self
set routing-options static route 172.16.36.0/24 reject
set routing-options static route 172.16.37.0/24 reject
set routing-options static route 172.16.38.0/24 reject
set routing-options static route 172.16.39.0/24 reject
set routing-options generate route 0.0.0.0/0 policy if-upstream-routes-exist
set routing-options router-id 192.168.0.3
```



```
set routing-options autonomous-system 64510
```

Device Exchange-1

```
set interfaces fe-1/2/3 unit 0 description to_ISP-1
set interfaces fe-1/2/3 unit 0 family inet address 10.2.0.5/30
set interfaces fe-1/2/2 unit 0 description to_Exchange-2
set interfaces fe-1/2/2 unit 0 family inet address 10.3.0.42/30
set interfaces fe-1/2/1 unit 0 description to_Private-Peer-1
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.45/30
set interfaces lo0 unit 0 family inet address 192.168.0.6/32
set protocols bgp group ext type external
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.2.0.6
set protocols bgp group ext neighbor 10.3.0.41 peer-as 64515
set policy-options policy-statement send-static from protocol static
set policy-options policy-statement send-static then accept
set routing-options static route 172.16.8.0/21 reject
set routing-options autonomous-system 64514
```

Device Exchange-2

```
set interfaces fe-1/2/0 unit 0 description to_ISP-2
set interfaces fe-1/2/0 unit 0 family inet address 10.3.0.1/30
set interfaces fe-1/2/2 unit 0 description to_Exchange-1
set interfaces fe-1/2/2 unit 0 family inet address 10.3.0.41/30
set interfaces fe-1/2/1 unit 0 description to_Private-Peer-2
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.49/30
set interfaces lo0 unit 0 family inet address 192.168.0.7/32
set protocols bgp group ext type external
set protocols bgp group ext export outbound-routes
set protocols bgp group ext neighbor 10.3.0.2 peer-as 64510
set protocols bgp group ext neighbor 10.3.0.50 peer-as 64516
set protocols bgp group ext neighbor 10.3.0.42 peer-as 64514
set policy-options policy-statement outbound-routes term statics from protocol static
set policy-options policy-statement outbound-routes term statics then accept
set routing-options autonomous-system 64515
set routing-options static route 172.16.16.0/21 reject
```

Device Private-Peer-1

```
set interfaces fe-1/2/2 unit 0 description to_ISP-1
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.1/30
set interfaces fe-1/2/1 unit 0 description to_Exchange-1
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.46/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.2.0.2
set routing-options autonomous-system 64513
```

Device Private-Peer-2

```
set interfaces fe-1/2/3 unit 0 description to_ISP-2
set interfaces fe-1/2/3 unit 0 family inet address 10.3.0.5/30
set interfaces fe-1/2/0 unit 0 description to_Customer-1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.22/30
set interfaces fe-1/2/1 unit 0 description to_Exchange-2
set interfaces fe-1/2/1 unit 0 family inet address 10.3.0.50/30
set interfaces lo0 unit 0 family inet address 192.168.0.5/32
```



```
set protocols bgp group ext type external
set protocols bgp group ext export outbound-routes
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.3.0.6
set protocols bgp group to-64512 type external
set protocols bgp group to-64512 peer-as 64512
set protocols bgp group to-64512 neighbor 10.0.0.21
set protocols bgp group to-64512 export internal-routes
set protocols bgp group to-64515 type external
set protocols bgp group to-64515 export outbound-routes
set protocols bgp group to-64515 peer-as 64515
set protocols bgp group to-64515 neighbor 10.3.0.49
set policy-options policy-statement if-upstream-routes-exist term as-64515-routes from
  route-filter 172.16.16.0/21 exact
set policy-options policy-statement if-upstream-routes-exist term as-64515-routes then
  accept
set policy-options policy-statement if-upstream-routes-exist term reject-all-other-routes
  then reject
set policy-options policy-statement internal-routes term statics from protocol static
set policy-options policy-statement internal-routes term statics then accept
set policy-options policy-statement internal-routes term default-route from route-filter
  0.0.0.0/0 exact
set policy-options policy-statement internal-routes term default-route then accept
set policy-options policy-statement internal-routes term reject-all-other-routes then
  reject
set policy-options policy-statement outbound-routes term statics from protocol static
set policy-options policy-statement outbound-routes term statics then accept
set policy-options policy-statement outbound-routes term allowed-bgp-routes from
  as-path my-own-routes
set policy-options policy-statement outbound-routes term allowed-bgp-routes from
  as-path AS64512-routes
set policy-options policy-statement outbound-routes term allowed-bgp-routes then
  accept
set policy-options policy-statement outbound-routes term no-transit then reject
set policy-options as-path my-own-routes "()"
set policy-options as-path AS64512-routes 64512
set routing-options static route 172.16.24.0/25 reject
set routing-options static route 172.16.24.128/25 reject
set routing-options static route 172.16.25.0/26 reject
set routing-options static route 172.16.25.64/26 reject
set routing-options generate route 0.0.0.0/0 policy if-upstream-routes-exist
set routing-options autonomous-system 64516
```

Configuring Device Customer-1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Device Customer-1 has multiple static routes configured to simulate customer routes. These routes are sent to the ISP.

To configure Device Customer-1:

1. Configure the device interfaces.


```
[edit interfaces]
user@Customer-1# set fe-1/2/3 unit 0 description to_ISP-3
user@Customer-1# set fe-1/2/3 unit 0 family inet address 10.1.0.6/30

user@Customer-1# set lo0 unit 0 family inet address 192.168.0.8/32
```

2. Configure the static routes.

```
[edit routing-options static]
user@Customer-1# set route 172.16.40.0/25 reject
user@Customer-1# set route 172.16.40.128/25 reject
user@Customer-1# set route 172.16.41.0/25 reject
user@Customer-1# set route 172.16.41.128/25 reject
```

3. Configure the policy to send static routes.

```
[edit policy-options policy-statement send-statics term static-routes]
user@Customer-1# set from protocol static
user@Customer-1# set then accept
```

4. Configure the external BGP (EBGP) connection to the ISP.

```
[edit protocols bgp group ext]
user@Customer-1# set type external
user@Customer-1# set export send-statics
user@Customer-1# set peer-as 64510
user@Customer-1# set neighbor 10.1.0.5
```

5. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@Customer-1# set autonomous-system 64511
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@Customer-1# show interfaces
fe-1/2/1 {
  unit 0 {
    description to_ISP-3;
    family inet {
      address 10.1.0.6/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.8/32;
    }
```

```
  }
}
user@Customer-1# show protocols
bgp {
  group ext {
    type external;
    export send-statics;
    peer-as 64510;
    neighbor 10.1.0.5;
  }
}
user@Customer-1# show policy-options
policy-statement send-statics {
  term static-routes {
    from protocol static;
    then accept;
  }
}
user@Customer-1# show routing-options
static {
  route 172.16.40.0/25 reject;
  route 172.16.40.128/25 reject;
  route 172.16.41.0/25 reject;
  route 172.16.41.128/25 reject;
}
autonomous-system 64511;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Device Customer-2

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Device Customer-2 has two static routes configured to simulate customer routes. These routes are sent to the ISP. Customer-2 has a link to the ISP, as well as a link to AS 8000. This customer has requested specific customer routes from the ISP, as well as from AS 64516. Customer-2 wants to use the ISP for transit service to the Internet, and has requested a default route from the ISP.

To configure Device Customer-2:

1. Configure the device interfaces.

```
[edit interfaces]
user@Customer-2# set fe-1/2/1 unit 0 description to_ISP-3
user@Customer-2# set fe-1/2/1 unit 0 family inet address 10.0.0.10/30

user@Customer-2# set fe-1/2/0 unit 0 description to-Private-Peer-2
user@Customer-2# set fe-1/2/0 unit 0 family inet address 10.0.0.21/30
```



```
user@Customer-2# set lo0 unit 0 family inet address 192.168.0.9/32
```

2. Configure the static routes.

```
[edit routing-options static]
user@Customer-2# set route 172.16.44.0/26 reject
user@Customer-2# set route 172.16.44.64/26 reject
user@Customer-2# set route 172.16.44.128/26 reject
user@Customer-2# set route 172.16.44.192/26 reject
```

3. Configure the import routing policy.

   The route with the highest local preference value is preferred. Routes from the ISP are preferred over the same routes from Device Private-Peer-2

```
[edit policy-options policy-statement inbound-routes]
user@Customer-2# set term AS64510-primary from protocol bgp
user@Customer-2# set term AS64510-primary from as-path AS64510-routes
user@Customer-2# set term AS64510-primary then local-preference 200
user@Customer-2# set term AS64510-primary then accept

[edit policy-options policy-statement inbound-routes]
user@Customer-2# set term AS64516-backup from protocol bgp
user@Customer-2# set term AS64516-backup from as-path AS64516-routes
user@Customer-2# set term AS64516-backup then local-preference 50
user@Customer-2# set term AS64516-backup then accept

[edit policy-options]
user@Customer-2# set as-path AS64510-routes "64510 .*"
user@Customer-2# set as-path AS64516-routes "64516 .*"
```

4. Configure the export routing policy.

```
[edit policy-options policy-statement outbound-routes]
user@Customer-2# set term statics from protocol static
user@Customer-2# set term statics then accept

user@Customer-2# set term internal-bgp-routes from protocol bgp
user@Customer-2# set term internal-bgp-routes from as-path my-own-routes
user@Customer-2# set term internal-bgp-routes then accept
user@Customer-2# set term no-transit then reject

[edit policy-options]
user@Customer-2# set as-path my-own-routes "()"
```

5. Configure the external BGP (EBGP) connection to the ISP and to Device Private-Peer-2.

```
[edit protocols bgp group ext]
user@Customer-2# set type external
user@Customer-2# set import inbound-routes
user@Customer-2# set export outbound-routes
user@Customer-2# set neighbor 10.0.0.9 peer-as 64510
```

```
user@Customer-2# set neighbor 10.0.0.22 peer-as 64516
```

6. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@Customer-2# set autonomous-system 64512
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@Customer-2# show interfaces
fe-1/2/1 {
   unit 0 {
      description to_ISP-3;
      family inet {
         address 10.0.0.10/30;
      }
   }
}
fe-1/2/0 {
   unit 0 {
      description to-Private-Peer-2;
      family inet {
         address 10.0.0.21/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.9/32;
      }
   }
}
```

```
user@Customer-2# show protocols
bgp {
   group ext {
      type external;
      import inbound-routes;
      export outbound-routes;
      neighbor 10.0.0.9 {
         peer-as 64510;
      }
      neighbor 10.0.0.22 {
         peer-as 64516;
      }
   }
}
```

```
user@Customer-2# show policy-options
policy-statement inbound-routes {
```



```
    term AS64510-primary {
        from {
            protocol bgp;
            as-path AS64510-routes;
        }
        then {
            local-preference 200;
            accept;
        }
    }
    term AS64516-backup {
        from {
            protocol bgp;
            as-path AS64516-routes;
        }
        then {
            local-preference 50;
            accept;
        }
    }
}
policy-statement outbound-routes {
    term statics {
        from protocol static;
        then accept;
    }
    term internal-bgp-routes {
        from {
            protocol bgp;
            as-path my-own-routes;
        }
        then accept;
    }
    term no-transit {
        then reject;
    }
}
as-path my-own-routes "()";
as-path AS64510-routes "64510 .*";
as-path AS64516-routes "64516 .*";
```

```
user@Customer-2# show routing-options
static {
    route 172.16.44.0/26 reject;
    route 172.16.44.64/26 reject;
    route 172.16.44.128/26 reject;
    route 172.16.44.192/26 reject;
}
autonomous-system 64512;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Devices ISP-1 and ISP-2

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Device ISP-1 and Device ISP-2 each have two policies configured: The **private-peer** policy and the **exchange-peer** policy. Because of their similar configurations, this example shows the step-by-step configuration only for Device ISP-2.

On Device ISP-2, the private-peer policy sends the ISP customer routes to Device Private-Peer-2. The policy accepts all local static routes (local Device ISP-2 customers) and all BGP routes in the 172.16.32.0/21 range (advertised by other ISP routers). These two policy terms represent the ISP customer routes. The final policy term rejects all other routes, which includes the entire Internet routing table sent by the exchange peers. These routes do not need to be sent to Device Private-Peer-2 for two reasons:

- The peer already maintains a connection to Device Exchange-2 in our example, so the routes are redundant.
- The private peer wants customer routes only. The **private-peer** policy accomplishes this goal. The **exchange-peer** policy sends routes to Device Exchange-2.

In the example, only two routes need to be sent to Device Exchange-2:

- The aggregate route that represents the AS 64510 routing space of 172.16.32.0/21. This route is configured as an aggregate route locally and is advertised by the **exchange-peer** policy.
- The address space assigned to Customer-2, 172.16.44.0/23. This smaller aggregate route needs to be sent to Device Exchange-2 because the customer is also attached to the AS 64516 peer (Device Private-Peer-2).

Sending these two routes to Device Exchange-2 allows other networks in the Internet to reach the customer through either the ISP or the private peer. If just the private peer were to advertise the /23 network while the ISP maintained only its /21 aggregate, all traffic destined for the customer would transit AS 64516 only. Because the customer also wants routes from the ISP, the 172.16.44.0/23 route is announced by Device ISP-2. Like the larger aggregate route, the 172.16.44.0/23 route is configured locally and is advertised by the exchange-peer policy. The final term in that policy rejects all routes, including the specific customer networks of the ISP, the customer routes from Device Private-Peer-1, the customer routes from Device Private-Peer-2, and the routing table from Device Exchange-1. In essence, this final term prevents the ISP from performing transit services for the Internet at large.

To configure Device ISP-2:

1. Configure the device interfaces.

```
[edit interfaces]
user@ISP-2# set fe-1/2/1 unit 0 description to_ISP-1
user@ISP-2# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30
```



```
user@ISP-2# set fe-1/2/2 unit 0 description to_ISP-3
user@ISP-2# set fe-1/2/2 unit 0 family inet address 10.0.0.6/30

user@ISP-2# set fe-1/2/3 unit 0 description to_Private-Peer-2
user@ISP-2# set fe-1/2/3 unit 0 family inet address 10.3.0.6/30

user@ISP-2# set fe-1/2/0 unit 0 description to_Exchange-2
user@ISP-2# set fe-1/2/0 unit 0 family inet address 10.3.0.2/30

user@ISP-2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the interior gateway protocol (IGP).

```
[edit protocols ospf area 0.0.0.0]
user@ISP-2# set interface fe-1/2/2.0
user@ISP-2# set interface fe-1/2/1.0
user@ISP-2# set interface lo0.0 passive
```

3. Configure the static and aggregate routes.

```
[edit routing-options static]
user@ISP-2# set route 172.16.34.0/24 reject
user@ISP-2# set route 172.16.35.0/24 reject

[edit routing-options aggregate]
user@ISP-2# set route 172.16.44.0/23
user@ISP-2# set route 172.16.32.0/21
```

4. Configure the routing policies for the exchange peers.

```
[edit policy-options policy-statement exchange-peer]
user@ISP-2# set term AS64510-Aggregate from protocol aggregate
user@ISP-2# set term AS64510-Aggregate from route-filter 172.16.32.0/21 exact
user@ISP-2# set term AS64510-Aggregate then accept
user@ISP-2# set term Customer-2-Aggregate from protocol aggregate
user@ISP-2# set term Customer-2-Aggregate from route-filter 172.16.44.0/23 exact
user@ISP-2# set term Customer-2-Aggregate then accept
user@ISP-2# set term reject-all-other-routes then reject
```

5. Configure the routing policies for the internal peers.

```
[edit policy-options policy-statement internal-peers]
user@ISP-2# set term statics from protocol static
user@ISP-2# set term statics then accept
user@ISP-2# set term next-hop-self then next-hop self
```

6. Configure the routing policies for the private peer.

```
[edit policy-options policy-statement private-peer]
user@ISP-2# set term statics from protocol static
user@ISP-2# set term statics then accept
user@ISP-2# set term isp-and-customer-routes from protocol bgp
```

```
user@ISP-2# set term isp-and-customer-routes from route-filter 172.16.32.0/21
  orlonger
user@ISP-2# set term isp-and-customer-routes then accept
user@ISP-2# set term reject-all then reject
```

7. Configure the internal BGP (IBGP) connections to the other ISP devices.

```
[edit protocols bgp group int]
user@ISP-2# set type internal
user@ISP-2# set local-address 192.168.0.2
user@ISP-2# set export internal-peers
user@ISP-2# set neighbor 192.168.0.1
user@ISP-2# set neighbor 192.168.0.3
```

8. Configure the EBGP connections to the exchange peer and the private peer.

```
[edit protocols bgp group AS-64516]
user@ISP-2# set type external
user@ISP-2# set export private-peer
user@ISP-2# set peer-as 64516
user@ISP-2# set neighbor 10.3.0.5

[edit protocols bgp group AS-64515]
user@ISP-2# set type external
user@ISP-2# set export exchange-peer
user@ISP-2# set peer-as 64515
user@ISP-2# set neighbor 10.3.0.1
```

9. Configure the autonomous system (AS) number and the router ID.

```
[edit routing-options]
user@ISP-2# set router-id 192.168.0.2
user@ISP-2# set autonomous-system 64510
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@ISP-2# show interfaces
fe-1/2/0 {
  unit 0{
    description to_Exchange-2;
    family inet {
      address 10.3.0.2/30;
    }
  }
}
fe-1/2/1 {
  unit 0{
    description to_ISP-1;
    family inet {
      address 10.1.0.1/30;
```

```
        }
    }
}
fe-1/2/2 {
    unit 0 {
        description to_ISP-3;
        family inet {
            address 10.0.0.6/30;
        }
    }
}
fe-1/2/3 {
    unit 0 {
        description to_Private-Peer-2;
        family inet {
            address 10.3.0.6/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}

user@ISP-2# show protocols
bgp {
    group int {
        type internal;
        local-address 192.168.0.2;
        export internal-peers;
        neighbor 192.168.0.1;
        neighbor 192.168.0.3;
    }
    group AS-64516 {
        type external;
        export private-peer;
        peer-as 64516;
        neighbor 10.3.0.5;
    }
    group AS-64515 {
        type external;
        export exchange-peer;
        peer-as 64515;
        neighbor 10.3.0.1;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/2.0;
        interface fe-1/2/1.0;
        interface lo0.0 {
            passive;
        }
```

```
  }
}
user@ISP-2# show policy-options
policy-statement exchange-peer {
  term AS64510-Aggregate {
    from {
      protocol aggregate;
      route-filter 172.16.32.0/21 exact;
    }
    then accept;
  }
  term Customer-2-Aggregate {
    from {
      protocol aggregate;
      route-filter 172.16.44.0/23 exact;
    }
    then accept;
  }
  term reject-all-other-routes {
    then reject;
  }
}
policy-statement internal-peers {
  term statics {
    from protocol static;
    then accept;
  }
  term next-hop-self {
    then {
      next-hop self;
    }
  }
}
policy-statement private-peer {
  term statics {
    from protocol static;
    then accept;
  }
  term isp-and-customer-routes {
    from {
      protocol bgp;
      route-filter 172.16.32.0/21 orlonger;
    }
    then accept;
  }
  term reject-all {
    then reject;
  }
}
user@ISP-2# show routing-options
static {
  route 172.16.34.0/24 reject;
  route 172.16.35.0/24 reject;
}
aggregate {
```

```
    route 172.16.44.0/23;
    route 172.16.32.0/21;
  }
  router-id 192.168.0.2;
  autonomous-system 64510;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Device ISP-3

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

On Device ISP-3, a separate policy is in place for each customer. The default route for Customer-1 is being sent by the **customer-1-peer** policy. This policy finds the 0.0.0.0/0 default route in inet.0 and accepts it. The policy also rejects all other routes, thereby not sending all BGP routes on the ISP router. The **customer-2-peer** policy is for Customer-2 and contains the same policy terms, which also send the default route and no other transit BGP routes. The additional terms in the **customer-2-peer** policy send the ISP customer routes to Customer-2. Because there are local static routes on Device ISP-3 that represent local customers, these routes are sent as well as all other internal routes announced to the local router by the other ISP routers.

If the upstream route from Device Exchange-1 (172.16.8.0/21) is present, Device ISP-3 generates a default route.

To configure Device ISP-3:

1. Configure the device interfaces.

```
[edit interfaces]
user@ISP-3# set fe-1/2/0 unit 0 description to_ISP-1
user@ISP-3# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30

user@ISP-3# set fe-1/2/2 unit 0 description to_ISP-2
user@ISP-3# set fe-1/2/2 unit 0 family inet address 10.0.0.5/30

user@ISP-3# set fe-1/2/3 unit 0 description to_Customer-1
user@ISP-3# set fe-1/2/3 unit 0 family inet address 10.1.0.5/30

user@ISP-3# set fe-1/2/1 unit 0 description to_Customer-2
user@ISP-3# set fe-1/2/1 unit 0 family inet address 10.0.0.9/30

user@ISP-3# set lo0 unit 0 family inet address 192.168.0.3/32
```

2. Configure the interior gateway protocol (IGP).

```
[edit protocols ospf area 0.0.0.0]
user@ISP-3# set interface fe-1/2/0.0
user@ISP-3# set interface fe-1/2/2.0
user@ISP-3# set interface lo0.0 passive
```

3. Configure the static routes.

   ```
   [edit routing-options static]
   user@ISP-3# set route 172.16.36.0/24 reject
   user@ISP-3# set route 172.16.37.0/24 reject
   user@ISP-3# set route 172.16.38.0/24 reject
   user@ISP-3# set route 172.16.39.0/24 reject
   ```

4. Configure a routing policy that generates a default static route only if a certain upstream route exists.

   ```
   [edit policy-options policy-statement if-upstream-routes-exist term
     only-certain-contributing-routes]
   user@ISP-3# set from route-filter 172.16.8.0/21 exact
   user@ISP-3# set then accept

   [edit policy-options policy-statement if-upstream-routes-exist]
   user@ISP-3# set term reject-all-other-routes then reject

   [edit routing-options generate route 0.0.0.0/0]
   user@ISP-3# set policy if-upstream-routes-exist
   ```

5. Configure the routing policy for Customer-1.

   ```
   [edit policy-options policy-statement customer-1-peer]
   user@ISP-3# set term defaut-route from route-filter 0.0.0.0/0 exact
   user@ISP-3# set term defaut-route then accept
   user@ISP-3# set term reject-all-other-routes then reject
   ```

6. Configure the routing policy for Customer-2.

   ```
   [edit policy-options policy-statement customer-2-peer]
   user@ISP-3# set term statics from protocol static
   user@ISP-3# set term statics then accept
   user@ISP-3# set term isp-and-customer-routes from protocol bgp
   user@ISP-3# set term isp-and-customer-routes from route-filter 172.16.32.0/21
     orlonger
   user@ISP-3# set term isp-and-customer-routes then accept
   user@ISP-3# set term default-route from route-filter 0.0.0.0/0 exact
   user@ISP-3# set term default-route then accept
   user@ISP-3# set term reject-all-other-routes then reject
   ```

7. Configure the routing policies for the internal peers.

   ```
   [edit policy-options policy-statement internal-peers]
   user@ISP-3# set term statics from protocol static
   user@ISP-3# set term statics then accept
   user@ISP-3# set term next then next-hop self
   ```

8. Configure the internal BGP (IBGP) connections to the other ISP devices.

   ```
   [edit protocols bgp group int]
   user@ISP-3# set type internal
   ```

```
user@ISP-3# set local-address 192.168.0.3
user@ISP-3# set export internal-peers
user@ISP-3# set neighbor 192.168.0.1
user@ISP-3# set neighbor 192.168.0.2
```

9. Configure the EBGP connections to the customer peers.

```
[edit protocols bgp group to_64511]
user@ISP-3# set type external
user@ISP-3# set export customer-1-peer
user@ISP-3# set neighbor 10.1.0.6 peer-as 64511

[edit protocols bgp group to_64512]
user@ISP-3# set type external
user@ISP-3# set export customer-2-peer
user@ISP-3# set neighbor 10.0.0.10 peer-as 64512
```

10. Configure the autonomous system (AS) number and the router ID.

```
[edit routing-options]
user@ISP-3# set router-id 192.168.0.3
user@ISP-3# set autonomous-system 64510
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@ISP-3# show interfaces
fe-1/2/0 {
  unit 0 {
    description to_ISP-1;
    family inet {
      address 10.0.0.1/30;
    }
  }
}
fe-1/2/1 {
  unit 0 {
    description to_Customer-2;
    family inet {
      address 10.0.0.9/30;
    }
  }
}
fe-1/2/2 {
  unit 0 {
    description to_ISP-2;
    family inet {
      address 10.0.0.5/30;
    }
  }
}
```

```
fe-1/2/3 {
   unit 0 {
      description to_Customer-1;
      family inet {
         address 10.1.0.5/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.3/32;
      }
   }
}
```

```
user@ISP-3# show protocols
bgp {
   group int {
      type internal;
      local-address 192.168.0.3;
      export internal-peers;
      neighbor 192.168.0.1;
      neighbor 192.168.0.2;
   }
   group to_64511 {
      type external;
      export customer-1-peer;
      neighbor 10.1.0.6 {
         peer-as 64511;
      }
   }
   group to_64512 {
      type external;
      export customer-2-peer;
      neighbor 10.0.0.10 {
         peer-as 64512;
      }
   }
}
ospf {
   area 0.0.0.0 {
      interface fe-1/2/0.0;
      interface fe-1/2/2.0;
      interface lo0.0 {
         passive;
      }
   }
}
```

```
user@ISP-3# show policy-options
policy-statement customer-1-peer {
   term defaut-route {
      from {
         route-filter 0.0.0.0/0 exact;
      }
      then accept;
```



```
  }
  term reject-all-other-routes {
    then reject;
  }
}
policy-statement customer-2-peer {
  term statics {
    from protocol static;
    then accept;
  }
  term isp-and-customer-routes {
    from {
      protocol bgp;
      route-filter 172.16.32.0/21 orlonger;
    }
    then accept;
  }
  term default-route {
    from {
      route-filter 0.0.0.0/0 exact;
    }
    then accept;
  }
  term reject-all-other-routes {
    then reject;
  }
}
policy-statement if-upstream-routes-exist {
  term only-certain-contributing-routes {
    from {
      route-filter 172.16.8.0/21 exact;
    }
    then accept;
  }
  term reject-all-other-routes {
    then reject;
  }
}
policy-statement internal-peers {
  term statics {
    from protocol static;
    then accept;
  }
  term next {
    then {
      next-hop self;
    }
  }
}
```

```
user@ISP-3# show routing-options
static {
  route 172.16.36.0/24 reject;
  route 172.16.37.0/24 reject;
  route 172.16.38.0/24 reject;
  route 172.16.39.0/24 reject;
```

```
}
generate {
   route 0.0.0.0/0 policy if-upstream-routes-exist;
}
router-id 192.168.0.3;
autonomous-system 64510;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Device Exchange-2

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Device Exchange-2 exchanges all BGP routes with all BGP peers. The outbound-routes policy for Device Exchange-2 advertises locally defined static routes using BGP. The exclusion of a final **then reject** term causes the default BGP export policy to take effect, which is to send all BGP routes to all external BGP peers.

To configure Device Exchange-2:

1. Configure the device interfaces.

```
[edit interfaces]
user@Exchange-2# set fe-1/2/0 unit 0 description to_ISP-2
user@Exchange-2# set fe-1/2/0 unit 0 family inet address 10.3.0.1/30

user@Exchange-2# set fe-1/2/2 unit 0 description to_Exchange-1
user@Exchange-2# set fe-1/2/2 unit 0 family inet address 10.3.0.41/30

user@Exchange-2# set fe-1/2/1 unit 0 description to_Private-Peer-2
user@Exchange-2# set fe-1/2/1 unit 0 family inet address 10.3.0.49/30

user@Exchange-2# set lo0 unit 0 family inet address 192.168.0.7/32
```

2. Configure the static routes.

```
[edit routing-options static]
set route 172.16.16.0/21 reject
```

3. Configure a routing policy that generates a default static route only if certain internal routes exist.

```
[edit policy-options policy-statement outbound-routes term statics]
user@Exchange-2# set from protocol static
user@Exchange-2# set then accept
```

4. Configure the EBGP connections to the customer peers.

```
[edit protocols bgp group ext]
user@Exchange-2# set type external
user@Exchange-2# set export outbound-routes
```



```
user@Exchange-2# set neighbor 10.3.0.2 peer-as 64510
user@Exchange-2# set neighbor 10.3.0.50 peer-as 64516
user@Exchange-2# set neighbor 10.3.0.42 peer-as 64514
```

5. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@Exchange-2# set autonomous-system 64515
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@Exchange-2 show interfaces
fe-1/2/0 {
  unit 0 {
    description to_ISP-2;
    family inet {
      address 10.3.0.1/30;
    }
  }
}
fe-1/2/1 {
  unit 0 {
    description to_Private-Peer-2;
    family inet {
      address 10.3.0.49/30;
    }
  }
}
fe-1/2/2 {
  unit 0 {
    description to_Exchange-1;
    family inet {
      address 10.3.0.41/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.7/32;
    }
  }
}
```

```
user@Exchange-2# show protocols
bgp {
  group ext {
    type external;
    export outbound-routes;
    neighbor 10.3.0.2 {
      peer-as 64510;
```

```
        }
        neighbor 10.3.0.50 {
            peer-as 64516;
        }
        neighbor 10.3.0.42 {
            peer-as 64514;
        }
    }
}
```

```
user@Exchange-2# show policy-options
policy-statement outbound-routes {
    term statics {
        from protocol static;
        then accept;
    }
}
```

```
user@Exchange-2# show routing-options
static {
    route 172.16.16.0/21 reject;
}
autonomous-system 64515;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Device Private-Peer-2

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

Device Private-Peer-2 performs two main functions:

- Advertises routes local to AS 64516 to both the exchange peers and the ISP routers. The **outbound-routes** policy advertises the local static routes (that is, customers) on the router, and also advertises all routes learned by BGP that originated in either AS 64516 or AS 64512. These routes include other AS 64516 customer routes in addition to the AS 64512 customer. The AS routes are identified by an AS path regular expression match criteria in the policy.
- Advertises the 0.0.0.0/0 default route to the AS 64512 customer router. To accomplish this, the private peer creates a generated route for 0.0.0.0/0 locally on the router. This generated route is further assigned a policy called **if-upstream-routes-exist**, which allows only certain routes to contribute to the generated route, making it an active route in the routing table. Once the route is active, it can be sent to the AS 64512 router using BGP and the configured policies. The **if-upstream-routes-exist** policy accepts only the 172.16.32.0/21 route from Device Exchange-2, and rejects all other routes. If the 172.16.32.0/21 route is withdrawn by the exchange peer, the private peer loses the 0.0.0.0/0 default route and withdraws the default route from the AS 64512 customer router.

To configure Device Private-Peer-2:

1. Configure the device interfaces.

```
[edit interfaces]
user@Private-Peer-2# set fe-1/2/3 unit 0 description to_ISP-2
user@Private-Peer-2# set fe-1/2/3 unit 0 family inet address 10.3.0.5/30

user@Private-Peer-2# set fe-1/2/0 unit 0 description to_Customer-1
user@Private-Peer-2# set fe-1/2/0 unit 0 family inet address 10.0.0.22/30

user@Private-Peer-2# set fe-1/2/1 unit 0 description to_Exchange-2
user@Private-Peer-2# set fe-1/2/1 unit 0 family inet address 10.3.0.50/30

user@Private-Peer-2# set lo0 unit 0 family inet address 192.168.0.5/32
```

2. Configure the static routes.

```
[edit routing-options static]
user@Private-Peer-2# set route 172.16.24.0/25 reject
user@Private-Peer-2# set route 172.16.24.128/25 reject
user@Private-Peer-2# set route 172.16.25.0/26 reject
user@Private-Peer-2# set route 172.16.25.64/26 reject
```

3. Configure a routing policy that generates a default static route only if certain internal routes exist.

```
[edit policy-options policy-statement if-upstream-routes-exist]
user@Private-Peer-2# set term as-64515-routes from route-filter 172.16.16.0/21
   exact
user@Private-Peer-2# set term as-64515-routes then accept
user@Private-Peer-2# set term reject-all-other-routes then reject
```

```
[edit routing-options generate route 0.0.0.0/0]
user@Private-Peer-2# set policy if-upstream-routes-exist
```

4. Configure the routing policy that advertises local static routes and the default route.

```
[edit policy-options policy-statement internal-routes]
user@Private-Peer-2# set term statics from protocol static
user@Private-Peer-2# set term statics then accept
user@Private-Peer-2# set term default-route from route-filter 0.0.0.0/0 exact
user@Private-Peer-2# set term default-route then accept
user@Private-Peer-2# set term reject-all-other-routes then reject
```

5. Configure the routing policy that advertises local customer routes.

```
[edit policy-options policy-statement outbound-routes]
user@Private-Peer-2# set term statics from protocol static
user@Private-Peer-2# set term statics then accept
user@Private-Peer-2# set term allowed-bgp-routes from as-path my-own-routes
user@Private-Peer-2# set term allowed-bgp-routes from as-path AS64512-routes
user@Private-Peer-2# set term allowed-bgp-routes then accept
user@Private-Peer-2# set term no-transit then reject
```

```
[edit policy-options]
user@Private-Peer-2# set as-path my-own-routes "()"
user@Private-Peer-2# set as-path AS64512-routes 64512
```

6. Configure the EBGP connection to Customer-2.

```
[edit protocols bgp group to-64512]
user@Private-Peer-2# set type external
user@Private-Peer-2# set export internal-routes
user@Private-Peer-2# set peer-as 64512
user@Private-Peer-2# set neighbor 10.0.0.21
```

7. Configure the EBGP connection to Device Exchange-2.

```
[edit protocols bgp group to-64515]
user@Private-Peer-2# set type external
user@Private-Peer-2# set export outbound-routes
user@Private-Peer-2# set peer-as 64515
user@Private-Peer-2# set neighbor 10.3.0.49
```

8. Configure the EBGP connections to the ISP.

```
[edit protocols bgp group ext]
```



```
user@Private-Peer-2# set type external
user@Private-Peer-2# set export outbound-routes
user@Private-Peer-2# set peer-as 64510
user@Private-Peer-2# set neighbor 10.3.0.6
```

9. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@Private-Peer-2# set autonomous-system 64516
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@Private-Peer-2# show interfaces
fe-1/2/0 {
  unit 0 {
    description to_Customer-1;
    family inet {
      address 10.0.0.22/30;
    }
  }
}
fe-1/2/1 {
  unit 0 {
    description to_Exchange-2;
    family inet {
      address 10.3.0.50/30;
    }
  }
}
fe-1/2/3 {
  unit 0 {
    description to_ISP-2;
    family inet {
      address 10.3.0.5/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.5/32;
    }
  }
}
```

```
user@Private-Peer-2# show protocols
bgp {
  group ext {
    type external;
    export outbound-routes;
    peer-as 64510;
```

```
        neighbor 10.3.0.6;
    }
    group to-64512 {
        type external;
        export internal-routes;
        peer-as 64512;
        neighbor 10.0.0.21;
    }
    group to-64515 {
        type external;
        export outbound-routes;
        peer-as 64515;
        neighbor 10.3.0.49;
    }
}
```

```
user@Private-Peer-2# show policy-options
policy-statement if-upstream-routes-exist {
    term as-64515-routes {
        from {
            route-filter 172.16.16.0/21 exact;
        }
        then accept;
    }
    term reject-all-other-routes {
        then reject;
    }
}
policy-statement internal-routes {
    term statics {
        from protocol static;
        then accept;
    }
    term default-route {
        from {
            route-filter 0.0.0.0/0 exact;
        }
        then accept;
    }
    term reject-all-other-routes {
        then reject;
    }
}
policy-statement outbound-routes {
    term statics {
        from protocol static;
        then accept;
    }
    term allowed-bgp-routes {
        from as-path [ my-own-routes AS64512-routes ];
        then accept;
    }
    term no-transit {
        then reject;
    }
}
```



```
as-path my-own-routes "()";
as-path AS64512-routes 64512;
```

```
user@Private-Peer-2# show routing-options
static {
  route 172.16.24.0/25 reject;
  route 172.16.24.128/25 reject;
  route 172.16.25.0/26 reject;
  route 172.16.25.64/26 reject;
}
generate {
  route 0.0.0.0/0 policy if-upstream-routes-exist;
}
autonomous-system 64516;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the Routes on Device Customer-1

Purpose On Device Customer-1, check the routes in the routing table.

Action user@Customer-1> **show route**

```
inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 00:09:25, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.1.0.5 via fe-1/2/3.0
10.1.0.4/30        *[Direct/0] 23:50:20
                    > via fe-1/2/3.0
10.1.0.6/32        *[Local/0] 5d 21:56:47
                      Local via fe-1/2/3.0
172.16.40.0/25     *[Static/5] 22:59:04
                      Reject
172.16.40.128/25   *[Static/5] 22:59:04
                      Reject
172.16.41.0/25     *[Static/5] 22:59:04
                      Reject
172.16.41.128/25   *[Static/5] 22:59:04
                      Reject
192.168.0.8/32     *[Direct/0] 5d 21:25:45
                    > via lo0.0
```

Meaning Device Customer-1 has its four static routes, and it has learned the default route through BGP.

Verifying the Routes on Device Customer-2

Purpose On Device Customer-2, check the routes in the routing table.

Action

```
user@Customer-2> show route
inet.0: 22 destinations, 23 routes (22 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 00:10:35, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
                    [BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
10.0.0.8/30        *[Direct/0] 23:51:29
                    > via fe-1/2/0.10
10.0.0.10/32       *[Local/0] 23:52:49
                      Local via fe-1/2/0.10
10.0.0.20/30       *[Direct/0] 23:52:49
                    > via fe-1/2/0.0
10.0.0.21/32       *[Local/0] 23:52:49
                      Local via fe-1/2/0.0
172.16.24.0/25     *[BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
172.16.24.128/25   *[BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
172.16.25.0/26     *[BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
172.16.25.64/26    *[BGP/170] 04:58:09, localpref 50
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.22 via fe-1/2/0.0
172.16.32.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.33.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.34.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.35.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.36.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.37.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.38.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.39.0/24     *[BGP/170] 22:38:47, localpref 200
                      AS path: 64510 I, validation-state: unverified
                    > to 10.0.0.9 via fe-1/2/0.10
172.16.44.0/26     *[Static/5] 22:57:28
                      Reject
172.16.44.64/26    *[Static/5] 22:57:28
                      Reject
172.16.44.128/26   *[Static/5] 22:57:28
                      Reject
```



```
172.16.44.192/26   *[Static/5] 22:57:28
                      Reject
192.168.0.9/32     *[Direct/0] 23:52:49
                    > via lo0.0
```

Meaning Device Customer-2 has learned the default route through its session with the ISP and also through its session with the private peer. The route learned from the ISP is preferred because it has a higher local preference.

Verifying the Routes on Device ISP-1

Purpose On Device ISP-1, check the routes in the routing table.

Action

```
user@ISP-1> show route
inet.0: 42 destinations, 53 routes (42 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 22:44:26, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
10.0.0.0/30        *[Direct/0] 23:52:01
                    > via fe-1/2/0.0
10.0.0.2/32        *[Local/0] 23:52:01
                      Local via fe-1/2/0.0
10.0.0.4/30        *[OSPF/10] 23:51:06, metric 2
                      to 10.1.0.1 via fe-1/2/1.0
                    > to 10.0.0.1 via fe-1/2/0.0
10.0.0.20/30       *[BGP/170] 23:50:55, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                    [BGP/170] 23:51:28, localpref 100
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
10.1.0.0/30        *[Direct/0] 23:52:01
                    > via fe-1/2/1.0
10.1.0.2/32        *[Local/0] 23:52:01
                      Local via fe-1/2/1.0
10.2.0.0/30        *[Direct/0] 23:52:01
                    > via fe-1/2/2.0
10.2.0.2/32        *[Local/0] 23:52:01
                      Local via fe-1/2/2.0
10.2.0.4/30        *[Direct/0] 23:52:00
                    > via fe-1/2/3.0
10.2.0.6/32        *[Local/0] 23:52:00
                      Local via fe-1/2/3.0
10.3.0.4/30        *[BGP/170] 23:51:28, localpref 100
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
10.3.0.48/30       *[BGP/170] 23:50:55, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
172.16.8.0/21      *[BGP/170] 00:11:08, localpref 100
                      AS path: 64514 I, validation-state: unverified
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.16.0/21     *[BGP/170] 02:02:10, localpref 100, from 192.168.0.2
                      AS path: 64515 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                    [BGP/170] 02:02:10, localpref 100
                      AS path: 64514 64515 I, validation-state: unverified
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.24.0/25     *[BGP/170] 23:06:33, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                    [BGP/170] 23:06:33, localpref 100
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
172.16.24.128/25   *[BGP/170] 23:06:33, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                    [BGP/170] 23:06:33, localpref 100
```



```
                           AS path: 64514 64515 64516 I, validation-state: unverified

                         > to 10.2.0.5 via fe-1/2/3.0
172.16.25.0/26      *[BGP/170] 23:06:33, localpref 100, from 192.168.0.2
                           AS path: 64516 I, validation-state: unverified
                         > to 10.1.0.1 via fe-1/2/1.0
                         [BGP/170] 23:06:33, localpref 100
                           AS path: 64514 64515 64516 I, validation-state: unverified

                         > to 10.2.0.5 via fe-1/2/3.0
172.16.25.64/26     *[BGP/170] 23:06:33, localpref 100, from 192.168.0.2
                           AS path: 64516 I, validation-state: unverified
                         > to 10.1.0.1 via fe-1/2/1.0
                         [BGP/170] 23:06:33, localpref 100
                           AS path: 64514 64515 64516 I, validation-state: unverified

                         > to 10.2.0.5 via fe-1/2/3.0
172.16.32.0/21      *[Aggregate/130] 22:44:27
                           Reject
172.16.32.0/24      *[Static/5] 22:44:27
                           Reject
172.16.33.0/24      *[Static/5] 22:44:27
                           Reject
172.16.34.0/24      *[BGP/170] 22:39:20, localpref 100, from 192.168.0.2
                           AS path: I, validation-state: unverified
                         > to 10.1.0.1 via fe-1/2/1.0
172.16.35.0/24      *[BGP/170] 22:39:20, localpref 100, from 192.168.0.2
                           AS path: I, validation-state: unverified
                         > to 10.1.0.1 via fe-1/2/1.0
172.16.36.0/24      *[BGP/170] 22:39:20, localpref 100, from 192.168.0.3
                           AS path: I, validation-state: unverified
                         > to 10.0.0.1 via fe-1/2/0.0
172.16.37.0/24      *[BGP/170] 22:39:20, localpref 100, from 192.168.0.3
                           AS path: I, validation-state: unverified
                         > to 10.0.0.1 via fe-1/2/0.0
172.16.38.0/24      *[BGP/170] 22:39:20, localpref 100, from 192.168.0.3
                           AS path: I, validation-state: unverified
                         > to 10.0.0.1 via fe-1/2/0.0
172.16.39.0/24      *[BGP/170] 22:39:20, localpref 100, from 192.168.0.3
                           AS path: I, validation-state: unverified
                         > to 10.0.0.1 via fe-1/2/0.0
172.16.40.0/22      *[Aggregate/130] 22:44:27
                           Reject
172.16.40.0/25      *[BGP/170] 23:00:47, localpref 100, from 192.168.0.3
                           AS path: 64511 I, validation-state: unverified
                         > to 10.0.0.1 via fe-1/2/0.0
172.16.40.128/25    *[BGP/170] 23:00:47, localpref 100, from 192.168.0.3
                           AS path: 64511 I, validation-state: unverified
                         > to 10.0.0.1 via fe-1/2/0.0
172.16.41.0/25      *[BGP/170] 23:00:47, localpref 100, from 192.168.0.3
                           AS path: 64511 I, validation-state: unverified
                         > to 10.0.0.1 via fe-1/2/0.0
172.16.41.128/25    *[BGP/170] 23:00:47, localpref 100, from 192.168.0.3
                           AS path: 64511 I, validation-state: unverified
                         > to 10.0.0.1 via fe-1/2/0.0
172.16.44.0/26      *[BGP/170] 22:58:01, localpref 100, from 192.168.0.3
                           AS path: 64512 I, validation-state: unverified
                         > to 10.0.0.1 via fe-1/2/0.0
                         [BGP/170] 22:58:01, localpref 100
                           AS path: 64514 64515 64516 64512 I, validation-state:
unverified
```



```
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.44.64/26    *[BGP/170] 22:58:01, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
                    [BGP/170] 22:58:01, localpref 100
                      AS path: 64514 64515 64516 64512 I, validation-state:
unverified
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.44.128/26   *[BGP/170] 22:58:01, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
                    [BGP/170] 22:58:01, localpref 100
                      AS path: 64514 64515 64516 64512 I, validation-state:
unverified
                    > to 10.2.0.5 via fe-1/2/3.0
172.16.44.192/26   *[BGP/170] 22:58:01, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.1 via fe-1/2/0.0
                    [BGP/170] 22:58:01, localpref 100
                      AS path: 64514 64515 64516 64512 I, validation-state:
unverified
                    > to 10.2.0.5 via fe-1/2/3.0
192.168.0.1/32     *[Direct/0] 23:52:01
                    > via lo0.0
192.168.0.2/32     *[OSPF/10] 23:51:06, metric 1
                    > to 10.1.0.1 via fe-1/2/1.0
192.168.0.3/32     *[OSPF/10] 23:51:06, metric 1
                    > to 10.0.0.1 via fe-1/2/0.0
192.168.0.5/32     *[BGP/170] 23:50:55, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.1.0.1 via fe-1/2/1.0
                    [BGP/170] 23:51:28, localpref 100
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.2.0.5 via fe-1/2/3.0
172.16.233.5/32       *[OSPF/10] 23:52:07, metric 1
                      MultiRecv
```

Verifying the Routes on Device ISP-2

Purpose On Device ISP-2, check the routes in the routing table.

Action

```
user@ISP-2> show route
inet.0: 41 destinations, 59 routes (41 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[BGP/170] 22:45:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
10.0.0.0/30        *[OSPF/10] 23:52:25, metric 2
                      to 10.0.0.5 via fe-1/2/2.0
                    > to 10.1.0.2 via fe-1/2/1.0
10.0.0.4/30        *[Direct/0] 23:53:21
                    > via fe-1/2/2.0
10.0.0.6/32        *[Local/0] 23:53:23
                      Local via fe-1/2/2.0
10.0.0.20/30       *[BGP/170] 23:53:11, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:53:09, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
10.1.0.0/30        *[Direct/0] 23:53:19
                    > via fe-1/2/1.0
10.1.0.1/32        *[Local/0] 23:53:23
                      Local via fe-1/2/1.0
10.3.0.0/30        *[Direct/0] 23:53:22
                    > via fe-1/2/0.0
10.3.0.2/32        *[Local/0] 23:53:23
                      Local via fe-1/2/0.0
10.3.0.4/30        *[Direct/0] 23:53:23
                    > via fe-1/2/3.0
                    [BGP/170] 23:53:11, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:53:09, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
                    [BGP/170] 23:52:13, localpref 100, from 192.168.0.1
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.1.0.2 via fe-1/2/1.0
10.3.0.6/32        *[Local/0] 23:53:23
                      Local via fe-1/2/3.0
10.3.0.48/30       *[BGP/170] 23:53:11, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
172.16.8.0/21      *[BGP/170] 00:12:26, localpref 100, from 192.168.0.1
                      AS path: 64514 I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
                    [BGP/170] 00:12:26, localpref 100
                      AS path: 64515 64514 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.16.0/21     *[BGP/170] 02:03:28, localpref 100
                      AS path: 64515 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.24.0/25     *[BGP/170] 23:07:51, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:07:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
```



```
172.16.24.128/25   *[BGP/170] 23:07:51, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:07:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.25.0/26     *[BGP/170] 23:07:51, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:07:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.25.64/26    *[BGP/170] 23:07:51, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:07:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.32.0/21     *[Aggregate/130] 22:40:38
                      Reject
172.16.32.0/24     *[BGP/170] 22:45:44, localpref 100, from 192.168.0.1
                      AS path: I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
172.16.33.0/24     *[BGP/170] 22:45:44, localpref 100, from 192.168.0.1
                      AS path: I, validation-state: unverified
                    > to 10.1.0.2 via fe-1/2/1.0
172.16.34.0/24     *[Static/5] 22:40:38
                      Reject
172.16.35.0/24     *[Static/5] 22:40:38
                      Reject
172.16.36.0/24     *[BGP/170] 22:40:38, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.37.0/24     *[BGP/170] 22:40:38, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.38.0/24     *[BGP/170] 22:40:38, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.39.0/24     *[BGP/170] 22:40:38, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.40.0/25     *[BGP/170] 23:02:05, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.40.128/25   *[BGP/170] 23:02:05, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.41.0/25     *[BGP/170] 23:02:05, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.41.128/25   *[BGP/170] 23:02:05, localpref 100, from 192.168.0.3
                      AS path: 64511 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
172.16.44.0/23     *[Aggregate/130] 22:40:38
                      Reject
172.16.44.0/26     *[BGP/170] 22:59:19, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
```



```
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.1 via fe-1/2/0.0
172.16.44.64/26    *[BGP/170] 22:59:19, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.1 via fe-1/2/0.0
172.16.44.128/26   *[BGP/170] 22:59:19, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.1 via fe-1/2/0.0
172.16.44.192/26   *[BGP/170] 22:59:19, localpref 100, from 192.168.0.3
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.5 via fe-1/2/2.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 22:59:19, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.1 via fe-1/2/0.0
192.168.0.1/32     *[OSPF/10] 23:52:25, metric 1
                    > to 10.1.0.2 via fe-1/2/1.0
192.168.0.2/32     *[Direct/0] 23:53:23
                    > via lo0.0
192.168.0.3/32     *[OSPF/10] 23:52:30, metric 1
                    > to 10.0.0.5 via fe-1/2/2.0
192.168.0.5/32     *[BGP/170] 23:53:11, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.5 via fe-1/2/3.0
                    [BGP/170] 23:53:09, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.1 via fe-1/2/0.0
172.16.233.5/32       *[OSPF/10] 23:53:25, metric 1
                      MultiRecv
```

Verifying the Routes on Device ISP-3

Purpose On Device ISP-3, check the routes in the routing table.

Action

```
user@ISP-3> show route

inet.0: 40 destinations, 41 routes (40 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Aggregate/130] 23:53:57, metric2 1
                    > to 10.0.0.2 via fe-1/2/0.0
                    [BGP/170] 22:46:17, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
10.0.0.0/30        *[Direct/0] 23:53:52
                    > via fe-1/2/0.0
10.0.0.1/32        *[Local/0] 23:53:53
                      Local via fe-1/2/0.0
10.0.0.4/30        *[Direct/0] 23:53:54
                    > via fe-1/2/2.0
10.0.0.5/32        *[Local/0] 23:53:54
                      Local via fe-1/2/2.0
10.0.0.8/30        *[Direct/0] 23:53:53
                    > via fe-1/2/1.0
10.0.0.9/32        *[Local/0] 23:53:53
                      Local via fe-1/2/1.0
10.0.0.20/30       *[BGP/170] 23:53:02, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
10.1.0.0/30        *[OSPF/10] 23:53:03, metric 2
                    > to 10.0.0.6 via fe-1/2/2.0
                      to 10.0.0.2 via fe-1/2/0.0
10.1.0.4/30        *[Direct/0] 23:53:54
                    > via fe-1/2/3.0
10.1.0.5/32        *[Local/0] 23:53:54
                      Local via fe-1/2/3.0
10.3.0.4/30        *[BGP/170] 23:52:46, localpref 100, from 192.168.0.1
                      AS path: 64514 64515 64516 I, validation-state: unverified

                    > to 10.0.0.2 via fe-1/2/0.0
10.3.0.48/30       *[BGP/170] 23:53:02, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.8.0/21      *[BGP/170] 00:12:59, localpref 100, from 192.168.0.1
                      AS path: 64514 I, validation-state: unverified
                    > to 10.0.0.2 via fe-1/2/0.0
172.16.16.0/21     *[BGP/170] 02:04:01, localpref 100, from 192.168.0.2
                      AS path: 64515 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.24.0/25     *[BGP/170] 23:08:24, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.24.128/25   *[BGP/170] 23:08:24, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.25.0/26     *[BGP/170] 23:08:24, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.25.64/26    *[BGP/170] 23:08:24, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.32.0/24     *[BGP/170] 22:46:17, localpref 100, from 192.168.0.1
                      AS path: I, validation-state: unverified
                    > to 10.0.0.2 via fe-1/2/0.0
```

```
172.16.33.0/24      *[BGP/170] 22:46:17, localpref 100, from 192.168.0.1
                      AS path: I, validation-state: unverified
                    > to 10.0.0.2 via fe-1/2/0.0
172.16.34.0/24      *[BGP/170] 22:41:11, localpref 100, from 192.168.0.2
                      AS path: I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.35.0/24      *[BGP/170] 22:41:11, localpref 100, from 192.168.0.2
                      AS path: I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.36.0/24      *[Static/5] 22:41:11
                      Reject
172.16.37.0/24      *[Static/5] 22:41:11
                      Reject
172.16.38.0/24      *[Static/5] 22:41:11
                      Reject
172.16.39.0/24      *[Static/5] 22:41:11
                      Reject
172.16.40.0/25      *[BGP/170] 23:02:38, localpref 100
                      AS path: 64511 I, validation-state: unverified
                    > to 10.1.0.6 via fe-1/2/3.0
172.16.40.128/25    *[BGP/170] 23:02:38, localpref 100
                      AS path: 64511 I, validation-state: unverified
                    > to 10.1.0.6 via fe-1/2/3.0
172.16.41.0/25      *[BGP/170] 23:02:38, localpref 100
                      AS path: 64511 I, validation-state: unverified
                    > to 10.1.0.6 via fe-1/2/3.0
172.16.41.128/25    *[BGP/170] 23:02:38, localpref 100
                      AS path: 64511 I, validation-state: unverified
                    > to 10.1.0.6 via fe-1/2/3.0
172.16.44.0/26      *[BGP/170] 22:59:52, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.10 via fe-1/2/1.0
172.16.44.64/26     *[BGP/170] 22:59:52, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.10 via fe-1/2/1.0
172.16.44.128/26    *[BGP/170] 22:59:52, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.10 via fe-1/2/1.0
172.16.44.192/26    *[BGP/170] 22:59:52, localpref 100
                      AS path: 64512 I, validation-state: unverified
                    > to 10.0.0.10 via fe-1/2/1.0
192.168.0.1/32      *[OSPF/10] 23:53:03, metric 1
                    > to 10.0.0.2 via fe-1/2/0.0
192.168.0.2/32      *[OSPF/10] 23:53:03, metric 1
                    > to 10.0.0.6 via fe-1/2/2.0
192.168.0.3/32      *[Direct/0] 23:53:54
                    > via lo0.0
192.168.0.5/32      *[BGP/170] 23:53:02, localpref 100, from 192.168.0.2
                      AS path: 64516 I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
172.16.233.5/32       *[OSPF/10] 23:53:58, metric 1
                      MultiRecv
```

Verifying the Routes on Device Exchange-1

Purpose On Device Exchange-1, check the routes in the routing table.

Action user@Exchange-1> **show route**

```
inet.0: 23 destinations, 24 routes (23 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.20/30       *[BGP/170] 23:53:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
10.2.0.4/30        *[Direct/0] 23:54:23
                    > via fe-1/2/3.0
10.2.0.5/32        *[Local/0] 23:54:29
                      Local via fe-1/2/3.0
10.3.0.4/30        *[BGP/170] 23:53:51, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
10.3.0.40/30       *[Direct/0] 23:54:27
                    > via fe-1/2/2.0
10.3.0.42/32       *[Local/0] 23:54:29
                      Local via fe-1/2/2.0
10.3.0.44/30       *[Direct/0] 23:54:29
                    > via fe-1/2/1.0
10.3.0.45/32       *[Local/0] 23:54:29
                      Local via fe-1/2/1.0
172.16.8.0/21      *[Static/5] 00:13:31
                      Reject
172.16.16.0/21     *[BGP/170] 02:04:33, localpref 100
                      AS path: 64515 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.24.0/25     *[BGP/170] 23:08:56, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.24.128/25   *[BGP/170] 23:08:56, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.25.0/26     *[BGP/170] 23:08:56, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.25.64/26    *[BGP/170] 23:08:56, localpref 100
                      AS path: 64515 64516 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.32.0/21     *[BGP/170] 22:46:49, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.6 via fe-1/2/3.0
                    [BGP/170] 22:41:43, localpref 100
                      AS path: 64515 64510 I, validation-state: unverified
                    > to 10.3.0.41 via fe-1/2/2.0
172.16.40.0/22     *[BGP/170] 22:46:49, localpref 100
                      AS path: 64510 64511 I, validation-state: unverified
                    > to 10.2.0.6 via fe-1/2/3.0
172.16.44.0/23     *[BGP/170] 22:41:43, localpref 100
                      AS path: 64515 64510 64512 I, validation-state: unverified

                    > to 10.3.0.41 via fe-1/2/2.0
172.16.44.0/26     *[BGP/170] 23:00:24, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified

                    > to 10.3.0.41 via fe-1/2/2.0
172.16.44.64/26    *[BGP/170] 23:00:24, localpref 100
                      AS path: 64515 64516 64512 I, validation-state: unverified
```

```
                        > to 10.3.0.41 via fe-1/2/2.0
172.16.44.128/26       *[BGP/170] 23:00:24, localpref 100
                          AS path: 64515 64516 64512 I, validation-state: unverified

                        > to 10.3.0.41 via fe-1/2/2.0
172.16.44.192/26       *[BGP/170] 23:00:24, localpref 100
                          AS path: 64515 64516 64512 I, validation-state: unverified

                        > to 10.3.0.41 via fe-1/2/2.0
192.168.0.5/32         *[BGP/170] 23:53:51, localpref 100
                          AS path: 64515 64516 I, validation-state: unverified
                        > to 10.3.0.41 via fe-1/2/2.0
192.168.0.6/32         *[Direct/0] 23:54:29
                        > via lo0.0
```

Verifying the Routes on Device Exchange-2

Purpose On Device Exchange-2, check the routes in the routing table.

Action

```
user@Exchange-2> show route
inet.0: 24 destinations, 26 routes (23 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.20/30       *[BGP/170] 23:54:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
10.3.0.0/30        *[Direct/0] 23:54:57
                    > via fe-1/2/0.0
10.3.0.1/32        *[Local/0] 23:54:57
                      Local via fe-1/2/0.0
10.3.0.4/30        *[BGP/170] 23:54:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
10.3.0.40/30       *[Direct/0] 23:54:57
                    > via fe-1/2/2.0
10.3.0.41/32       *[Local/0] 23:54:57
                      Local via fe-1/2/2.0
10.3.0.48/30       *[Direct/0] 23:54:57
                    > via fe-1/2/1.0
                    [BGP/170] 23:54:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
10.3.0.49/32       *[Local/0] 23:54:57
                      Local via fe-1/2/1.0
172.16.8.0/21      *[BGP/170] 00:14:01, localpref 100
                      AS path: 64514 I, validation-state: unverified
                    > to 10.3.0.42 via fe-1/2/2.0
172.16.16.0/21     *[Static/5] 02:05:03
                      Reject
172.16.24.0/25     *[BGP/170] 23:09:26, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.24.128/25   *[BGP/170] 23:09:26, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.25.0/26     *[BGP/170] 23:09:26, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.25.64/26    *[BGP/170] 23:09:26, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.32.0/21     *[BGP/170] 22:42:13, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.2 via fe-1/2/0.0
                    [BGP/170] 22:47:19, localpref 100
                      AS path: 64514 64510 I, validation-state: unverified
                    > to 10.3.0.42 via fe-1/2/2.0
172.16.40.0/22     *[BGP/170] 22:47:19, localpref 100
                      AS path: 64514 64510 64511 I, validation-state: unverified

                    > to 10.3.0.42 via fe-1/2/2.0
172.16.44.0/23     *[BGP/170] 22:42:13, localpref 100
                      AS path: 64510 64512 I, validation-state: unverified
                    > to 10.3.0.2 via fe-1/2/0.0
172.16.44.0/26     *[BGP/170] 23:00:54, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.44.64/26    *[BGP/170] 23:00:54, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
```

```
                          > to 10.3.0.50 via fe-1/2/1.0
172.16.44.128/26   *[BGP/170] 23:00:54, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
172.16.44.192/26   *[BGP/170] 23:00:54, localpref 100
                      AS path: 64516 64512 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
192.168.0.5/32     *[BGP/170] 23:54:44, localpref 100
                      AS path: 64516 I, validation-state: unverified
                    > to 10.3.0.50 via fe-1/2/1.0
192.168.0.7/32     *[Direct/0] 23:54:57
                    > via lo0.0
```

Meaning On Device Exchange-2, the default route 0/0 is hidden because the next hop for the route is its own interface to Device Private-Peer-2, from which the route was received. The route is hidden to avoid a loop.

Verifying the Routes on Device Private-Peer-1

Purpose On Device Private-Peer-1, check the routes in the routing table.

Action

```
user@Private-Peer-1> show route

inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.2.0.0/30        *[Direct/0] 23:58:57
                    > via fe-1/2/2.0
10.2.0.1/32        *[Local/0] 5d 21:34:22
                      Local via fe-1/2/2.0
10.3.0.44/30       *[Direct/0] 23:59:02
                    > via fe-1/2/1.0
10.3.0.46/32       *[Local/0] 1d 03:19:52
                      Local via fe-1/2/1.0
172.16.32.0/24     *[BGP/170] 22:51:22, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.33.0/24     *[BGP/170] 22:51:22, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.34.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.35.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.36.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.37.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.38.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
172.16.39.0/24     *[BGP/170] 22:46:16, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.2.0.2 via fe-1/2/2.0
192.168.0.4/32     *[Direct/0] 5d 21:34:22
                    > via lo0.0
```

Verifying the Routes on Device Private-Peer-2

Purpose On Device Private-Peer-2, check the routes in the routing table.

Action `user@Private-Peer-2>` **show route**

```
inet.0: 29 destinations, 29 routes (29 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Aggregate/130] 1d 02:13:28
                    > to 10.3.0.49 via fe-1/2/1.0
10.0.0.20/30       *[Direct/0] 1d 00:00:53
                    > via fe-1/2/0.0
10.0.0.22/32       *[Local/0] 4d 23:51:14
                      Local via fe-1/2/0.0
10.3.0.4/30        *[Direct/0] 23:59:36
                    > via fe-1/2/3.0
10.3.0.5/32        *[Local/0] 5d 21:34:57
                      Local via fe-1/2/3.0
10.3.0.48/30       *[Direct/0] 23:59:35
                    > via fe-1/2/1.0
10.3.0.50/32       *[Local/0] 1d 03:20:27
                      Local via fe-1/2/1.0
172.16.8.0/21      *[BGP/170] 00:18:39, localpref 100
                      AS path: 64515 64514 I, validation-state: unverified
                    > to 10.3.0.49 via fe-1/2/1.0
172.16.16.0/21     *[BGP/170] 02:09:41, localpref 100
                      AS path: 64515 I, validation-state: unverified
                    > to 10.3.0.49 via fe-1/2/1.0
172.16.24.0/25     *[Static/5] 23:14:04
                      Reject
172.16.24.128/25   *[Static/5] 23:14:04
                      Reject
172.16.25.0/26     *[Static/5] 23:14:04
                      Reject
172.16.25.64/26    *[Static/5] 23:14:04
                      Reject
172.16.32.0/21     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64515 64510 I, validation-state: unverified
                    > to 10.3.0.49 via fe-1/2/1.0
172.16.32.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.33.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.34.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.35.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.36.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.37.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.38.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
                    > to 10.3.0.6 via fe-1/2/3.0
172.16.39.0/24     *[BGP/170] 22:46:51, localpref 100
                      AS path: 64510 I, validation-state: unverified
```



```
                          > to 10.3.0.6 via fe-1/2/3.0
172.16.40.0/22           *[BGP/170] 22:51:57, localpref 100
                            AS path: 64515 64514 64510 64511 I, validation-state:
unverified
                          > to 10.3.0.49 via fe-1/2/1.0
172.16.44.0/23           *[BGP/170] 22:46:51, localpref 100
                            AS path: 64515 64510 64512 I, validation-state: unverified

                          > to 10.3.0.49 via fe-1/2/1.0
172.16.44.0/26           *[BGP/170] 23:05:32, localpref 100
                            AS path: 64512 I, validation-state: unverified
                          > to 10.0.0.21 via fe-1/2/0.0
172.16.44.64/26          *[BGP/170] 23:05:32, localpref 100
                            AS path: 64512 I, validation-state: unverified
                          > to 10.0.0.21 via fe-1/2/0.0
172.16.44.128/26         *[BGP/170] 23:05:32, localpref 100
                            AS path: 64512 I, validation-state: unverified
                          > to 10.0.0.21 via fe-1/2/0.0
172.16.44.192/26         *[BGP/170] 23:05:32, localpref 100
                            AS path: 64512 I, validation-state: unverified
                          > to 10.0.0.21 via fe-1/2/0.0
192.168.0.5/32           *[Direct/0] 5d 21:34:57
                          > via lo0.0
```

Related Documentation

- Example: Configuring Policy Chains and Route Filters on page 186
- Example: Configuring Routing Policy Prefix Lists on page 282

Understanding Policy Expressions

Policy expressions give the policy framework software a different way to evaluate routing policies. A *policy expression* uses Boolean logical operators with policies. The logical operators establish rules by which the policies are evaluated.

During evaluation of a routing policy in a policy expression, the policy action of accept, reject, or next policy is converted to the value of TRUE or FALSE. This value is then evaluated against the specified logical operator to produce output of either TRUE or FALSE. The output is then converted back to a flow control action of accept, reject, or next policy. The result of the policy expression is applied as it would be applied to a single policy; the route is accepted or rejected and the evaluation ends, or the next policy is evaluated.

Table 14 on page 148 summarizes the policy actions and their corresponding TRUE and FALSE values and flow control action values. Table 15 on page 148 describes the logical operators. For complete information about policy expression evaluation, see "Policy Expression Evaluation" on page 150.

You must enclose a policy expression in parentheses. You can place a policy expression anywhere in the **import** or **export** statements and in the **from policy** statement.

Table 14: Policy Action Conversion Values

Policy Action	Conversion Value	Flow Control Action Conversion Value
Accept	TRUE	Accept
Reject	FALSE	Reject
Next policy	TRUE	Next policy

Table 15: Policy Expression Logical Operators

Logical Operator	Policy Expression Logic	How Logical Operator Affects Policy Expression Evaluation
&& (Logical AND)	Logical AND requires that all values must be TRUE to produce output of TRUE. Routing policy value of TRUE and TRUE produces output of TRUE. Value of TRUE and FALSE produces output of FALSE. Value of FALSE and FALSE produces output of FALSE.	If the first routing policy returns the value of TRUE, the next policy is evaluated. If the first policy returns the value of FALSE, the evaluation of the expression ends and subsequent policies in the expression are not evaluated.
|| (Logical OR)	Logical OR requires that at least one value must be TRUE to produce output of TRUE. Routing policy value of TRUE and FALSE produces output of TRUE. Value of TRUE and TRUE produces output of TRUE. Value of FALSE and FALSE produces output of FALSE.	If the first routing policy returns the value of TRUE, the evaluation of the expression ends and subsequent policies in the expression are not evaluated. If the first policy returns the value of FALSE, the next policy is evaluated.
! (Logical NOT)	Logical NOT reverses value of TRUE to FALSE and of FALSE to TRUE. It also reverses the actions of accept and next policy to reject, and reject to accept.	If used with the logical AND operator and the first routing policy value of FALSE is reversed to TRUE, the next policy is evaluated. If the value of TRUE is reversed to FALSE, the evaluation of the expression ends and subsequent policies in the expression are not evaluated. If used with the logical OR operator and the first routing policy value of FALSE is reversed to TRUE, the evaluation of the expression ends and subsequent policies in the expression are not evaluated. If the value of TRUE is reversed to FALSE, the next policy is evaluated. If used with a policy and the flow control action is accept or next policy, these actions are reversed to reject. If the flow control action is reject, this action is reversed to accept.

For more information, see the following sections:

- Policy Expression Examples on page 149
- Policy Expression Evaluation on page 150
- Evaluating Policy Expressions on page 151

Policy Expression Examples

The following examples show how to use the logical operators to create policy expressions:

- Logical AND—In the following example, **policy1** is evaluated first. If after **policy1** is evaluated, a value of TRUE is returned, **policy2** is evaluated. If a value of FALSE is returned, **policy2** is not evaluated.

  export (policy1 && policy2)

- Logical OR—In the following example, **policy1** is evaluated first. If after **policy1** is evaluated, a value of TRUE is returned, **policy2** is not evaluated. If a value of FALSE is returned, **policy2** is evaluated.

  export (policy1 || policy2)

- Logical OR and logical AND—In the following example, **policy1** is evaluated first. If after **policy1** is evaluated, a value of TRUE is returned, **policy2** is skipped and **policy3** is evaluated. If after **policy1** is evaluated, a value of FALSE is returned, **policy2** is evaluated. If **policy2** returns a value of TRUE, **policy3** is evaluated. If **policy2** returns a value of FALSE, **policy3** is not evaluated.

  export [(policy1 || policy2) && policy3]

- Logical NOT—In the following example, **policy1** is evaluated first. If after **policy1** is evaluated, a value of TRUE is returned, the value is reversed to FALSE and **policy2** is not evaluated. If a value of FALSE is returned, the value is reversed to TRUE and **policy2** is evaluated.

  export (!policy1 && policy2)

The sequential list **[policy1 policy2 policy3]** is not the same as the policy expression **(policy1 && policy2 && policy3)**.

The sequential list is evaluated on the basis of a route matching a routing policy. For example, if **policy1** matches and the action is **accept** or **reject**, **policy2** and **policy3** are not evaluated. If **policy1** does not match, **policy2** is evaluated and so on until a match occurs and the action is **accept** or **reject**.

The policy expressions are evaluated on the basis of the action in a routing policy that is converted to the value of TRUE or FALSE and the logic of the specified logical operator. (For complete information about policy expression evaluation, see "Policy Expression Evaluation" on page 150.) For example, if **policy1** returns a value of FALSE, **policy2** and **policy3** are not evaluated. If **policy1** returns a value of TRUE, **policy2** is evaluated. If **policy2** returns a value of FALSE, **policy3** is not evaluated. If **policy2** returns a value of TRUE, **policy3** is evaluated.

You can also combine policy expressions and sequential lists. In the following example, if **policy1** returns a value of FALSE, **policy2** is evaluated. If **policy2** returns a value of TRUE and contains a **next policy** action, **policy3** is evaluated. If **policy2** returns a value of TRUE but does not contain an action, including a **next policy** action, **policy3** is still evaluated (because if you do not specify an action, next term or next policy are the default actions). If **policy2** returns a value of TRUE and contains an **accept** action, **policy3** is not evaluated.

export [(policy1 || policy2) policy3]

Policy Expression Evaluation

During evaluation, the policy framework software converts policy actions to values of TRUE or FALSE, which are factors in determining the flow control action that is performed upon a route. However, the software does not actually perform a flow control action on a route until it evaluates an entire policy expression.

The policy framework software evaluates a policy expression as follows:

1. The software evaluates a route against the first routing policy in a policy expression and converts the specified or default action to a value of TRUE or FALSE. (For information about the policy action conversion values, see Table 14 on page 148.)

2. The software takes the value of TRUE or FALSE and evaluates it against the logical operator used in the policy expression (see Table 15 on page 148). Based upon the logical operator used, the software determines whether or not to evaluate the next policy, if one is present.

   The policy framework software uses a shortcut method of evaluation: if the result of evaluating a policy predetermines the value of the entire policy expression, the software does not evaluate the subsequent policies in the expression. For example, if the policy expression uses the logical AND operator and the evaluation of a policy returns the value of FALSE, the software does not evaluate subsequent policies in the expression because the final value of the expression is guaranteed to be FALSE no matter what the values of the unevaluated policies.

3. The software performs Step 1 and Step 2 for each subsequent routing policy in the policy expression, if they are present and it is necessary to evaluate them.

4. After evaluating the last routing policy, if it is appropriate, the software evaluates the value of TRUE or FALSE obtained from each routing policy evaluation. Based upon the logical operator used, it calculates an output of TRUE or FALSE.

5. The software converts the output of TRUE or FALSE back to an action. (For information about the policy action conversion values, see Table 14 on page 148.) The action is performed.

   If each policy in the expression returned a value of TRUE, the software converts the output of TRUE back to the flow control action specified in the last policy. For example, if the policy expression **(policy1 && policy2)** is specified and **policy1** specifies **accept** and **policy2** specifies **next term**, the **next term** action is performed.

   If an action specified in one of the policies manipulates a route characteristic, the policy framework software carries the new route characteristic forward during the evaluation of the remaining policies. For example, if the action specified in the first policy of a policy expression sets a route's metric to 500, this route matches the criteria of **metric 500** defined in the next policy. However, if a route characteristic manipulation action is specified in a policy located in the middle or the end of a policy expression, it is possible, because of the shortcut evaluation, that the policy is never evaluated and the manipulation of the route characteristic never occurs.

Evaluating Policy Expressions

The following sample routing policy uses three policy expressions:

```
[edit]
policy-options {
  policy-statement policy-A {
    from {
      route-filter 10.10.0.0/16 orlonger;
    }
    then reject;
  }
}
policy-options {
  policy-statement policy-B {
    from {
      route-filter 10.20.0.0/16 orlonger;
    }
    then accept;
  }
}
protocols {
  bgp {
    neighbor 192.168.1.1 {
      export (policy-A && policy-B);
    }
    neighbor 192.168.2.1 {
      export (policy-A || policy-B);
    }
    neighbor 192.168.3.1 {
      export (!policy-A);
    }
  }
}
```

The policy framework software evaluates the transit BGP route 10.10.1.0/24 against the three policy expressions specified in the sample routing policy as follows:

- (policy-A && policy-B)—10.10.1.0/24 is evaluated against **policy-A**. 10.10.1.0/24 matches the route list specified in **policy-A**, so the specified action of **reject** is returned. **reject** is converted to a value of FALSE, and FALSE is evaluated against the specified logical AND. Because the result of FALSE is certain no matter what the results of the evaluation of **policy-B** are (in policy expression logic, any result AND a value of FALSE produces the output of FALSE), **policy-B** is not evaluated and the output of FALSE is produced. The FALSE output is converted to **reject**, and 10.10.1.0/24 is rejected.
- (policy-A || policy-B)—10.10.1.0/24 is evaluated against **policy-A**. 10.10.1.0/24 matches the route list specified in **policy-A**, so the specified action of **reject** is returned. **reject** is converted to a value of FALSE, then FALSE is evaluated against the specified logical OR. Because logical OR requires at least one value of TRUE to produce an output of TRUE, 10.10.1.0/24 is evaluated against **policy-B**. 10.10.1.0/24 does not match **policy-B**,

so the default action of **next-policy** is returned. The **next-policy** is converted to a value of TRUE, then the value of FALSE (for **policy-A** evaluation) and TRUE (for **policy-B** evaluation) are evaluated against the specified logical OR. In policy expression logic, FALSE OR TRUE produce an output of TRUE. The output of TRUE is converted to **next-policy**. (TRUE is converted to **next-policy** because **next-policy** was the last action retained by the policy framework software.) **policy-B** is the last routing policy in the policy expression, so the action specified by the default export policy for BGP is taken.

- (!policy-A)—10.10.1.0/24 is evaluated against **policy-A**. 10.10.1.0/24 matches the route list specified in **policy-A**, so the specified action of **reject** is returned. **reject** is converted to a value of FALSE, and FALSE is evaluated against the specified logical NOT. The value of FALSE is reversed to an output of TRUE based on the rules of logical NOT. The output of TRUE is converted to **accept**, and route 10.10.1.0/24 is accepted.

Related Documentation

- Example: Testing a Routing Policy with Complex Regular Expressions on page 544
- Example: Configuring a Policy Subroutine on page 203
- Example: Configuring Policy Chains and Route Filters on page 186
- Example: Configuring Routing Policy Prefix Lists on page 282

Understanding Backup Selection Policy for OSPF Protocol

Support for OSPF loop-free alternate (LFA) routes essentially adds IP fast-reroute capability for OSPF. Junos OS precomputes multiple loop-free backup routes for all OSPF routes. These backup routes are pre-installed in the Packet Forwarding Engine, which performs a local repair and implements the backup path when the link for a primary next hop for a particular route is no longer available. The selection of LFA is done randomly by selecting any matching LFA to progress to the given destination. This does not ensure best backup coverage available for the network. In order to choose the best LFA, Junos OS allows you to configure network-wide backup selection policies for each destination (IPv4 and IPv6) and a primary next-hop interface. These policies are evaluated based on admin-group, srlg, bandwidth, protection-type, metric, and node information.

During backup shortest-path-first (SPF) computation, each node and link attribute of the backup path is accumulated by IGP and is associated with every node (router) in the topology. The next hop in the best backup path is selected as the backup next hop in the routing table. In general, backup evaluation policy rules are categorized into the following types:

- Pruning — Rules configured to select the eligible backup path.
- Ordering — Rules configured to select the best among the eligible backup paths.

The backup selection policies can be configured with both pruning and ordering rules. While evaluating the backup policies, each backup path is assigned a score, an integer value that signifies the total weight of the evaluated criteria. The backup path with the highest score is selected.

To enforce LFA selection, configure various rules for the following attributes:

- admin-group– Administrative groups, also known as link coloring or resource class, are manually assigned attributes that describe the "color" of links, such that links with the same color conceptually belong to the same class. These configured administrative groups are defined under protocol MPLS. You can use administrative groups to implement a variety of backup selection policies using exclude, include-all, include-any, or preference.
- srlg— A shared risk link group (SRLG) is a set of links sharing a common resource, which affects all links in the set if the common resource fails. These links share the same risk of failure and are therefore considered to belong to the same SRLG. For example, links sharing a common fiber are said to be in the same SRLG because a fault with the fiber might cause all links in the group to fail. An SRLG is represented by a 32-bit number unique within an IGP (OSPF) domain. A link might belong to multiple SRLGs. You can define the backup selection to either allow or reject the common SRLGs between the primary and the backup path. This rejection of common SRLGs are based on the non-existence of link having common SRLGs in the primary next-hop and the backup SPF.

  NOTE: **Administrative groups and SRLGs can be created only for default topologies.**

- bandwidth—The bandwidth specifies the bandwidth constraints between the primary and the backup path. The backup next-hop link can be used only if the bandwidth of the backup next-hop interface is greater than or equal to the bandwidth of the primary next hop.
- protection-type— The protection-type protects the destination from node failure of the primary node or link failure of the primary link. You can configure node, link, or node-link to protect the destination. If link-node is configured , then the node-protecting LFA is preferred over link-protection LFA.
- node- The node is per-node policy information. Here, node can be a directly connected router, remote router like RSVP backup LSP tail-end, or any other router in the backup SPF path. The nodes are identified through the route-id advertised by a node in the LSP. You can list the nodes to either prefer or exclude them in the backup path.
- metric— Metric decides how the LFAs should be preferred. In backup selection path, root metric and dest-metric are the two types of metrics. root-metric indicates the metric to the one-hop neighbor or a remote router such as an RSVP backup LSP tail-end router. The dest-metric indicates the metric from a one-hop neighbor or remote router such as an RSVP backup LSP tail-end router to the final destination. The metric evaluation is done either in ascending or descending order. By default, the first preference is given to backup paths with lowest destination evaluation and then to backup paths with lowest root metrics.

The evaluation-order allows you to control the order and criteria of evaluating these attributes in the backup path. You can explicitly configure the evaluation order. Only the configured attributes influence the backup path selection. The default order of evaluation of these attributes for the LFA is [admin-group srlg bandwidth protection-type node metric] .

NOTE: TE attributes are not supported in OSPFv3 and cannot be used for backup selection policy evaluation for IPv6 prefixes.

Related Documentation

- Example: Configuring Backup Selection Policy for the OSPF or OSPF3 Protocol on page 159
- Configuring Backup Selection Policy for the OSPF Protocol on page 154
- *backup-selection (Protocols ISIS)*

Configuring Backup Selection Policy for the OSPF Protocol

Support for OSPF loop-free alternate (LFA) routes essentially adds IP fast-reroute capability for OSPF. Junos OS precomputes multiple loop-free backup routes for all OSPF routes. These backup routes are pre-installed in the Packet Forwarding Engine, which performs a local repair and implements the backup path when the link for a primary next hop for a particular route is no longer available. The selection of LFA is done randomly by selecting any matching LFA to progress to the given destination. This does not ensure best backup coverage available for the network. In order to choose the best LFA, Junos OS allows you to configure network-wide backup selection policies for each destination (IPv4 and IPv6) and a primary next-hop interface. These policies are evaluated based on admin-group, srlg, bandwidth, protection-type, metric, and node information.

Before you begin to configure the backup selection policy for the OSPF protocol:

- Configure the router interfaces. See the *Junos OS Network Management Administration Guide for Routing Devices*.
- Configure an interior gateway protocol or static routing. See the *Junos OS Routing Protocols Library for Routing Devices*.

To configure the backup selection policy for the OSPF protocol:

1. Configure per-packet load balancing.

   ```
   [edit policy-options]
   user@host# set policy-statement ecmp term 1 then load-balance per-packet
   ```

2. Enable RSVP on all the interfaces.

   ```
   [edit protocols]
   user@host# set rsvp interface all
   ```

3. Configure administrative groups.

   ```
   [edit protocols mpls]
   user@host# set admin-groups group-name
   ```

4. Configure srlg values.

[edit routing-options]
user@host# **set srlg *srlg-name* srlg-value *srlg-value***

5. Enable MPLS on all the interfaces.

   [edit protocols mpls]
   user@host# **set interface all**

6. Apply MPLS to an interface configured with an administrative group.

   [edit protocols mpls]
   user@host# **set interface *interface-name* admin-group *group-name***

7. Configure the ID of the router.

   [edit routing-options]
   user@host# **set router-id *router-id***

8. Apply the routing policy to all equal cost multipaths exported from the routing table to the forwarding table.

   [edit routing-options]
   user@host# **set forwarding-table export ecmp**

9. Enable link protection and configure metric values on all the interfaces for an area.

   [edit protocols ospf]
   user@host# **set area *area-id* interface *interface-name* link-protection**
   user@host# **set area *area-id* interface *interface-name* metric *metric***

10. Configure the administrative group of the backup selection policy for an IP address.

    You can choose to exclude, include all, include any, or prefer the administrative groups from the backup path.

    [edit routing-options]
    user@host# **set backup-selection destination *ip-address* interface *interface-name* admin-group**

    - Specify the administrative group to be excluded.

      [edit routing-options backup-selection destination *ip-address* interface *interface-name admin-group]*
      user@host# **set exclude *group-name***

      The backup path is not selected as the loop-free alternate (LFA) or backup nexthop if any of the links in the path have any one of the listed administrative groups.

      For example, to exclude the group c1 from the administrative group:

      [edit routing-options backup-selection destination 0.0.0.0/0 interface all admin-group]
      user@host# **set exclude c1**

    - Configure all the administrative groups if each link in the backup path requires all the listed administrative groups in order to accept the path.

[edit routing-options backup-selection destination *ip-address* interface *interface-name admin-group]*
user@host# **set include-all *group-name***

For example, to set all the administrative groups if each link requires all the listed administrative groups in order to accept the path:

[edit routing-options backup-selection destination 0.0.0.0/0 interface all admin-group]
user@host# **set include-all c2**

- Configure any administrative group if each link in the backup path requires at least one of the listed administrative groups in order to select the path.

  [edit routing-options backup-selection destination *ip-address* interface *interface-name admin-group]*
  user@host# **set include-any *group-name***

  For example, to set any administrative group if each link in the backup path requires at least one of the listed administrative groups in order to select the path:

  [edit routing-options backup-selection destination 0.0.0.0/0 interface all admin-group]
  user@host# **set include-any c3**

- Define an ordered set of an administrative group that specifies the preference of the backup path.

  The leftmost element in the set is given the highest preference.

  [edit routing-options backup-selection destination *ip-address* interface *interface-name admin-group]*
  user@host# **set preference *group-name***

  For example, to set an ordered set of an administrative group that specifies the preference of the backup path:

  [edit routing-options backup-selection destination 0.0.0.0/0 interface all admin-group]
  user@host# **set preference c4**

11. Configure the backup path to allow the selection of the backup next hop only if the bandwidth is greater than or equal to the bandwidth of the primary next hop.

    [edit routing-options]
    user@host# **set backup-selection destination *ip-address* interface *interface-name* bandwidth-greater-equal-primary**

12. Configure the backup path to specify the metric from the one-hop neighbor or from the remote router such as an RSVP backup label-switched-path (LSP) tail-end router to the final destination.

    The destination metric can be either highest or lowest.

    - Configure the backup path that has the highest destination metric.

      [edit routing-options]
      user@host# **set backup-selection destination *ip-address* interface *interface-name* dest-metric highest**

- Configure the backup path that has the lowest destination metric.

  [edit routing-options]
  user@host# **set backup-selection destination** *ip-address* **interface** *interface-name* **dest-metric lowest**

13. Configure the backup path that is a downstream path to the destination.

    [edit routing-options]
    user@host# **set backup-selection destination** *ip-address* **interface** *interface-name* **downstream-paths-only**

14. Set the order of preference of the root and the destination metric during backup path selection.

    The preference order can be :

    - [root dest] — Backup path selection or preference is first based on the root-metric criteria. If the criteria of all the root-metric is the same, then the selection or preference is based on the dest-metric.
    - [dest root] — Backup path selection or preference is first based on the dest-metric criteria. If the criteria of all the dest-metric is the same, then the selection is based on the root-metric.

    [edit routing-options]
    user@host# **set backup-selection destination** *ip-address* **interface** *interface-name* **metric-order dest**
    user@host# **set backup-selection destination** *ip-address* **interface** *interface-name* **metric-order root**

15. Configure the backup path to define a list of loop-back IP addresses of the adjacent neighbors to either exclude or prefer in the backup path selection.

    The neighbor can be a local (adjacent router) neighbor, remote neighbor, or any other router in the backup path.

    [edit routing-options]
    user@host# **set backup-selection destination** *ip-address* **interface** *interface-name* **node**

    - Configure the list of neighbors to be excluded.

      [edit routing-options backup-selection destination *ip-address* interface *interface-name* node]
      user@host# **set exclude** *node-address*

      The backup path that has a router from the list is not selected as the loop-free alternative or backup next hop.

    - Configure an ordered set of neighbors to be preferred.

      [edit routing-options backup-selection destination *ip-address* interface *interface-name* node]
      user@host# **set preference** *node-address*

      The backup path having the leftmost neighbor is selected.

16. Configure the backup path to specify the required protection type of the backup path to be link, node, or node-link.

    - Select the backup path that provides link protection.

      [edit routing-options]
      user@host# **set backup-selection destination *ip-address* interface *interface-name* protection-type link**

    - Select the backup path that provides node protection.

      [edit routing-options]
      user@host# **set backup-selection destination *ip-address* interface *interface-name* protection-type node**

    - Select the backup path that allows either node or link protection LFA where node-protection LFA is preferred over link-protection LFA.

      [edit routing-options]
      user@host# **set backup-selection destination *ip-address* interface *interface-name* protection-type node-link**

17. Specify the metric to the one-hop neighbor or to the remote router such as an RSVP backup label-switched-path (LSP) tail-end router.

    - Select the path with highest root metric.

      [edit routing-options]
      user@host# **set backup-selection destination *ip-address* interface all root-metric highest**

    - Select the path with lowest root metric.

      [edit routing-options]
      user@host# **set backup-selection destination *ip-address* interface all root-metric lowest**

18. Configure the backup selection path to either allow or reject the common shared risk link groups (SRLGs) between the primary link and each link in the backup path.

    - Configure the backup path to allow common srlgs between the primary link and each link in the backup path.

      [edit routing-options]
      user@host# **set backup-selection destination *ip-address* interface all srlg loose**

      A backup path with a fewer number of srlg collisions is preferred.

    - Configure the backup path to reject the backup path that has common srlgs between the primary next-hop link and each link in the backup path.

      [edit routing-options]
      user@host# **set backup-selection destination *ip-address* interface all srlg strict**

19. Configure the backup path to control the order and the criteria of evaluating the backup path based on the administrative group, srlg, bandwidth, protection type, node, and metric.

    The default order of evaluation is admin-group, srlg, bandwidth, protection-type, node, and metric.

    [edit routing-options]

user@host# **set backup-selection destination *ip-address* interface all evaluation-order admin-group**
user@host# **set backup-selection destination *ip-address* interface all evaluation-order srlg**
user@host# **set backup-selection destination *ip-address* interface all evaluation-order bandwidth**

Related Documentation

- Example: Configuring Backup Selection Policy for the OSPF or OSPF3 Protocol on page 159
- Understanding Backup Selection Policy for OSPF Protocol on page 152

- *backup-selection (Protocols ISIS)*

Example: Configuring Backup Selection Policy for the OSPF or OSPF3 Protocol

This example shows how to configure the backup selection policy for the OSPF or OSPF3 protocol, which enables you to select a loop-free alternate (LFA) in the network.

When you enable backup selection policies, Junos OS allows selection of LFA based on the policy rules and attributes of the links and nodes in the network. These attributes are admin-group, srlg, bandwidth, protection-type, metric, and node.

- Requirements on page 159
- Overview on page 159
- Configuration on page 160
- Verification on page 179

Requirements

This example uses the following hardware and software components:

- Eight routers that can be a combination of M Series Multiservice Edge Routers, MX Series 3D Universal Edge Routers, PTX Series Packet Transport Routers, and T Series Core Routers
- Junos OS Release 15.1 or later running on all devices

Before you begin:

1. Configure the device interfaces.
2. Configure OSPF.

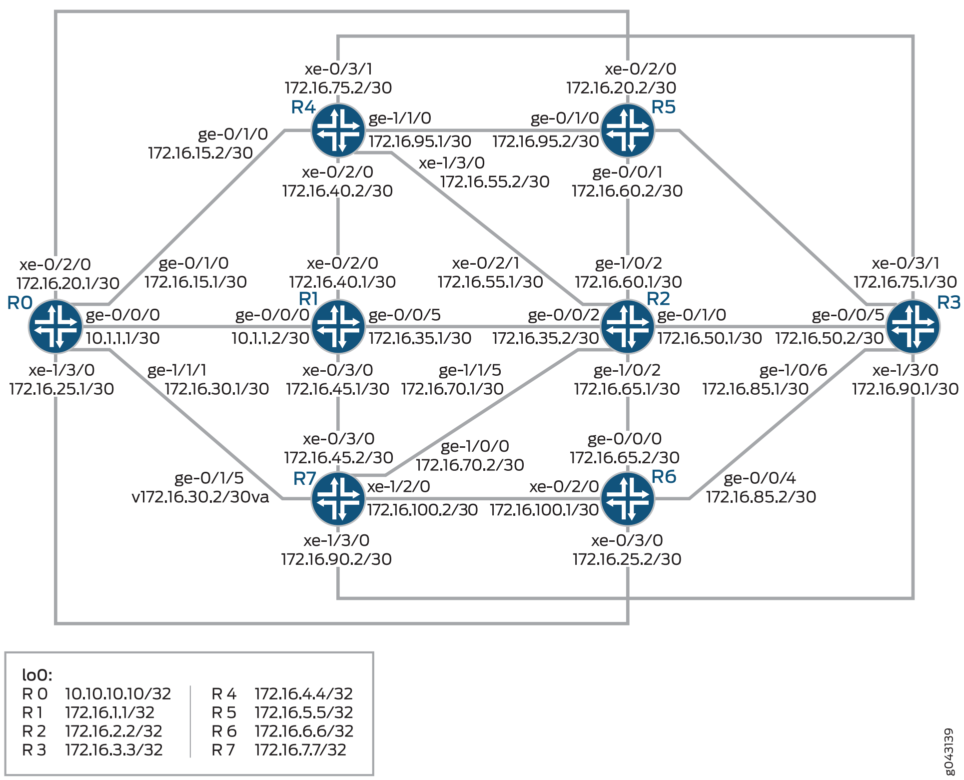
Overview

In Junos OS, the default loop-free alternative (LFA) selection algorithm or criteria can be overridden with an LFA policy. These policies are configured for each destination (IPv4 and IPv6) and a primary next-hop interface . These backup policies enforce LFA selection based on admin-group, srlg, bandwidth, protection-type, metric, and node attributes of the backup path. During backup shortest-path-first (SPF) computation, each attribute (both node and link) of the backup path, stored per backup next-hop, is accumulated by IGP. For the routes created internally by IGP, the attribute set of every backup path is evaluated against the policy configured for each destination (IPv4 and IPv6) and a primary next-hop interface. The first or the best backup path is selected and installed as the backup next hop in the routing table. To configure the backup selection policy, include the **backup-selection** configuration statement at the **[edit routing-options]** hierarchy level. The **show backup-selection** command displays the configured policies for a given interface and destination. The display can be filtered against a particular destination, prefix, interface, or logical systems.

Topology

In this topology shown in Figure 14 on page 160, the backup selection policy is configured on Device R3.

Figure 14: Example Backup Selection Policy for OSPF or OPSF3



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

R0

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.1.1/30
```

```
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10:1:1::1/64
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/1/0 unit 0 family inet address 172.16.15.1/30
set interfaces ge-0/1/0 unit 0 family inet6 address 2001:db8:15:1:1::1/64
set interfaces ge-0/1/0 unit 0 family mpls
set interfaces xe-0/2/0 unit 0 family inet address 172.16.20.1/30
set interfaces xe-0/2/0 unit 0 family inet6 address 2001:db8:20:1:1::1/64
set interfaces xe-0/2/0 unit 0 family mpls
set interfaces ge-1/0/5 unit 0 family inet address 172.16.150.1/24
set interfaces ge-1/0/5 unit 0 family inet6 address 2001:db8:150:1:1::1/64
set interfaces ge-1/0/5 unit 0 family mpls
set interfaces ge-1/1/1 unit 0 family inet address 172.16.30.1/30
set interfaces ge-1/1/1 unit 0 family inet6 address 2001:db8:30:1:1::1/64
set interfaces ge-1/1/1 unit 0 family mpls
set interfaces xe-1/3/0 unit 0 family inet address 172.16.25.1/30
set interfaces xe-1/3/0 unit 0 family inet6 address 2001:db8:25:1:1::1/64
set interfaces xe-1/3/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 10.10.10.10/32 primary
set interfaces lo0 unit 0 family inet6 address 2001:db8::10:10:10:10/128 primary
set interfaces lo0 unit 0 family mpls
set routing-options srlg srlg1 srlg-value 1001
set routing-options srlg srlg2 srlg-value 1002
set routing-options srlg srlg3 srlg-value 1003
set routing-options srlg srlg4 srlg-value 1004
set routing-options srlg srlg5 srlg-value 1005
set routing-options srlg srlg6 srlg-value 1006
set routing-options srlg srlg7 srlg-value 1007
set routing-options srlg srlg8 srlg-value 1008
set routing-options srlg srlg9 srlg-value 1009
set routing-options srlg srlg10 srlg-value 10010
set routing-options srlg srlg11 srlg-value 10011
set routing-options srlg srlg12 srlg-value 10012
set routing-options router-id 10.10.10.10
set protocols rsvp interface all
set protocols mpls admin-groups c0 0
set protocols mpls admin-groups c1 1
set protocols mpls admin-groups c2 2
set protocols mpls admin-groups c3 3
set protocols mpls admin-groups c4 4
set protocols mpls admin-groups c5 5
set protocols mpls admin-groups c6 6
set protocols mpls admin-groups c7 7
set protocols mpls admin-groups c8 8
set protocols mpls admin-groups c9 9
set protocols mpls admin-groups c10 10
set protocols mpls admin-groups c11 11
set protocols mpls admin-groups c12 12
set protocols mpls admin-groups c13 13
set protocols mpls admin-groups c14 14
set protocols mpls admin-groups c15 15
set protocols mpls admin-groups c16 16
set protocols mpls admin-groups c17 17
set protocols mpls admin-groups c18 18
set protocols mpls admin-groups c19 19
set protocols mpls admin-groups c20 20
set protocols mpls admin-groups c21 21
```



```
set protocols mpls admin-groups c22 22
set protocols mpls admin-groups c23 23
set protocols mpls admin-groups c24 24
set protocols mpls admin-groups c25 25
set protocols mpls admin-groups c26 26
set protocols mpls admin-groups c27 27
set protocols mpls admin-groups c28 28
set protocols mpls admin-groups c29 29
set protocols mpls admin-groups c30 30
set protocols mpls admin-groups c31 31
set protocols mpls interface all
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/1/0.0 metric 18
set protocols ospf area 0.0.0.0 interface xe-0/2/0.0 metric 51
set protocols ospf area 0.0.0.0 interface ge-1/1/1.0 metric 23
set protocols ospf area 0.0.0.0 interface xe-1/3/0.0 metric 52
set protocols ospf area 0.0.0.0 interface ge-1/0/5.0
set protocols ospf3 area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-0/1/0.0 metric 18
set protocols ospf3 area 0.0.0.0 interface xe-0/2/0.0 metric 51
set protocols ospf3 area 0.0.0.0 interface ge-1/1/1.0 metric 23
set protocols ospf3 area 0.0.0.0 interface xe-1/3/0.0 metric 52
set protocols ospf3 area 0.0.0.0 interface ge-1/0/5.0
```

R1

```
set interfaces ge-0/0/0 unit 0 family inet address 10.1.1.2/30
set interfaces ge-0/0/0 unit 0 family inet6 address 2001:db8:10:1:1::2/64
set interfaces ge-0/0/0 unit 0 family mpls
set interfaces ge-0/0/5 unit 0 family inet address 172.16.35.1/30
set interfaces ge-0/0/5 unit 0 family inet6 address 2001:db8:35:1:1::1/64
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces xe-0/2/0 unit 0 family inet address 172.16.40.1/30
set interfaces xe-0/2/0 unit 0 family inet6 address 2001:db8:40:1:1::1/64
set interfaces xe-0/2/0 unit 0 family mpls
set interfaces xe-0/3/0 unit 0 family inet address 172.16.45.1/30
set interfaces xe-0/3/0 unit 0 family inet6 address 2001:db8:45:1:1::1/64
set interfaces xe-0/3/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.1.1/32 primary
set interfaces lo0 unit 0 family inet6 address 2001:db8::1:1:1:1/128 primary
set interfaces lo0 unit 0 family mpls
set routing-options srlg srlg1 srlg-value 1001
set routing-options srlg srlg2 srlg-value 1002
set routing-options srlg srlg3 srlg-value 1003
set routing-options srlg srlg4 srlg-value 1004
set routing-options srlg srlg5 srlg-value 1005
set routing-options srlg srlg6 srlg-value 1006
set routing-options srlg srlg7 srlg-value 1007
set routing-options srlg srlg8 srlg-value 1008
set routing-options srlg srlg9 srlg-value 1009
set routing-options srlg srlg10 srlg-value 10010
set routing-options srlg srlg11 srlg-value 10011
set routing-options srlg srlg12 srlg-value 10012
set routing-options router-id 172.16.1.1
set protocols rsvp interface all
set protocols mpls admin-groups c0 0
set protocols mpls admin-groups c1 1
```



```
set protocols mpls admin-groups c2 2
set protocols mpls admin-groups c3 3
set protocols mpls admin-groups c4 4
set protocols mpls admin-groups c5 5
set protocols mpls admin-groups c6 6
set protocols mpls admin-groups c7 7
set protocols mpls admin-groups c8 8
set protocols mpls admin-groups c9 9
set protocols mpls admin-groups c10 10
set protocols mpls admin-groups c11 11
set protocols mpls admin-groups c12 12
set protocols mpls admin-groups c13 13
set protocols mpls admin-groups c14 14
set protocols mpls admin-groups c15 15
set protocols mpls admin-groups c16 16
set protocols mpls admin-groups c17 17
set protocols mpls admin-groups c18 18
set protocols mpls admin-groups c19 19
set protocols mpls admin-groups c20 20
set protocols mpls admin-groups c21 21
set protocols mpls admin-groups c22 22
set protocols mpls admin-groups c23 23
set protocols mpls admin-groups c24 24
set protocols mpls admin-groups c25 25
set protocols mpls admin-groups c26 26
set protocols mpls admin-groups c27 27
set protocols mpls admin-groups c28 28
set protocols mpls admin-groups c29 29
set protocols mpls admin-groups c30 30
set protocols mpls admin-groups c31 31
set protocols mpls interface all
set protocols mpls interface ge-0/0/0.0 srlg srlg9
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/2/0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/3/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-0/0/5.0 metric 10
set protocols ospf3 area 0.0.0.0 interface xe-0/2/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface xe-0/3/0.0 metric 10
```

R2

```
set interfaces ge-0/0/2 unit 0 family inet address 172.16.35.2/30
set interfaces ge-0/0/2 unit 0 family inet6 address 2001:db8:35:1:1::2/64
set interfaces ge-0/0/2 unit 0 family mpls
set interfaces ge-0/1/0 unit 0 family inet address 172.16.50.1/30
set interfaces ge-0/1/0 unit 0 family inet6 address 2001:db8:50:1:1::1/64
set interfaces ge-0/1/0 unit 0 family mpls
set interfaces xe-0/2/1 unit 0 family inet address 172.16.55.1/30
set interfaces xe-0/2/1 unit 0 family inet6 address 2001:db8:55:1:1::1/64
set interfaces xe-0/2/1 unit 0 family mpls
set interfaces ge-1/0/2 unit 0 family inet address 172.16.60.1/30
set interfaces ge-1/0/2 unit 0 family inet6 address 2001:db8:60:1:1::1/64
set interfaces ge-1/0/2 unit 0 family mpls
set interfaces ge-1/0/9 unit 0 family inet address 172.16.65.1/30
set interfaces ge-1/0/9 unit 0 family inet6 address 2001:db8:65:1:1::1/64
```

```
set interfaces ge-1/0/9 unit 0 family mpls
set interfaces ge-1/1/5 unit 0 family inet address 172.16.70.1/30
set interfaces ge-1/1/5 unit 0 family inet6 address 2001:db8:70:1:1::1/64
set interfaces ge-1/1/5 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.2.2/32 primary
set interfaces lo0 unit 0 family inet6 address 2001:db8::2:2:2:2/128 primary
set interfaces lo0 unit 0 family mpls
set routing-options srlg srlg1 srlg-value 1001
set routing-options srlg srlg2 srlg-value 1002
set routing-options srlg srlg3 srlg-value 1003
set routing-options srlg srlg4 srlg-value 1004
set routing-options srlg srlg5 srlg-value 1005
set routing-options srlg srlg6 srlg-value 1006
set routing-options srlg srlg7 srlg-value 1007
set routing-options srlg srlg8 srlg-value 1008
set routing-options srlg srlg9 srlg-value 1009
set routing-options srlg srlg10 srlg-value 10010
set routing-options srlg srlg11 srlg-value 10011
set routing-options srlg srlg12 srlg-value 10012
set routing-options router-id 172.16.2.2
set protocols rsvp interface all
set protocols mpls admin-groups c0 0
set protocols mpls admin-groups c1 1
set protocols mpls admin-groups c2 2
set protocols mpls admin-groups c3 3
set protocols mpls admin-groups c4 4
set protocols mpls admin-groups c5 5
set protocols mpls admin-groups c6 6
set protocols mpls admin-groups c7 7
set protocols mpls admin-groups c8 8
set protocols mpls admin-groups c9 9
set protocols mpls admin-groups c10 10
set protocols mpls admin-groups c11 11
set protocols mpls admin-groups c12 12
set protocols mpls admin-groups c13 13
set protocols mpls admin-groups c14 14
set protocols mpls admin-groups c15 15
set protocols mpls admin-groups c16 16
set protocols mpls admin-groups c17 17
set protocols mpls admin-groups c18 18
set protocols mpls admin-groups c19 19
set protocols mpls admin-groups c20 20
set protocols mpls admin-groups c21 21
set protocols mpls admin-groups c22 22
set protocols mpls admin-groups c23 23
set protocols mpls admin-groups c24 24
set protocols mpls admin-groups c25 25
set protocols mpls admin-groups c26 26
set protocols mpls admin-groups c27 27
set protocols mpls admin-groups c28 28
set protocols mpls admin-groups c29 29
set protocols mpls admin-groups c30 30
set protocols mpls admin-groups c31 31
set protocols mpls interface all
set protocols mpls interface ge-0/1/0.0 srlg srlg1
set protocols mpls interface ge-1/0/9.0 srlg srlg1
```



```
set protocols mpls interface ge-1/1/5.0 srlg srlg7
set protocols ospf area 0.0.0.0 interface ge-0/0/2.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/1/0.0 link-protection
set protocols ospf area 0.0.0.0 interface xe-0/2/1.0 metric 12
set protocols ospf area 0.0.0.0 interface ge-1/0/2.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-1/0/9.0 metric 12
set protocols ospf area 0.0.0.0 interface ge-1/1/5.0 metric 13
set protocols ospf3 area 0.0.0.0 interface ge-0/0/2.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-0/1/0.0 link-protection
set protocols ospf3 area 0.0.0.0 interface xe-0/2/1.0 metric 12
set protocols ospf3 area 0.0.0.0 interface ge-1/0/2.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-1/0/9.0 metric 12
set protocols ospf3 area 0.0.0.0 interface ge-1/1/5.0 metric 13
```

R3

```
set interfaces ge-0/0/5 unit 0 family inet address 172.16.50.2/30
set interfaces ge-0/0/5 unit 0 family inet6 address 2001:db8:50:1:1::2/64
set interfaces ge-0/0/5 unit 0 family mpls
set interfaces xe-0/3/1 unit 0 family inet address 172.16.75.1/30
set interfaces xe-0/3/1 unit 0 family inet6 address 2001:db8:75:1:1::1/64
set interfaces xe-0/3/1 unit 0 family mpls
set interfaces ge-1/0/0 unit 0 family inet address 172.16.80.1/30
set interfaces ge-1/0/0 unit 0 family inet6 address 2001:db8:80:1:1::1/64
set interfaces ge-1/0/0 unit 0 family mpls
set interfaces ge-1/0/5 unit 0 family inet address 172.16.200.1/24
set interfaces ge-1/0/5 unit 0 family inet6 address 2001:db8:200:1:1::1/64
set interfaces ge-1/0/6 unit 0 family inet address 172.16.85.1/30
set interfaces ge-1/0/6 unit 0 family inet6 address 2001:db8:85:1:1::1/64
set interfaces ge-1/0/6 unit 0 family mpls
set interfaces xe-1/3/0 unit 0 family inet address 172.16.90.1/30
set interfaces xe-1/3/0 unit 0 family inet6 address 2001:db8:90:1:1::1/64
set interfaces xe-1/3/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.3.3/32 primary
set interfaces lo0 unit 0 family inet6 address 2001:db8::3:3:3:3/128 primary
set interfaces lo0 unit 0 family mpls
set routing-options srlg srlg1 srlg-value 1001
set routing-options srlg srlg2 srlg-value 1002
set routing-options srlg srlg3 srlg-value 1003
set routing-options srlg srlg4 srlg-value 1004
set routing-options srlg srlg5 srlg-value 1005
set routing-options srlg srlg6 srlg-value 1006
set routing-options srlg srlg7 srlg-value 1007
set routing-options srlg srlg8 srlg-value 1008
set routing-options srlg srlg9 srlg-value 1009
set routing-options srlg srlg10 srlg-value 10010
set routing-options srlg srlg11 srlg-value 10011
set routing-options srlg srlg12 srlg-value 10012
set routing-options router-id 172.16.3.3
set routing-options forwarding-table export ecmp
set routing-options backup-selection destination 10.1.1.0/30 interface xe-1/3/0.0
  admin-group include-all c2
set routing-options backup-selection destination 10.1.1.0/30 interface all admin-group
  exclude c3
set routing-options backup-selection destination 10.1.1.0/30 interface all srlg strict
set routing-options backup-selection destination 10.1.1.0/30 interface all protection-type
  node
```

```
set routing-options backup-selection destination 10.1.1.0/30 interface all
  bandwidth-greater-equal-primary
set routing-options backup-selection destination 10.1.1.0/30 interface all neighbor
  preference 172.16.7.7
set routing-options backup-selection destination 10.1.1.0/30 interface all root-metric
  lowest
set routing-options backup-selection destination 10.1.1.0/30 interface all metric-order
  root
set routing-options backup-selection destination 172.16.30.0/30 interface all admin-group
  exclude c5
set routing-options backup-selection destination 172.16.30.0/30 interface all srlg strict
set routing-options backup-selection destination 172.16.30.0/30 interface all
  protection-type node
set routing-options backup-selection destination 172.16.30.0/30 interface all
  bandwidth-greater-equal-primary
set routing-options backup-selection destination 172.16.30.0/30 interface all neighbor
  preference 172.16.7.7
set routing-options backup-selection destination 172.16.30.0/30 interface all root-metric
  lowest
set routing-options backup-selection destination 172.16.30.0/30 interface all metric-order
  root
set routing-options backup-selection destination 172.16.45.0/30 interface all admin-group
  exclude c5
set routing-options backup-selection destination 172.16.45.0/30 interface all srlg strict
set routing-options backup-selection destination 172.16.45.0/30 interface all
  protection-type node
set routing-options backup-selection destination 172.16.45.0/30 interface all
  bandwidth-greater-equal-primary
set routing-options backup-selection destination 172.16.45.0/30 interface all neighbor
  preference 172.16.7.7
set routing-options backup-selection destination 172.16.45.0/30 interface all root-metric
  lowest
set routing-options backup-selection destination 172.16.45.1/30 interface all metric-order
  root
set protocols rsvp interface all
set protocols mpls admin-groups c0 0
set protocols mpls admin-groups c1 1
set protocols mpls admin-groups c2 2
set protocols mpls admin-groups c3 3
set protocols mpls admin-groups c4 4
set protocols mpls admin-groups c5 5
set protocols mpls admin-groups c6 6
set protocols mpls admin-groups c7 7
set protocols mpls admin-groups c8 8
set protocols mpls admin-groups c9 9
set protocols mpls admin-groups c10 10
set protocols mpls admin-groups c11 11
set protocols mpls admin-groups c12 12
set protocols mpls admin-groups c13 13
set protocols mpls admin-groups c14 14
set protocols mpls admin-groups c15 15
set protocols mpls admin-groups c16 16
set protocols mpls admin-groups c17 17
set protocols mpls admin-groups c18 18
set protocols mpls admin-groups c19 19
set protocols mpls admin-groups c20 20
```



```
set protocols mpls admin-groups c21 21
set protocols mpls admin-groups c22 22
set protocols mpls admin-groups c23 23
set protocols mpls admin-groups c24 24
set protocols mpls admin-groups c25 25
set protocols mpls admin-groups c26 26
set protocols mpls admin-groups c27 27
set protocols mpls admin-groups c28 28
set protocols mpls admin-groups c29 29
set protocols mpls admin-groups c30 30
set protocols mpls admin-groups c31 31
set protocols mpls interface all
set protocols mpls interface ge-0/0/5.0 admin-group c0
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 link-protection
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/3/1.0 metric 21
set protocols ospf area 0.0.0.0 interface ge-1/0/0.0 metric 13
set protocols ospf area 0.0.0.0 interface ge-1/0/6.0 metric 15
set protocols ospf area 0.0.0.0 interface xe-1/3/0.0 link-protection
set protocols ospf area 0.0.0.0 interface xe-1/3/0.0 metric 22
set protocols ospf3 area 0.0.0.0 interface ge-0/0/5.0 link-protection
set protocols ospf3 area 0.0.0.0 interface ge-0/0/5.0 metric 10
set protocols ospf3 area 0.0.0.0 interface xe-0/3/1.0 metric 21
set protocols ospf3 area 0.0.0.0 interface ge-1/0/0.0 metric 13
set protocols ospf3 area 0.0.0.0 interface ge-1/0/6.0 metric 15
set protocols ospf3 area 0.0.0.0 interface xe-1/3/0.0 link-protection
set protocols ospf3 area 0.0.0.0 interface xe-1/3/0.0 metric 22
set policy-options policy-statement ecmp term 1 then load-balance per-packet
```

R4

```
set routing-options srlg srlg1 srlg-value 1001
set routing-options srlg srlg2 srlg-value 1002
set routing-options srlg srlg3 srlg-value 1003
set routing-options srlg srlg4 srlg-value 1004
set routing-options srlg srlg5 srlg-value 1005
set routing-options srlg srlg6 srlg-value 1006
set routing-options srlg srlg7 srlg-value 1007
set routing-options srlg srlg8 srlg-value 1008
set routing-options srlg srlg9 srlg-value 1009
set routing-options srlg srlg10 srlg-value 10010
set routing-options srlg srlg11 srlg-value 10011
set routing-options srlg srlg12 srlg-value 10012
set routing-options router-id 172.16.4.4
set protocols rsvp interface all
set protocols mpls admin-groups c0 0
set protocols mpls admin-groups c1 1
set protocols mpls admin-groups c2 2
set protocols mpls admin-groups c3 3
set protocols mpls admin-groups c4 4
set protocols mpls admin-groups c5 5
set protocols mpls admin-groups c6 6
set protocols mpls admin-groups c7 7
set protocols mpls admin-groups c8 8
set protocols mpls admin-groups c9 9
set protocols mpls admin-groups c10 10
set protocols mpls admin-groups c11 11
```

```
set protocols mpls admin-groups c12 12
set protocols mpls admin-groups c13 13
set protocols mpls admin-groups c14 14
set protocols mpls admin-groups c15 15
set protocols mpls admin-groups c16 16
set protocols mpls admin-groups c17 17
set protocols mpls admin-groups c18 18
set protocols mpls admin-groups c19 19
set protocols mpls admin-groups c20 20
set protocols mpls admin-groups c21 21
set protocols mpls admin-groups c22 22
set protocols mpls admin-groups c23 23
set protocols mpls admin-groups c24 24
set protocols mpls admin-groups c25 25
set protocols mpls admin-groups c26 26
set protocols mpls admin-groups c27 27
set protocols mpls admin-groups c28 28
set protocols mpls admin-groups c29 29
set protocols mpls admin-groups c30 30
set protocols mpls admin-groups c31 31
set protocols mpls interface all
set protocols ospf area 0.0.0.0 interface ge-0/1/0.0 metric 18
set protocols ospf area 0.0.0.0 interface xe-0/2/0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-1/3/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-1/1/0.0 metric 10
set protocols ospf area 0.0.0.0 interface xe-0/3/1.0 metric 21
set protocols ospf3 area 0.0.0.0 interface ge-0/1/0.0 metric 18
set protocols ospf3 area 0.0.0.0 interface xe-0/2/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface xe-1/3/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-0/0/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-1/1/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface xe-0/3/1.0 metric 21
```

R5

```
set routing-options srlg srlg1 srlg-value 1001
set routing-options srlg srlg2 srlg-value 1002
set routing-options srlg srlg3 srlg-value 1003
set routing-options srlg srlg4 srlg-value 1004
set routing-options srlg srlg5 srlg-value 1005
set routing-options srlg srlg6 srlg-value 1006
set routing-options srlg srlg7 srlg-value 1007
set routing-options srlg srlg8 srlg-value 1008
set routing-options srlg srlg9 srlg-value 1009
set routing-options srlg srlg10 srlg-value 10010
set routing-options srlg srlg11 srlg-value 10011
set routing-options srlg srlg12 srlg-value 10012
set routing-options router-id 172.16.5.5
set protocols rsvp interface all
set protocols mpls admin-groups c0 0
set protocols mpls admin-groups c1 1
set protocols mpls admin-groups c2 2
set protocols mpls admin-groups c3 3
set protocols mpls admin-groups c4 4
set protocols mpls admin-groups c5 5
set protocols mpls admin-groups c6 6
```



```
set protocols mpls admin-groups c7 7
set protocols mpls admin-groups c8 8
set protocols mpls admin-groups c9 9
set protocols mpls admin-groups c10 10
set protocols mpls admin-groups c11 11
set protocols mpls admin-groups c12 12
set protocols mpls admin-groups c13 13
set protocols mpls admin-groups c14 14
set protocols mpls admin-groups c15 15
set protocols mpls admin-groups c16 16
set protocols mpls admin-groups c17 17
set protocols mpls admin-groups c18 18
set protocols mpls admin-groups c19 19
set protocols mpls admin-groups c20 20
set protocols mpls admin-groups c21 21
set protocols mpls admin-groups c22 22
set protocols mpls admin-groups c23 23
set protocols mpls admin-groups c24 24
set protocols mpls admin-groups c25 25
set protocols mpls admin-groups c26 26
set protocols mpls admin-groups c27 27
set protocols mpls admin-groups c28 28
set protocols mpls admin-groups c29 29
set protocols mpls admin-groups c30 30
set protocols mpls admin-groups c31 31
set protocols mpls interface all
set protocols ospf area 0.0.0.0 interface xe-0/2/0.0 metric 51
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-0/0/5.0 metric 13
set protocols ospf area 0.0.0.0 interface ge-0/1/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface xe-0/2/0.0 metric 51
set protocols ospf3 area 0.0.0.0 interface ge-0/0/1.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-0/0/5.0 metric 13
set protocols ospf3 area 0.0.0.0 interface ge-0/1/0.0 metric 10
```

R6

```
set routing-options srlg srlg1 srlg-value 1001
set routing-options srlg srlg2 srlg-value 1002
set routing-options srlg srlg3 srlg-value 1003
set routing-options srlg srlg4 srlg-value 1004
set routing-options srlg srlg5 srlg-value 1005
set routing-options srlg srlg6 srlg-value 1006
set routing-options srlg srlg7 srlg-value 1007
set routing-options srlg srlg8 srlg-value 1008
set routing-options srlg srlg9 srlg-value 1009
set routing-options srlg srlg10 srlg-value 10010
set routing-options srlg srlg11 srlg-value 10011
set routing-options srlg srlg12 srlg-value 10012
set routing-options router-id 172.16.6.6
set protocols rsvp interface all
set protocols mpls admin-groups c0 0
set protocols mpls admin-groups c1 1
set protocols mpls admin-groups c2 2
set protocols mpls admin-groups c3 3
set protocols mpls admin-groups c4 4
set protocols mpls admin-groups c5 5
```

```
set protocols mpls admin-groups c6 6
set protocols mpls admin-groups c7 7
set protocols mpls admin-groups c8 8
set protocols mpls admin-groups c9 9
set protocols mpls admin-groups c10 10
set protocols mpls admin-groups c11 11
set protocols mpls admin-groups c12 12
set protocols mpls admin-groups c13 13
set protocols mpls admin-groups c14 14
set protocols mpls admin-groups c15 15
set protocols mpls admin-groups c16 16
set protocols mpls admin-groups c17 17
set protocols mpls admin-groups c18 18
set protocols mpls admin-groups c19 19
set protocols mpls admin-groups c20 20
set protocols mpls admin-groups c21 21
set protocols mpls admin-groups c22 22
set protocols mpls admin-groups c23 23
set protocols mpls admin-groups c24 24
set protocols mpls admin-groups c25 25
set protocols mpls admin-groups c26 26
set protocols mpls admin-groups c27 27
set protocols mpls admin-groups c28 28
set protocols mpls admin-groups c29 29
set protocols mpls admin-groups c30 30
set protocols mpls admin-groups c31 31
set protocols mpls interface all
set protocols ospf area 0.0.0.0 interface xe-0/3/0.0 metric 52
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 metric 12
set protocols ospf area 0.0.0.0 interface ge-0/0/4.0 metric 15
set protocols ospf area 0.0.0.0 interface xe-0/2/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface xe-0/3/0.0 metric 52
set protocols ospf3 area 0.0.0.0 interface ge-0/0/0.0 metric 12
set protocols ospf3 area 0.0.0.0 interface ge-0/0/4.0 metric 15
set protocols ospf3 area 0.0.0.0 interface xe-0/2/0.0 metric 10
```

R7

```
set routing-options srlg srlg1 srlg-value 1001
set routing-options srlg srlg2 srlg-value 1002
set routing-options srlg srlg3 srlg-value 1003
set routing-options srlg srlg4 srlg-value 1004
set routing-options srlg srlg5 srlg-value 1005
set routing-options srlg srlg6 srlg-value 1006
set routing-options srlg srlg7 srlg-value 1007
set routing-options srlg srlg8 srlg-value 1008
set routing-options srlg srlg9 srlg-value 1009
set routing-options srlg srlg10 srlg-value 10010
set routing-options srlg srlg11 srlg-value 10011
set routing-options srlg srlg12 srlg-value 10012
set routing-options router-id 172.16.7.7
set protocols rsvp interface all
set protocols mpls admin-groups c0 0
set protocols mpls admin-groups c1 1
set protocols mpls admin-groups c2 2
set protocols mpls admin-groups c3 3
set protocols mpls admin-groups c4 4
```



```
set protocols mpls admin-groups c5 5
set protocols mpls admin-groups c6 6
set protocols mpls admin-groups c7 7
set protocols mpls admin-groups c8 8
set protocols mpls admin-groups c9 9
set protocols mpls admin-groups c10 10
set protocols mpls admin-groups c11 11
set protocols mpls admin-groups c12 12
set protocols mpls admin-groups c13 13
set protocols mpls admin-groups c14 14
set protocols mpls admin-groups c15 15
set protocols mpls admin-groups c16 16
set protocols mpls admin-groups c17 17
set protocols mpls admin-groups c18 18
set protocols mpls admin-groups c19 19
set protocols mpls admin-groups c20 20
set protocols mpls admin-groups c21 21
set protocols mpls admin-groups c22 22
set protocols mpls admin-groups c23 23
set protocols mpls admin-groups c24 24
set protocols mpls admin-groups c25 26
set protocols mpls admin-groups c27 27
set protocols mpls admin-groups c28 28
set protocols mpls admin-groups c29 29
set protocols mpls admin-groups c30 30
set protocols mpls admin-groups c31 31
set protocols mpls interface all
set protocols mpls interface xe-0/3/0.0 srlg srlg8
set protocols ospf area 0.0.0.0 interface ge-0/1/5.0 metric 23
set protocols ospf area 0.0.0.0 interface xe-0/3/0.0 metric 10
set protocols ospf area 0.0.0.0 interface ge-1/0/0.0 metric 13
set protocols ospf area 0.0.0.0 interface xe-1/3/0.0 metric 22
set protocols ospf area 0.0.0.0 interface xe-1/2/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-0/1/5.0 metric 23
set protocols ospf3 area 0.0.0.0 interface xe-0/3/0.0 metric 10
set protocols ospf3 area 0.0.0.0 interface ge-1/0/0.0 metric 13
set protocols ospf3 area 0.0.0.0 interface xe-1/3/0.0 metric 22
set protocols ospf3 area 0.0.0.0 interface xe-1/2/0.0 metric 10
```

Configuring Device R3

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R3:

1. Configure the interfaces.

```
[edit interfaces]
user@R3# set ge-0/0/5 unit 0 family inet address 172.16.50.2/30
user@R3# set ge-0/0/5 unit 0 family inet6 address 2001:db8:50:1:1::2/64
user@R3# set ge-0/0/5 unit 0 family mpls

user@R3# set xe-0/3/1 unit 0 family inet address 172.16.75.1/30
```

```
user@R3# set xe-0/3/1 unit 0 family inet6 address 2001:db8:75:1:1::1/64
user@R3# set xe-0/3/1 unit 0 family mpls

user@R3# set ge-1/0/0 unit 0 family inet address 172.16.80.1/30
user@R3# set ge-1/0/0 unit 0 family inet6 address 2001:db8:80:1:1::1/64
user@R3# set ge-1/0/0 unit 0 family mpls

user@R3# set ge-1/0/5 unit 0 family inet address 172.16.200.1/24
user@R3# set ge-1/0/5 unit 0 family inet6 address 2001:db8:200:1:1::1/64

user@R3# set ge-1/0/6 unit 0 family inet address 172.16.85.1/30
user@R3# set ge-1/0/6 unit 0 family inet6 address 2001:db8:85:1:1::1/64
user@R3# set ge-1/0/6 unit 0 family mpls

user@R3# set xe-1/3/0 unit 0 family inet address 172.16.90.1/30
user@R3# set xe-1/3/0 unit 0 family inet6 address 2001:db8:90:1:1::1/64
user@R3# set xe-1/3/0 unit 0 family mpls

user@R3# set lo0 unit 0 family inet address 172.16.3.3/32 primary
user@R3# set lo0 unit 0 family inet6 address 2001:db8::3:3:3:3/128 primary
user@R3# set lo0 unit 0 family mpls
```

2. Configure srlg values.

```
[edit routing-options]
user@R3# set srlg srlg1 srlg-value 1001
user@R3# set srlg srlg2 srlg-value 1002
user@R3# set srlg srlg3 srlg-value 1003
user@R3# set srlg srlg4 srlg-value 1004
user@R3# set srlg srlg5 srlg-value 1005
user@R3# set srlg srlg6 srlg-value 1006
user@R3# set srlg srlg7 srlg-value 1007
user@R3# set srlg srlg8 srlg-value 1008
user@R3# set srlg srlg9 srlg-value 1009
user@R3# set srlg srlg10 srlg-value 10010
user@R3# set srlg srlg11 srlg-value 10011
user@R3# set srlg srlg12 srlg-value 10012
```

3. Configure the ID of the router.

```
[edit routing-options]
user@R3# set router-id 172.16.3.3
```

4. Apply the routing policy to all equal-cost multipaths exported from the routing table to the forwarding table.

```
[edit routing-options]
user@R3# set forwarding-table export ecmp
```

5. Configure attributes of the backup selection policy.

```
[edit routing-options backup-selection]
```



```
user@R3# set destination 10.1.1.0/30 interface xe-1/3/0.0 admin-group include-all
  c2
user@R3# set destination 10.1.1.0/30 interface all admin-group exclude c3
user@R3# set destination 10.1.1.0/30 interface all srlg strict
user@R3# set destination 10.1.1.0/30 interface all protection-type node
user@R3# set destination 10.1.1.0/30 interface all bandwidth-greater-equal-primary
user@R3# set destination 10.1.1.0/30 interface all neighbor preference 172.16.7.7
user@R3# set destination 10.1.1.0/30 interface all root-metric lowest
user@R3# set destination 10.1.1.0/30 interface all metric-order root

user@R3# set destination 172.16.30.0/30 interface all admin-group exclude c5
user@R3# set destination 172.16.30.0/30 interface all srlg strict
user@R3# set destination 172.16.30.0/30 interface all protection-type node
user@R3# set destination 172.16.30.0/30 interface all
  bandwidth-greater-equal-primary
user@R3# set destination 172.16.30.0/30 interface all neighbor preference 172.16.7.7
user@R3# set destination 172.16.30.0/30 interface all root-metric lowest
user@R3# set destination 172.16.30.0/30 interface all metric-order root

user@R3# set destination 192.168.45.0/30 interface all admin-group exclude c5
user@R3# set destination 192.168.45.0/30 interface all srlg strict
user@R3# set destination 192.168.45.0/30 interface all protection-type node
user@R3# set destination 192.168.45.0/30 interface all
  bandwidth-greater-equal-primary
user@R3# set destination 192.168.45.0/30 interface all neighbor preference 172.16.7.7
user@R3# set destination 192.168.45.0/30 interface all root-metric lowest
user@R3# set destination 192.168.45.0/30 interface all metric-order root
```

6. Enable RSVP on all the interfaces.

```
[edit protocols]
user@R3# set rsvp interface all
```

7. Configure administrative groups.

```
[edit protocols mpls]
user@R3# set admin-groups c0 0
user@R3# set admin-groups c1 1
user@R3# set admin-groups c2 2
user@R3# set admin-groups c3 3
user@R3# set admin-groups c4 4
user@R3# set admin-groups c5 5
user@R3# set admin-groups c6 6
user@R3# set admin-groups c7 7
user@R3# set admin-groups c8 8
user@R3# set admin-groups c9 9
user@R3# set admin-groups c10 10
user@R3# set admin-groups c11 11
user@R3# set admin-groups c12 12
user@R3# set admin-groups c13 13
user@R3# set admin-groups c14 14
user@R3# set admin-groups c15 15
user@R3# set admin-groups c16 16
user@R3# set admin-groups c17 17
```



```
user@R3# set admin-groups c18 18
user@R3# set admin-groups c19 19
user@R3# set admin-groups c20 20
user@R3# set admin-groups c21 21
user@R3# set admin-groups c22 22
user@R3# set admin-groups c23 23
user@R3# set admin-groups c24 24
user@R3# set admin-groups c25 25
user@R3# set admin-groups c26 26
user@R3# set admin-groups c27 27
user@R3# set admin-groups c28 28
user@R3# set admin-groups c29 29
user@R3# set admin-groups c30 30
user@R3# set admin-groups c31 31
```

8. Enable MPLS on all the interfaces and configure administrative group for an interface.

```
[edit protocols mpls]
user@R3# set interface all
user@R3# set interface ge-0/0/5.0 admin-group c0
```

9. Enable link protection and configure metric values on all the interfaces for an OSPF area.

```
[edit protocols ospf]
user@R3# set area 0.0.0.0 interface ge-0/0/5.0 link-protection
user@R3# set area 0.0.0.0 interface ge-0/0/5.0 metric 10
user@R3# set area 0.0.0.0 interface xe-0/3/1.0 metric 21
user@R3# set area 0.0.0.0 interface ge-1/0/0.0 metric 13
user@R3# set area 0.0.0.0 interface ge-1/0/6.0 metric 15
user@R3# set area 0.0.0.0 interface xe-1/3/0.0 link-protection
user@R3# set area 0.0.0.0 interface xe-1/3/0.0 metric 22
```

10. Enable link protection and configure metric values on all the interfaces for an OSPF3 area.

```
[edit protocols ospf3]
user@R3# set area 0.0.0.0 interface ge-0/0/5.0 link-protection
user@R3# set area 0.0.0.0 interface ge-0/0/5.0 metric 10
user@R3# set area 0.0.0.0 interface xe-0/3/1.0 metric 21
user@R3# set area 0.0.0.0 interface ge-1/0/0.0 metric 13
user@R3# set area 0.0.0.0 interface ge-1/0/6.0 metric 15
user@R3# set area 0.0.0.0 interface xe-1/3/0.0 link-protection
user@R3# set area 0.0.0.0 interface xe-1/3/0.0 metric 22
```

11. Configure the routing policy.

```
[edit policy-options]
user@R3# set policy-statement ecmp term 1 then load-balance per-packet
```


Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R3# show interfaces
ge-0/0/5 {
  unit 0 {
    family inet {
      address 192.168.50.2/30;
    }
    family inet6 {
      address 2001:db8:50:1:1::2/64;
    }
    family mpls;
  }
}
xe-0/3/1 {
  unit 0 {
    family inet {
      address 192.168.75.1/30;
    }
    family inet6 {
      address 2001:db8:75:1:1::1/64;
    }
    family mpls;
  }
}
ge-1/0/0 {
  unit 0 {
    family inet {
      address 192.168.80.1/30;
    }
    family inet6 {
      address 2001:db8:80:1:1::1/64;
    }
    family mpls;
  }
}
ge-1/0/5 {
  unit 0 {
    family inet {
      address 172.16.200.1/24;
    }
    family inet6 {
      address 2001:db8:200:1:1::1/64;
    }
  }
}
ge-1/0/6 {
  unit 0 {
    family inet {
```

```
            address 192.168.85.1/30;
        }
        family inet6 {
            address 2001:db8:85:1:1::1/64;
        }
        family mpls;
    }
}
xe-1/3/0 {
    unit 0 {
        family inet {
            address 192.168.90.1/30;
        }
        family inet6 {
            address 2001:db8:90:1:1::1/64;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        family inet {
            address 172.16.3.3/32 {
                primary;
            }
        }
        family inet6 {
            address 2001:db8:3:3:3:3/128 {
                primary;
            }
        }
        family mpls;
    }
}
```

```
user@R3# show protocols
rsvp {
    interface all;
}
mpls {
    admin-groups {
        c0 0;
        c1 1;
        c2 2;
        c3 3;
        c4 4;
        c5 5;
        c6 6;
        c7 7;
        c8 8;
        c9 9;
        c10 10;
        c11 11;
        c12 12;
        c13 13;
        c14 14;
```



```
        c15 15;
        c16 16;
        c17 17;
        c18 18;
        c19 19;
        c20 20;
        c21 21;
        c22 22;
        c23 23;
        c24 24;
        c25 25;
        c26 26;
        c27 27;
        c28 28;
        c29 29;
        c30 30;
        c31 31;
    }
    interface all;
    interface ge-0/0/5.0 {
        admin-group c0;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-0/0/5.0 {
            link-protection;
            metric 10;
        }
        interface xe-0/3/1.0 {
            metric 21;
        }
        interface ge-1/0/0.0 {
            metric 13;
        }
        interface ge-1/0/6.0 {
            metric 15;
        }
        interface xe-1/3/0.0 {
            link-protection;
            metric 22;
        }
    }
}
ospf3 {
    area 0.0.0.0 {
        interface ge-0/0/5.0 {
            link-protection;
            metric 10;
        }
        interface xe-0/3/1.0 {
            metric 21;
        }
        interface ge-1/0/0.0 {
            metric 13;
        }
```

```
        interface ge-1/0/6.0 {
            metric 15;
        }
        interface xe-1/3/0.0 {
            link-protection;
            metric 22;
        }
    }
}
```

```
user@R3# show routing-options
srlg {
    srlg1 srlg-value 1001;
    srlg2 srlg-value 1002;
    srlg3 srlg-value 1003;
    srlg4 srlg-value 1004;
    srlg5 srlg-value 1005;
    srlg6 srlg-value 1006;
    srlg7 srlg-value 1007;
    srlg8 srlg-value 1008;
    srlg9 srlg-value 1009;
    srlg10 srlg-value 10010;
    srlg11 srlg-value 10011;
    srlg12 srlg-value 10012;
}
router-id 172.16.3.3;
forwarding-table {
    export ecmp;
}
backup-selection {
    destination 10.1.1.0/30 {
        interface xe-1/3/0.0 {
            admin-group {
                include-all c2;
            }
        }
        interface all {
            admin-group {
                exclude c3;
            }
            srlg strict;
            protection-type node;
            bandwidth-greater-equal-primary;
            node {
                preference 172.16.7.7;
            }
            root-metric lowest;
            metric-order root;
        }
    }
    destination 172.16.30.0/30 {
        interface all {
            admin-group {
                exclude c5;
            }
            srlg strict;
```



```
            protection-type node;
            bandwidth-greater-equal-primary;
            node {
               preference 172.16.7.7;
            }
            root-metric lowest;
            metric-order root;
         }
      }
      destination 192.168.45.0/30 {
         interface all {
            admin-group {
               exclude c5;
            }
            srlg strict;
            protection-type node;
            bandwidth-greater-equal-primary;
            node {
               preference 172.16.7.7;
            }
            root-metric lowest;
            metric-order root;
         }
      }
   }
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the Routes

Purpose Verify that the expected routes are learned.

Action From operational mode, run the **show route** command for the routing table.

```
user@R3> show route
inet.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.3.3/32         *[Direct/0] 02:22:27
                    > via lo0.0
10.4.0.0/16        *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.5.0.0/16        *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.6.128.0/17      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.9.0.0/16        *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.10.0.0/16       *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.13.4.0/23       *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.13.10.0/23      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.82.0.0/15       *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.84.0.0/16       *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.85.12.0/22      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.92.0.0/16       *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.92.16.0/20      *[Direct/0] 02:22:57
                    > via fxp0.0
10.92.24.195/32    *[Local/0] 02:22:57
                      Local via fxp0.0
10.94.0.0/16       *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.99.0.0/16       *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.102.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.150.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.155.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.157.64.0/19     *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.160.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.204.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.205.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.206.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.207.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.209.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.212.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
```



```
10.213.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.214.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.215.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.216.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.218.13.0/24     *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.218.14.0/24     *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.218.16.0/20     *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.218.32.0/20     *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
10.227.0.0/16      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
172.16.50.0/30       *[Direct/0] 02:19:55
                    > via ge-0/0/5.0
172.16.50.2/32       *[Local/0] 02:19:58
                      Local via ge-0/0/5.0
172.16.75.0/30       *[Direct/0] 02:19:55
                    > via xe-0/3/1.0
172.16.75.1/32       *[Local/0] 02:19:57
                      Local via xe-0/3/1.0
172.16.24.195/32   *[Direct/0] 02:22:57
                    > via lo0.0
172.16.0.0/12      *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
192.168.0.0/16     *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
192.168.102.0/23   *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
192.168.136.0/24    *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
192.168.136.192/32  *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
192.168.137.0/24    *[Static/5] 02:22:57
                    > to 10.92.31.254 via fxp0.0
192.168.233.5/32       *[OSPF/10] 00:16:55, metric 1
                      MultiRecv

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

47.0005.80ff.f800.0000.0108.0001.1280.9202.4195/152
                   *[Direct/0] 02:22:57
                    > via lo0.0

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 00:16:55, metric 1
                      Receive
1                  *[MPLS/0] 00:16:55, metric 1
                      Receive
2                  *[MPLS/0] 00:16:55, metric 1
                      Receive
13                 *[MPLS/0] 00:16:55, metric 1
                      Receive
```

```
inet6.0: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8:50:1:1::/64   *[Direct/0] 02:19:44
                    > via ge-0/0/5.0
2001:db8:50:1:1::2/128 *[Local/0] 02:19:58
                      Local via ge-0/0/5.0
2001:db8:75:1:1::/64   *[Direct/0] 02:19:44
                    > via xe-0/3/1.0
2001:db8:75:1:1::1/128 *[Local/0] 02:19:57
                      Local via xe-0/3/1.0
2001:db8::3:3:3:3/128  *[Direct/0] 02:22:27
                    > via lo0.0
2001:db8::128:92:24:195/128
                   *[Direct/0] 02:22:57
                    > via lo0.0
fe80::/64          *[Direct/0] 02:19:44
                    > via ge-0/0/5.0
                    [Direct/0] 02:19:43
                    > via xe-0/3/1.0
fe80::205:86ff:fe00:ed05/128
                   *[Local/0] 02:19:58
                      Local via ge-0/0/5.0
fe80::205:86ff:fe00:ed3d/128
                   *[Local/0] 02:19:57
                      Local via xe-0/3/1.0
fe80::5668:a50f:fcc1:3ca2/128
                   *[Direct/0] 02:22:57
                    > via lo0.0
```

Meaning The output shows all Device R3 routes.

Verifying the OSPF Route

Purpose Verify the routing table of OSPF.

Action From operational mode, run the **show ospf route detail** command for Device R3.

```
user@R3> show ospf route detail
Topology default Route Table:

Prefix             Path  Route      NH       Metric NextHop       Nexthop
                   Type  Type       Type            Interface     Address/LSP
172.16.50.0/30        Intra Network    IP            10 ge-0/0/5.0
  area 0.0.0.0, origin 172.16.3.3, priority low
172.16.75.0/30        Intra Network    IP            21 xe-0/3/1.0
  area 0.0.0.0, origin 172.16.3.3, priority low
```

Meaning The output displays the routing table of OSPF routers.

Verifying the OSPF3 Route

Purpose Verify the routing table of OSPF3.

Action From operational mode, run the **show ospf3 route detail** command for Device R3.

```
user@R3> show ospf3 route detail

Prefix                                        Path  Route      NH   Metric
                                              Type  Type       Type
2001:db8:50:1:1::/64                              Intra Network    IP   10
  NH-interface ge-0/0/5.0
  Area 0.0.0.0, Origin 172.16.3.3, Priority low
2001:db8:75:1:1::/64                              Intra Network    IP   21
  NH-interface xe-0/3/1.0
  Area 0.0.0.0, Origin 172.16.3.3, Priority low
```

Meaning The output displays the routing table of OSPF3 routers.

Verifying the Backup Selection Policy for Device R3

Purpose Verify the backup selection policy for Device R3.

Action From operational mode, run the **show backup-selection** command for Device R3.

```
user@R3> show backup-selection
Prefix: 10.1.1.0/30
  Interface: all
    Admin-group exclude: c3
    Neighbor preference: 172.16.7.7
    Protection Type: Node, Downstream Paths Only: Disabled, SRLG: Strict, B/w >=
 Primary: Enabled, Root-metric: lowest, Dest-metric: lowest
    Metric Evaluation Order: Root-metric, Dest-metric
    Policy Evaluation Order: Admin-group, SRLG, Bandwidth, Protection, node,
Metric
  Interface: xe-1/3/0.0
    Admin-group include-all: c2
    Protection Type: Link, Downstream Paths Only: Disabled, SRLG: Loose, B/w >=
Primary: Disabled, Root-metric: lowest, Dest-metric: lowest
    Metric Evaluation Order: Dest-metric, Root-metric
    Policy Evaluation Order: Admin-group, SRLG, Bandwidth, Protection, node,
Metric Prefix: 172.16.30.0/30
  Interface: all
    Admin-group exclude: c5
    Neighbor preference: 172.16.7.7
    Protection Type: Node, Downstream Paths Only: Disabled, SRLG: Strict, B/w >=
 Primary: Enabled, Root-metric: lowest, Dest-metric: lowest
    Metric Evaluation Order: Root-metric, Dest-metric
    Policy Evaluation Order: Admin-group, SRLG, Bandwidth, Protection, node,
Metric
Prefix: 172.16.45.0/30
  Interface: all
    Admin-group exclude: c5
```

```
    Neighbor preference: 172.16.7.7
    Protection Type: Node, Downstream Paths Only: Disabled, SRLG: Strict, B/w >=
 Primary: Enabled, Root-metric: lowest, Dest-metric: lowest
    Metric Evaluation Order: Root-metric, Dest-metric
    Policy Evaluation Order: Admin-group, SRLG, Bandwidth, Protection, node,
Metric
```

Meaning The output displays the configured policies per prefix per primary next-hop interface.

Related Documentation

- Configuring Backup Selection Policy for the OSPF Protocol on page 154
- Understanding Backup Selection Policy for OSPF Protocol on page 152
- *backup-selection (Protocols ISIS)*

CHAPTER 4

Evaluating Complex Cases Using Policy Chains and Subroutines

Understanding How a Routing Policy Chain Is Evaluated

Figure 15 on page 186 shows how a chain of routing policies is evaluated. These routing policies consist of multiple terms. Each term consists of match conditions and actions to apply to matching routes. Each route is evaluated against the policies as follows:

1. The route is evaluated against the first term in the first routing policy. If it matches, the specified action is taken. If the action is to accept or reject the route, that action is taken and the evaluation of the route ends. If the **next term** action is specified, if no action is specified, or if the route does not match, the evaluation continues as described in Step 2. If the **next policy** action is specified, any accept or reject action specified in this term is skipped, all remaining terms in this policy are skipped, all other actions are taken, and the evaluation continues as described in Step 3.

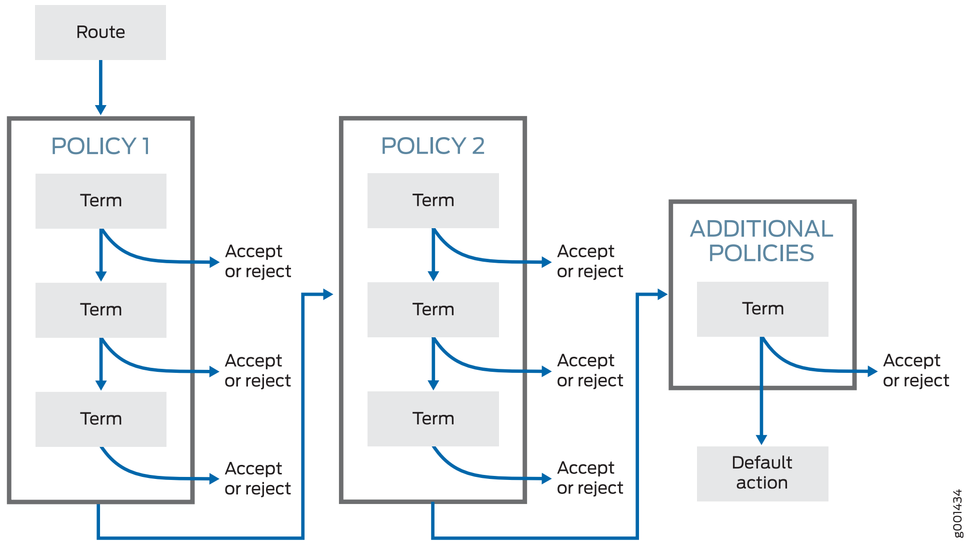
2. The route is evaluated against the second term in the first routing policy. If it matches, the specified action is taken. If the action is to accept or reject the route, that action is taken and the evaluation of the route ends. If the **next term** action is specified, if no action is specified, or if the route does not match, the evaluation continues in a similar manner against the last term in the first routing policy. If the **next policy** action is specified, any accept or reject action specified in this term is skipped, all remaining terms in this policy are skipped, all other actions are taken, and the evaluation continues as described in Step 3.

3. If the route does not match a term or matches a term with a **next policy** action in the first routing policy, it is evaluated against the first term in the second routing policy.

4. The evaluation continues until the route matches a term with an accept or reject action defined or until there are no more routing policies to evaluate. If there are no

more routing policies, then the accept or reject action specified by the default policy is taken.

Figure 15: Routing Policy Chain Evaluation



Related Documentation

- Default Routing Policies on page 30
- Example: Configuring Policy Chains and Route Filters on page 186

Example: Configuring Policy Chains and Route Filters

A *policy chain* is the application of multiple policies within a specific section of the configuration. A *route filter* is a collection of match prefixes.

- Requirements on page 186
- Overview on page 186
- Configuration on page 188
- Verification on page 195

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

An example of a policy chain applied to BGP is as follows:

```
user@R1# show protocols bgp
group int {
  type internal;
  local-address 192.168.0.1;
  export [ adv-statics adv-large-aggregates adv-small-aggregates ];
```



```
    neighbor 192.168.0.2;
    neighbor 192.168.0.3;
  }
```

The **adv-statics**, **adv-large-aggregates**, and **adv-small-aggregates** policies, in addition to the default BGP policy, make up the policy chain applied to the BGP peers of Device R1. Two of the policies demonstrate route filters with different match types. The other policy matches all static routes, so no route filter is needed.

```
user@R1# show policy-options
policy-statement adv-large-aggregates {
  term between-16-and-18 {
    from {
      protocol aggregate;
      route-filter 172.16.0.0/16 upto /18;
    }
    then accept;
  }
}
policy-statement adv-small-aggregates {
  term between-19-and-24 {
    from {
      protocol aggregate;
      route-filter 172.16.0.0/16 prefix-length-range /19-/24;
    }
    then accept;
  }
}
policy-statement adv-statics {
  term statics {
    from protocol static;
    then accept;
  }
}
```

Optionally, you can convert this policy chain into a single multiterm policy for the internal BGP (IBGP) peers. If you do this, one of the advantages of a policy chain is lost—the ability to reuse policies for different purposes.

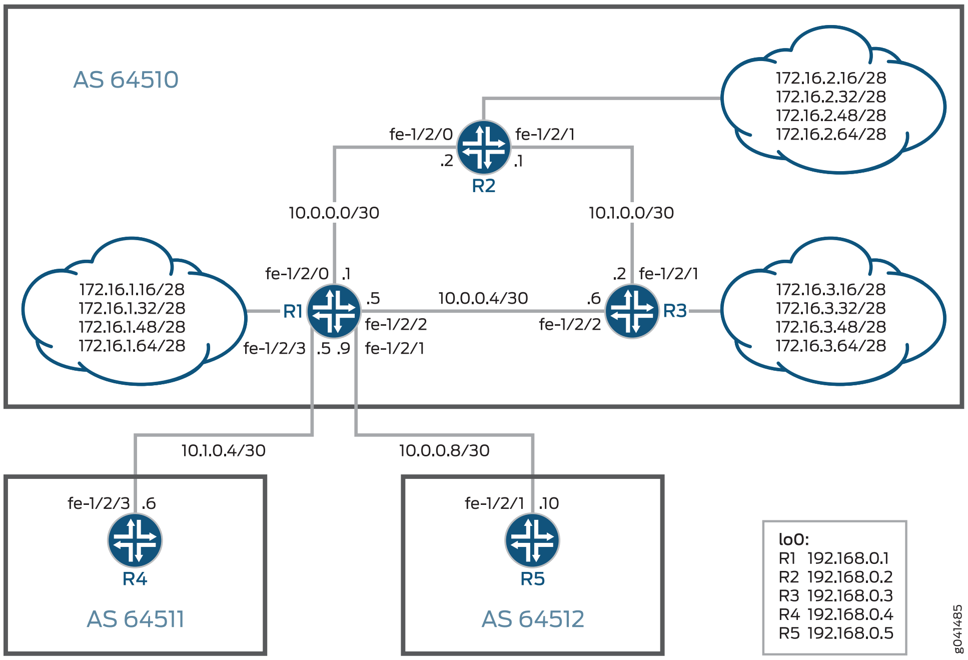
Figure 16 on page 188 displays Device R1 in AS 64510 with its IBGP peers, Device R2 and Device R3. Device R1 also has external BGP (EBGP) connections to Device R4 in AS 64511 and Device R5 in AS 64512. The current administrative policy within AS 64510 is to send the customer static routes only to other IBGP peers. Any EBGP peer providing transit service only receives aggregate routes with mask lengths smaller than 18 bits. Any EBGP peer providing peering services receives all customer routes and all aggregates whose mask length is larger than 19 bits. Each portion of these administrative policies is configured in a separate routing policy within the **[edit policy-opitons]** configuration hierarchy. These policies provide the administrators of AS 64510 with multiple configuration options for advertising routes to peers.

Device R4 is providing transit service to AS 64510, which allows the AS to advertise its assigned routing space to the Internet. On the other hand, the peering service provided by Device R5 allows AS 64510 to route traffic directly between the autonomous systems (ASs) for all customer routes.

Topology

Figure 16 on page 188 shows the sample network.

Figure 16: BGP Topology for Policy Chains



"CLI Quick Configuration" on page 188 shows the configuration for all of the devices in Figure 16 on page 188.

The section "Step-by-Step Procedure" on page 190 describes the steps on Device R1.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 description to_R2
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/2 unit 0 description to_R3
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30
set interfaces fe-1/2/3 unit 0 description to_R4
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30
set interfaces fe-1/2/1 unit 0 description to_R5
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.10/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.1
set protocols bgp group int export adv-statics
set protocols bgp group int export adv-large-aggregates
set protocols bgp group int export adv-small-aggregates
```



```
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group to_64511 type external
set protocols bgp group to_64511 export adv-large-aggregates
set protocols bgp group to_64511 neighbor 10.1.0.6 peer-as 64511
set protocols bgp group to_64512 type external
set protocols bgp group to_64512 export adv-small-aggregates
set protocols bgp group to_64512 export adv-statics
set protocols bgp group to_64512 neighbor 10.0.0.9 peer-as 64512
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement adv-large-aggregates term between-16-and-18 from
  protocol aggregate
set policy-options policy-statement adv-large-aggregates term between-16-and-18 from
  route-filter 172.16.0.0/16 upto /18
set policy-options policy-statement adv-large-aggregates term between-16-and-18 then
  accept
set policy-options policy-statement adv-small-aggregates term between-19-and-24
  from protocol aggregate
set policy-options policy-statement adv-small-aggregates term between-19-and-24
  from route-filter 172.16.0.0/16 prefix-length-range /19-/24
set policy-options policy-statement adv-small-aggregates term between-19-and-24
  then accept
set policy-options policy-statement adv-statics term statics from protocol static
set policy-options policy-statement adv-statics term statics then accept
set routing-options static route 172.16.1.16/28 discard
set routing-options static route 172.16.1.32/28 discard
set routing-options static route 172.16.1.48/28 discard
set routing-options static route 172.16.1.64/28 discard
set routing-options aggregate route 172.16.0.0/16
set routing-options aggregate route 172.16.1.0/24
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device R2

```
set interfaces fe-1/2/0 unit 0 description to_R1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 description to_R3
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int neighbor 192.168.0.1 export send-static-aggregate
set protocols bgp group int neighbor 192.168.0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static-aggregate term 1 from protocol static
set policy-options policy-statement send-static-aggregate term 1 from protocol aggregate
set policy-options policy-statement send-static-aggregate term 1 then accept
set routing-options static route 172.16.2.16/28 discard
set routing-options static route 172.16.2.32/28 discard
set routing-options static route 172.16.2.48/28 discard
set routing-options static route 172.16.2.64/28 discard
set routing-options aggregate route 172.16.2.0/24
```

```
set routing-options aggregate route 172.16.0.0/16
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/1 unit 0 description to_R2
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/2 unit 0 description to_R1
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.3
set protocols bgp group int neighbor 192.168.0.1 export send-static-aggregate
set protocols bgp group int neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static-aggregate from protocol static
set policy-options policy-statement send-static-aggregate from protocol aggregate
set policy-options policy-statement send-static-aggregate then accept
set routing-options static route 172.16.3.16/28 discard
set routing-options static route 172.16.3.32/28 discard
set routing-options static route 172.16.3.48/28 discard
set routing-options static route 172.16.3.64/28 discard
set routing-options aggregate route 172.16.0.0/16
set routing-options aggregate route 172.16.3.0/24
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64510
```

Device R4

```
set interfaces fe-1/2/3 unit 0 description to_R1
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.1.0.5
set routing-options autonomous-system 64511
```

Device R5

```
set interfaces fe-1/2/1 unit 0 description to_R1
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.9/30
set interfaces lo0 unit 0 family inet address 192.168.0.5/32
set protocols bgp group ext type external
set protocols bgp group ext neighbor 10.0.0.10 peer-as 64510
set routing-options autonomous-system 64512
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 0 description to_R2
```

```
user@R1# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30

user@R1# set fe-1/2/2 unit 0 description to_R3
user@R1# set fe-1/2/2 unit 0 family inet address 10.0.0.5/30

user@R1# set fe-1/2/3 unit 0 description to_R4
user@R1# set fe-1/2/3 unit 0 family inet address 10.1.0.5/30

user@R1# set fe-1/2/1 unit 0 description to_R5
user@R1# set fe-1/2/1 unit 0 family inet address 10.0.0.10/30

user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure the IBGP connections to Device R2 and Device R3.

```
[edit protocols bgp group int]
user@R1# set type internal
user@R1# set local-address 192.168.0.1
user@R1# set neighbor 192.168.0.2
user@R1# set neighbor 192.168.0.3
```

3. Apply the export policies for the internal peers.

```
[edit protocols bgp group int]
user@R1# set export adv-statics
user@R1# set export adv-large-aggregates
user@R1# set export adv-small-aggregates
```

4. Configure the EBGP connection to Device R4.

```
[edit protocols bgp group to_64511]
user@R1# set type external
user@R1# set neighbor 10.1.0.6 peer-as 64511
```

5. Apply the export policy for Device R4.

```
[edit protocols bgp group to_64511]
user@R1# set export adv-large-aggregates
```

6. Configure the EBGP connection to Device R5.

```
[edit protocols bgp group to_64512]
user@R1# set type external
user@R1# set neighbor 10.0.0.9 peer-as 64512
```

7. Apply the export policies for Device R5.

```
[edit protocols bgp group to_64512]
user@R1# set export adv-small-aggregates
user@R1# set export adv-statics
```

8. Configure OSPF connections to Device R2 and Device R3.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.0
user@R1# set interface fe-1/2/2.0
user@R1# set interface lo0.0 passive
```

9. Configure the routing policies.

```
[edit policy-options policy-statement adv-large-aggregates term between-16-and-18]
user@R1# set from protocol aggregate
user@R1# set from route-filter 172.16.0.0/16 upto /18
user@R1# set then accept

[edit policy-options policy-statement adv-small-aggregates term
  between-19-and-24]
user@R1# set from protocol aggregate
user@R1# set from route-filter 172.16.0.0/16 prefix-length-range /19-/24
user@R1# set then accept

[edit policy-options policy-statement adv-statics term statics]
user@R1# set from protocol static
user@R1# set then accept
```

10. Configure the static and aggregate routes.

```
[edit routing-options static]
user@R1# set route 172.16.1.16/28 discard
user@R1# set route 172.16.1.32/28 discard
user@R1# set route 172.16.1.48/28 discard
user@R1# set route 172.16.1.64/28 discard

[edit routing-options aggregate]
user@R1# set route 172.16.0.0/16
user@R1# set route 172.16.1.0/24
```

11. Configure the autonomous system (AS) number and router ID.

```
[edit routing-options]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 64510
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
fe-1/2/0 {
  unit 0 {
    description to_R2;
    family inet {
      address 10.0.0.1/30;
    }
```

```
    }
}
fe-1/2/2 {
    unit 0 {
        description to_R3;
        family inet {
            address 10.0.0.5/30;
        }
    }
}
fe-1/2/3 {
    unit 0 {
        description to_R4;
        family inet {
            address 10.1.0.5/30;
        }
    }
}
fe-1/2/1 {
    unit 0 {
        description to_R5;
        family inet {
            address 10.0.0.10/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
    }
}

user@R1# show protocols
bgp {
    group int {
        type internal;
        local-address 192.168.0.1;
        export [ adv-statics adv-large-aggregates adv-small-aggregates ];
        neighbor 192.168.0.2;
        neighbor 192.168.0.3;
    }
    group to_64511 {
        type external;
        export adv-large-aggregates;
        neighbor 10.1.0.6 {
            peer-as 64511;
        }
    }
    group to_64512 {
        type external;
        export [ adv-small-aggregates adv-statics ];
        neighbor 10.0.0.9 {
            peer-as 64512;
        }
```

```
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/0.0;
        interface fe-1/2/2.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement adv-large-aggregates {
    term between-16-and-18 {
        from {
            protocol aggregate;
            route-filter 172.16.0.0/16 upto /18;
        }
        then accept;
    }
}
policy-statement adv-small-aggregates {
    term between-19-and-24 {
        from {
            protocol aggregate;
            route-filter 172.16.0.0/16 prefix-length-range /19-/24;
        }
        then accept;
    }
}
policy-statement adv-statics {
    term statics {
        from protocol static;
        then accept;
    }
}
```

```
user@R1# show routing-options
static {
    route 172.16.1.16/28 discard;
    route 172.16.1.32/28 discard;
    route 172.16.1.48/28 discard;
    route 172.16.1.64/28 discard;
}
aggregate {
    route 172.16.0.0/16;
    route 172.16.1.0/24;
}
router-id 192.168.0.1;
autonomous-system 64510;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the Route Advertisement to Device R4

Purpose On Device R1, make sure that the customer routes are advertised to Device R4.

Action user@R1> **show route advertising-protocol bgp 10.1.0.6**

```
inet.0: 29 destinations, 31 routes (29 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.0.0/16           Self                                    I
* 172.16.2.0/24           Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.0/24           Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```

Meaning The **adv-large-aggregates** policy is applied to the peering session with Device R4 to advertise the aggregate routes with a subnet mask length between 16 and 18 bits. The 172.16.0.0/16 aggregate route is being sent as defined by the administrative policy, but a number of other routes with larger subnet masks are also being sent to Device R4.

Checking Where the Longer Routes Are Originating

Purpose On Device R1, find where the other routes are coming from.

Action

```
user@R1> show route 172.16.3.16/28

inet.0: 29 destinations, 31 routes (29 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.3.16/28     *[BGP/170] 20:16:00, localpref 100, from 192.168.0.3
                      AS path: I, validation-state: unverified
                    > to 10.0.0.6 via fe-1/2/2.0
```

Meaning Device R1 has learned this route through its BGP session with Device R3. Because it is an active BGP route, it is automatically advertised by the BGP default policy. Remember that the default policy is always applied to the end of every policy chain. What is needed is a policy to block the more specific routes from being advertised.

Blocking the More Specific Routes

Purpose Create a policy called **not-larger-than-18** that rejects all routes within the 172.16.0.0 /16 address space that have a subnet mask length greater than or equal to 19 bits. This ensures that all aggregates with a mask between 16 and 18 bits are advertised, thus accomplishing the goal of the administrative policy.

Action

1. On Device R1, configure the **not-larger-than-18** policy.

   ```
   [edit policy-options policy-statement not-larger-than-18 term
     reject-greater-than-18-bits]
   user@R1# set from route-filter 172.16.0.0/16 prefix-length-range /19-/32
   user@R1# set then reject
   ```

2. On Device R1, apply the policy to the peering session with Device R4.

   ```
   [edit protocols bgp group to_64511]
   user@R1# set export not-larger-than-18
   user@R1# commit
   ```

3. On Device R1, check which routes are advertised to Device R4.

   ```
   user@R1> show route advertising-protocol bgp 10.1.0.6

   inet.0: 29 destinations, 31 routes (29 active, 0 holddown, 0 hidden)
     Prefix                  Nexthop              MED     Lclpref    AS path
   * 172.16.0.0/16           Self                                    I
   ```

Meaning The policy chain is working correctly. Only the 172.16.0.0 /16 route is advertised to Device R4.

Verifying the Route Advertisement to Device R5

Purpose On Device R1, make sure that the customer routes are advertised to Device R5.

Device R5 is Device R1's EBGP peer in AS 64512. The administrative policy states that this peer receives only aggregate routes larger than 18 bits in length and all customer routes. In anticipation of encountering a problem similar to the problem on Device R4, you can create a policy called **not-smaller- than-18** that rejects all aggregates with mask lengths between 16 and 18 bits.

Action

1. On Device R2, configure an aggregate route for 172.16.128.0/17.

   [edit routing-options aggregate]
   user@R2# **set route 172.16.128.0/17 discard**
   user@R2# **commit**

2. On Device R1, check which routes are advertised to Device R5.

```
user@R1> show route advertising-protocol bgp 10.0.0.9

inet.0: 30 destinations, 32 routes (30 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.0/24           Self                                    I
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.0/24           Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.0/24           Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
* 172.16.128.0/17         Self                                    I
```

   The aggregate route 172.16.128.0/17 is advertised, in violation of the administrative policy

3. On Device R1, configure the **not-smaller-than-18** policy.

   [edit policy-options policy-statement not-smaller-than-18 term reject-less-than-18-bits]
   user@R1# **set from protocol aggregate**
   user@R1# **set from route-filter 172.16.0.0/16 upto /18**
   user@R1# **set then reject**

4. On Device R1, apply the policy to the peering session with Device R5.

   [edit protocols bgp group to_64512]
   user@R1# **set export not-smaller-than-18**
   user@R1# **commit**

5. On Device R1, check which routes are advertised to Device R5.

```
user@R1> show route advertising-protocol bgp 10.0.0.9

inet.0: 29 destinations, 31 routes (29 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.0/24           Self                                    I
```

```
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.0/24           Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.0/24           Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```

Meaning The policy chain is working correctly. Only aggregate routes larger than 18 bits in length and all customer routes are advertised to Device R5.

Related Documentation

- Understanding Route Filters for Use in Routing Policy Match Conditions on page 213
- Route Filter Match Conditions on page 53
- Example: Configuring Routing Policy Prefix Lists on page 282
- Example: Configuring a Policy Subroutine on page 203

Understanding Policy Subroutines in Routing Policy Match Conditions

You can use a routing policy called from another routing policy as a match condition. This process makes the called policy a *subroutine*.

In some ways, the Junos OS policy framework is similar to a programming language. This similarity includes the concept of nesting policies into a policy subroutine. A subroutine in a software program is a section of code that you reference on a regular basis. A policy subroutine works in the same fashion—you reference an existing policy as a match criterion in another policy. The routing device first evaluates the subroutine and then evaluates the main policy. The evaluation of the subroutine returns a true or false Boolean result to the main policy. Because you are referencing the subroutine as a match criterion, a true result means that the main policy has a match and can perform any configured actions. A false result from the subroutine, however, means that the main policy does not have a match.

Configuring Subroutines

To configure a subroutine in a routing policy to be called from another routing policy, create the subroutine and specify its name using the **policy** match condition in the **from** or **to** statement of another routing policy.

NOTE: Do not evaluate a routing policy within itself. The result is that no prefixes ever match the routing policy.

The action specified in a subroutine is used to provide a match condition to the calling policy. If the subroutine specifies an action of accept, the calling policy considers the route to be a match. If the subroutine specifies an action of reject, the calling policy considers the route not to match. If the subroutine specifies an action that is meant to manipulate the route characteristics, the changes are made.

Possible Consequences of Termination Actions in Subroutines

A subroutine with particular statements can behave differently from a routing policy that contains the same statements. With a subroutine, you must remember that the possible termination actions of accept or reject specified by the subroutine or the default policy can greatly affect the expected results.

In particular, you must consider what happens if a match does not occur with routes specified in a subroutine and if the default policy action that is taken is the action that you expect and want.

For example, imagine that you are a network administrator at an Internet service provider (ISP) that provides service to Customer A. You have configured several routing policies for the different classes of neighbors that Customer A presents on various links. To save time maintaining the routing policies for Customer A, you have configured a subroutine that identifies their routes and various routing policies that call the subroutine, as shown below:

```
[edit]
policy-options {
  policy-statement customer-a-subroutine {
    from {
      route-filter 10.1/16 exact;
      route-filter 10.5/16 exact;
      route-filter 192.168.10/24 exact;
    }
    then accept;
  }
}
policy-options {
  policy-statement send-customer-a-default {
    from {
      policy customer-a-subroutine;
    }
    then {
      set metric 500;
      accept;
    }
  }
}
policy-options {
  policy-statement send-customer-a-primary {
    from {
      policy customer-a-subroutine;
    }
    then {
      set metric 100;
      accept;
```

```
        }
      }
    }
    policy-options {
      policy-statement send-customer-a-secondary {
        from {
          policy customer-a-subroutine;
        }
        then {
          set metric 200;
          accept;
        }
      }
    }
    protocols {
      bgp {
        group customer-a {
          export send-customer-a-default;
          neighbor 10.1.1.1;
          neighbor 10.1.2.1;
          neighbor 10.1.3.1 {
            export send-customer-a-primary;
          }
          neighbor 10.1.4.1 {
            export send-customer-a-secondary;
          }
        }
      }
    }
```

The following results occur with this configuration:

- The group-level **export** statement resets the metric to 500 when advertising all BGP routes to neighbors 10.1.1.1 and 10.1.2.1 rather than just the routes that match the subroutine route filters.
- The neighbor-level **export** statements reset the metric to 100 and 200 when advertising all BGP routes to neighbors 10.1.3.1 and 10.1.4.1, respectively, rather than just the BGP routes that match the subroutine route filters.

These unexpected results occur because the subroutine policy does not specify a termination action for routes that do not match the route filter and therefore, the default BGP export policy of accepting all BGP routes is taken.

If the statements included in this particular subroutine had been contained within the calling policies themselves, only the desired routes would have their metrics reset.

This example illustrates the differences between routing policies and subroutines and the importance of the termination action in a subroutine. Here, the default BGP export policy action for the subroutine was not carefully considered. A solution to this particular example is to add one more term to the subroutine that rejects all other routes that do not match the route filters:

```
[edit]
policy-options {
```



```
    policy-statement customer-a-subroutine {
      term accept-exact {
        from {
          route-filter 10.1/16 exact;
          route-filter 10.5/16 exact;
          route-filter 192.168.10/24 exact;
        }
        then accept;
      }
      term reject-others {
        then reject;
      }
    }
  }
```

Termination action strategies for subroutines in general include the following:

- Depend upon the default policy action to handle all other routes.
- Add a term that accepts all other routes.
- Add a term that rejects all other routes.

The option that you choose depends upon what you want to achieve with your subroutine. Plan your subroutines carefully.

Related Documentation

- How a Routing Policy Subroutine Is Evaluated on page 201
- Example: Configuring a Policy Subroutine on page 203

How a Routing Policy Subroutine Is Evaluated

Figure 17 on page 203 shows how a subroutine is evaluated. The subroutine is included in the first term of the first routing policy in a chain. Each route is evaluated against the subroutine as follows:

1. The route is evaluated against the first term in the first routing policy. If the route does not match all match conditions specified before the subroutine, the subroutine is skipped and the next term in the routing policy is evaluated (see Step 2). If the route matches all match conditions specified before the subroutine, the route is evaluated against the subroutine. If the route matches the match conditions in any of the subroutine terms, two levels of evaluation occur in the following order:

   a. The actions in the subroutine term are evaluated. If one of the actions is accept, evaluation of the subroutine ends and a Boolean value of TRUE is returned to the calling policy. If one of the actions is reject, evaluation of the subroutine ends and FALSE is returned to the calling policy.

      If the subroutine does not specify the **accept**, **reject** or **next-policy** action, it uses the **accept** or **reject** action specified by the default policy, and the values of TRUE or FALSE are returned to the calling policy as described in the previous paragraph.

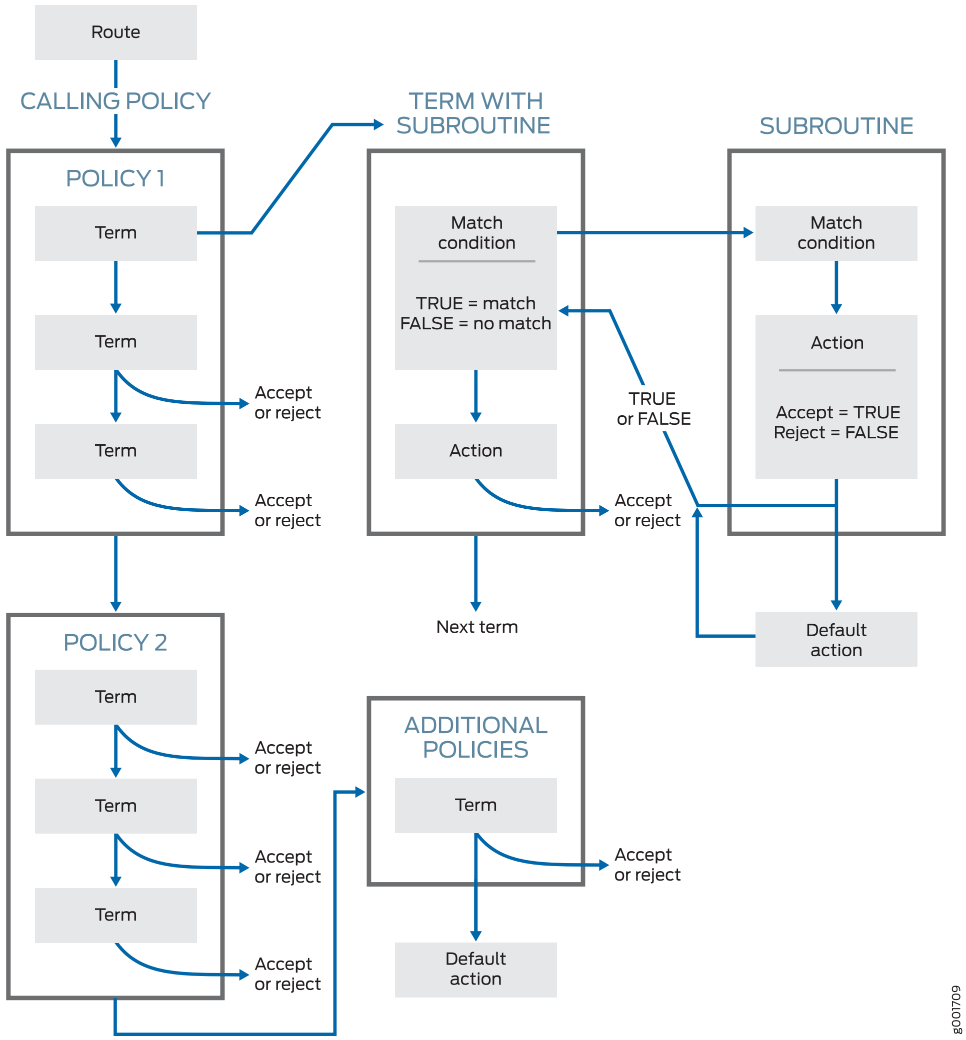
   b. The calling policy's subroutine match condition is evaluated. During this part of the evaluation, TRUE equals a match and FALSE equals no match. If the subroutine returns TRUE to the calling policy, then the evaluation of the calling policy continues. If the subroutine returns FALSE to the calling policy, then the evaluation of the current term ends and the next term is evaluated.

2. The route is evaluated against the second term in the first routing policy.

If you specify a policy chain as a subroutine, the entire chain acts as a single subroutine. As with other chains, the action specified by the default policy is taken only when the entire chain does not accept or reject a route.

If a term defines multiple match conditions, including a subroutine, and a route does not match a condition specified before the subroutine, the evaluation of the term ends and the subroutine is not called and evaluated. In this situation, an action specified in the subroutine that manipulates a route's characteristics is not implemented.

Figure 17: Routing Policy Subroutine Evaluation



Related Documentation

- Default Routing Policies on page 30
- Understanding Policy Subroutines in Routing Policy Match Conditions on page 198
- Understanding How a Routing Policy Chain Is Evaluated on page 185
- Example: Configuring a Policy Subroutine on page 203

Example: Configuring a Policy Subroutine

This example demonstrates the use of a policy subroutine in a routing policy match condition.

- Requirements on page 204
- Overview on page 204

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

On Device R1, a policy called **main** is configured.

```
user@R1# show policy-options
policy-statement main {
   term subroutine-as-a-match {
      from policy subroutine;
      then accept;
   }
   term nothing-else {
      then reject;
   }
}
```

This main policy calls a subroutine called **subroutine**.

```
user@R1# show policy-options
policy-statement subroutine {
   term get-routes {
      from protocol static;
      then accept;
   }
   term nothing-else {
      then reject;
   }
}
```

The router evaluates the logic of **main** in a defined manner. The match criterion of **from policy subroutine** allows the routing device to locate the subroutine. All terms of the subroutine are evaluated, in order, following the normal policy processing rules. In this example, all static routes in the routing table match the subroutine with an action of accept. This returns a true result to the original, or calling, policy which informs the device that a positive match has occurred. The actions in the calling policy are executed and the route is accepted. All other routes in the routing table do not match the subroutine and return a false result to the calling policy. The device evaluates the second term of **main** and rejects the routes.

The actions in the subroutine do not actually accept or reject a specific route. The subroutine actions are only translated into a true or a false result. Actions that modify a route's attributes, however, are applied to the route regardless of the outcome of the subroutine.

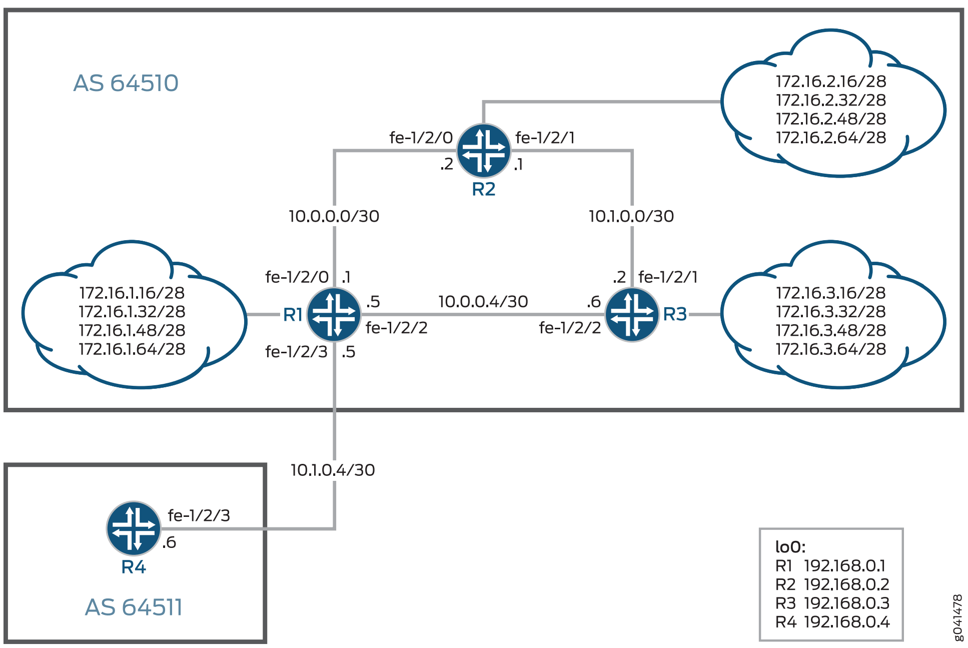
Device R1 in AS 64510 has multiple customer routes, some of which are static routes configured locally, and some of which are received from Device R2 and Device R3 through internal BGP (IBGP). AS 64510 is connected to Device R4 in AS 64511. The policy **main** is applied as an export policy in Device R1's BGP peering session with Device R4. This causes Device R1 to send only its own static routes to Device R4. Because of the policy **main**, Device R1 does not send the routes received from its internal peers, Device R2 and Device R3.

When you are working with policy subroutines, it is important to remember that the default EBGP export policy is to advertise all learned BGP routes to all EBGP peers. This default policy is in effect in the main policy and also in the subroutine. Therefore, as shown in this example, if you do not want the default EBGP export policy to take effect, you must configure a **then reject** terminating action as the final term in both the main policy and in the policy subroutine. This example demonstrates what happens when the final **then reject** term is missing either from the main policy or from the policy subroutine.

Topology

Figure 18 on page 205 shows the sample network.

Figure 18: BGP Topology for Policy Subroutine



"CLI Quick Configuration" on page 205 shows the configuration for all of the devices in Figure 18 on page 205.

The section "Step-by-Step Procedure" on page 207 describes the steps on Device R1.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 description to_R2
```

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/2 unit 0 description to_R3
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30
set interfaces fe-1/2/3 unit 0 description to_R4
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.1
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group to_64511 type external
set protocols bgp group to_64511 export main
set protocols bgp group to_64511 neighbor 10.1.0.6 peer-as 64511
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement main term subroutine-as-a-match from policy
  subroutine
set policy-options policy-statement main term subroutine-as-a-match then accept
set policy-options policy-statement main term nothing-else then reject
set policy-options policy-statement subroutine term get-routes from protocol static
set policy-options policy-statement subroutine term get-routes then accept
set policy-options policy-statement subroutine term nothing-else then reject
set routing-options static route 172.16.1.16/28 discard
set routing-options static route 172.16.1.32/28 discard
set routing-options static route 172.16.1.48/28 discard
set routing-options static route 172.16.1.64/28 discard
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device R2

```
set interfaces fe-1/2/0 unit 0 description to_R1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 description to_R3
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int neighbor 192.168.0.1 export send-static
set protocols bgp group int neighbor 192.168.0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.16.2.16/28 discard
set routing-options static route 172.16.2.32/28 discard
set routing-options static route 172.16.2.48/28 discard
set routing-options static route 172.16.2.64/28 discard
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/1 unit 0 description to_R2
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/2 unit 0 description to_R1
```



```
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.3
set protocols bgp group int neighbor 192.168.0.1 export send-static
set protocols bgp group int neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/2.6
set protocols ospf area 0.0.0.0 interface fe-1/2/0.4
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static from protocol static
set policy-options policy-statement send-static then accept
set routing-options static route 172.16.3.16/28 discard
set routing-options static route 172.16.3.32/28 discard
set routing-options static route 172.16.3.48/28 discard
set routing-options static route 172.16.3.64/28 discard
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64510
```

Device R4

```
set interfaces fe-1/2/3 unit 0 description to_R1
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.1.0.5
set routing-options autonomous-system 64511
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@R1# set fe-1/2/0 unit 0 description to_R2
   user@R1# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30

   user@R1# set fe-1/2/2 unit 0 description to_R3
   user@R1# set fe-1/2/2 unit 0 family inet address 10.0.0.5/30

   user@R1# set fe-1/2/3 unit 0 description to_R4
   user@R1# set fe-1/2/3 unit 0 family inet address 10.1.0.5/30

   user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
   ```

2. Configure the internal BGP (IBGP) connections to Device R2 and Device R3.

   ```
   [edit protocols bgp group int]
   user@R1# set type internal
   user@R1# set local-address 192.168.0.1
   user@R1# set neighbor 192.168.0.2
   user@R1# set neighbor 192.168.0.3
   ```

3. Configure the EBGP connection to Device R4.

```
[edit protocols bgp group to_64511]
user@R1# set type external
user@R1# set export main
user@R1# set neighbor 10.1.0.6 peer-as 64511
```

4. Configure OSPF connections to Device R2 and Device R3.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.0
user@R1# set interface fe-1/2/2.0
user@R1# set interface lo0.0 passive
```

5. Configure the policy **main**.

```
[edit policy-options policy-statement main term subroutine-as-a-match]
user@R1# set from policy subroutine
user@R1# set then accept

[edit policy-options policy-statement main term nothing-else]
user@R1# set then reject
```

6. Configure the policy **subroutine**.

```
[edit policy-options policy-statement subroutine term get-routes]
user@R1# set from protocol static
user@R1# set then accept

[edit policy-options policy-statement subroutine term nothing-else]
user@R1# set then reject
```

7. Configure the static route to the 172.16.5.0/24 network.

```
[edit routing-options static]
user@R1# set route 172.16.1.16/28 discard
user@R1# set route 172.16.1.32/28 discard
user@R1# set route 172.16.1.48/28 discard
user@R1# set route 172.16.1.64/28 discard
```

8. Configure the autonomous system (AS) number and router ID.

```
[edit routing-options]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 64510
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
```



```
fe-1/2/0 {
   unit 0 {
      description to_R2;
      family inet {
         address 10.0.0.1/30;
      }
   }
}
fe-1/2/2 {
   unit 0 {
      description to_R3;
      family inet {
         address 10.0.0.5/30;
      }
   }
}
fe-1/2/3 {
   unit 0 {
      description to_R4;
      family inet {
         address 10.1.0.5/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.1/32;
      }
   }
}
```

```
user@R1# show protocols
bgp {
   group int {
      type internal;
      local-address 192.168.0.1;
      neighbor 192.168.0.2;
      neighbor 192.168.0.3;
   }
   group to_64511 {
      type external;
      export main;
      neighbor 10.1.0.6 {
         peer-as 64511;
      }
   }
}
ospf {
   area 0.0.0.0 {
      interface fe-1/2/0.0;
      interface fe-1/2/2.0;
      interface lo0.0 {
         passive;
      }
   }
```

```
}
user@R1# show policy-options
policy-statement main {
  term subroutine-as-a-match {
    from policy subroutine;
    then accept;
  }
  term nothing-else {
    then reject;
  }
}
policy-statement subroutine {
  term get-routes {
    from protocol static;
    then accept;
  }
  term nothing-else {
    then reject;
  }
}
user@R1# show routing-options
static {
  route 172.6.1.16/28 discard;
  route 172.6.1.32/28 discard;
  route 172.6.1.48/28 discard;
  route 172.6.1.64/28 discard;
}
router-id 192.168.0.1;
autonomous-system 64510;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the Routes on Device R1

Purpose On Device R1, check the static routes in the routing table.

Action

```
user@R1> show route protocol static

inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.16/28     *[Static/5] 1d 02:02:13
                      Discard
172.16.1.32/28     *[Static/5] 1d 02:02:13
                      Discard
172.16.1.48/28     *[Static/5] 1d 02:02:13
                      Discard
172.16.1.64/28     *[Static/5] 1d 02:02:13
                      Discard
```

Meaning Device R1 has four static routes.

Verifying the Route Advertisement to Device R4

Purpose On Device R1, make sure that the static routes are advertised to Device R4.

Action

```
user@R1> show route advertising-protocol bgp 10.1.0.6

inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
```

Meaning As expected, Device R1 only advertises its static routes to Device R4.

Experimenting with the Default BGP Export Policy

Purpose See what can happen when you remove the final **then reject** term from the policy **main** or the policy **subroutine**.

Action

1. On Device R1, deactivate the final term in the policy **main**.

   ```
   [edit policy-options policy-statement main]
   user@R1# deactivate term nothing-else
   user@R1# commit
   ```

2. On Device R1, check to see which routes are advertised to Device R4.

   ```
   user@R1> show route advertising-protocol bgp 10.1.0.6

   inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
     Prefix                  Nexthop              MED     Lclpref    AS path
   * 172.16.1.16/28          Self                                    I
   * 172.16.1.32/28          Self                                    I
   ```

```
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```

Now, all the BGP routes from Device R1 are sent to Device R4. This is because after the processing is returned to policy **main**, the default BGP export policy takes effect.

3. On Device R1, reactivate the final term in the policy **main**, and deactivate the final term in the policy **subroutine**.

   [edit policy-options policy-statement main]
   user@R1# **activate term nothing-else**

   [edit policy-options policy-statement subroutine]
   user@R1# **deactivate term nothing-else**
   user@R1# **commit**

4. On Device R1, check to see which routes are advertised to Device R4.

```
user@R1> show route advertising-protocol bgp 10.1.0.6

inet.0: 23 destinations, 23 routes (23 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```

Now, all the BGP routes from Device R1 are sent to Device R4. This is because before the processing is returned to policy **main**, the default BGP export policy takes effect in the policy **subroutine**.

Meaning To prevent the default BGP export policy from taking effect, you must include a final **then reject** term in the main policy and in all referenced subroutines.

Related Documentation

- Understanding Policy Subroutines in Routing Policy Match Conditions on page 198
- How a Routing Policy Subroutine Is Evaluated on page 201

CHAPTER 5

Configuring Route Filters and Prefix Lists as Match Conditions

Understanding Route Filters for Use in Routing Policy Match Conditions

A *route filter* is a collection of match prefixes. When specifying a match prefix, you can specify an exact match with a particular route or a less precise match. You can configure either a common action that applies to the entire list or an action associated with each prefix.

NOTE: Because the configuration of route filters includes setting up prefixes and prefix lengths, we strongly recommend that you have a thorough understanding of IP addressing, including supernetting, before proceeding with the configuration.

It is also important to understand how a route filter is evaluated, particularly if the route filter includes multiple **route-filter** options in a **from** statement. We strongly recommend that you read "How Route Filters Are Evaluated in Routing Policy Match Conditions" on page 221 before proceeding with the configuration. Not fully understanding the evaluation process can result in faulty configuration and unexpected results.

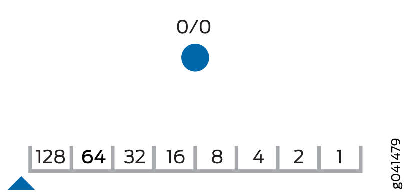
This section discusses the following topics:

Radix Trees

To understand the operation of a route filter, you need to be familiar with a device used for binary number matching known as a radix tree (sometimes called a patricia trie or radix trie). A radix tree uses binary lookups to identify IP addresses (routes). Remember that an IP address is a 32-bit number represented in a dotted decimal format for easy comprehension by humans. These 8-bit groupings can each have a value between 0 and 255. A radix tree can be a graphical representation of these binary numbers.

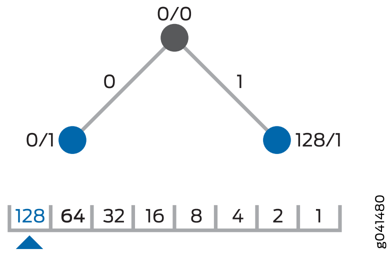
In Figure 19 on page 214, the radix tree starts with no configured value (starts at 0) and is at the leftmost position of the binary IP address. This is shown as 0/0, which is often referred to as the default route.

Figure 19: Beginning of a Radix Tree



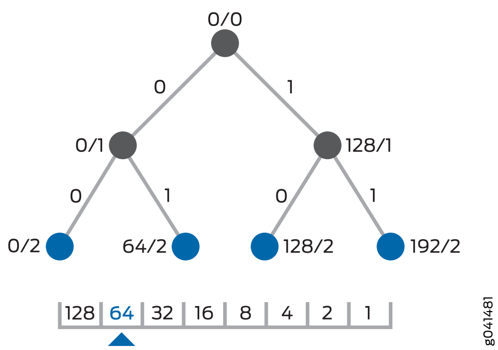
Because this is binary, each bit can have only one of two possible values—a 0 or a 1. Moving down the left branch represents a value of 0, while moving to the right represents a value of 1. The first step is shown in Figure 20 on page 215. At the first position, the first octet of the IP address has a value of 00000000 or 10000000—a 0 or 128, respectively. This is represented in Figure 20 on page 215 by the values 1/1 and 128/1.

Figure 20: First Step of a Radix Tree



The second step is shown in Figure 21 on page 215. This second level of the tree has four possible binary values for the first octet: 00000000, 01000000, 10000000, and 11000000. These decimal values of 0, 64, 128, and 192 are represented by the IP addresses of 0/2, 64/2, 128/2, and 192/2 on the radix tree.

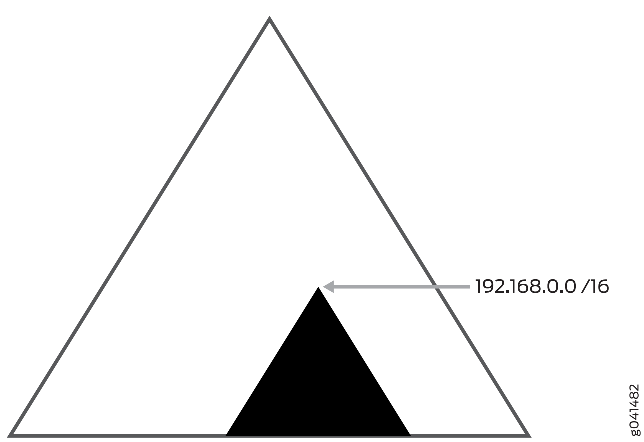
Figure 21: Second Step of a Radix Tree



This step-by-step process continues for 33 total levels to represent every possible IP address.

The radix tree structure is helpful when locating a group of routes that all share the same most significant bits. Figure 22 on page 215 shows the point in the radix tree that represents the 192.168.0.0/16 network. All of the routes that are more specific than 192.168.0.0/16 are shown in the highlighted section.

Figure 22: Locating a Group of Routes



Configuring Route Filters

NOTE: The topic, Configuring Route Filters, describes default Junos OS behavior. The walkup feature, which is not covered in this topic, alters the evaluation results discussed in this topic by allowing the router to consider shorter match conditions configured within the same term. See “Walkup for Route Filters Overview” on page 234 for details.

To configure a route filter, include one or more **route-filter** or **source-address-filter** statements:

```
[edit policy-options policy-statement policy-name term term-name from]
route-filter destination-prefix match-type {
  actions;
}
```

The **route-filter** option is typically used to match an incoming route address to destination match prefixes of any type except for unicast source addresses.

The ***destination-prefix*** address is the IP version 4 (IPv4) or IP version 6 (IPv6) address prefix specified as ***prefix/prefix-length***. If you omit ***prefix-length*** for an IPv4 prefix, the default is /32. If you omit ***prefix-length*** for an IPv6 prefix, the default is /128. Prefixes specified in a **from** statement must be either all IPv4 addresses or all IPv6 addresses.

The **source-address-filter** option is typically used to match an incoming route address to unicast source addresses in multiprotocol BGP (MBGP) and Multicast Source Discovery Protocol (MSDP) environments.

```
source-address-filter source-prefix match-type {
  actions;
}
```

source-prefix address is the IPv4 or IPv6 address prefix specified as ***prefix/prefix-length***. If you omit ***prefix-length*** for an IPv4 prefix, the default is /32***prefix-length***. If you omit ***prefix-length*** for an IPv6 prefix, the default is /128. Prefixes specified in a **from** statement must be either all IPv4 addresses or all IPv6 addresses.

match-type is the type of match to apply to the source or destination prefix. It can be one of the match types listed in Table 16 on page 217. For examples of the match types and the results when presented with various routes, see Table 17 on page 220.

actions are the actions to take if a route address matches the criteria specified for a destination match prefix (specified as part of a **route-filter** option) or for a source match prefix (specified as part of a **destination-address-filter** option). The actions can consist of one or more of the actions described in “Actions in Routing Policy Terms” on page 55.

In a route filter you can specify actions in two ways:

- In the **route-filter** or **source-address-filter** option—These actions are taken immediately after a match occurs, and the **then** statement is not evaluated.

- In the **then** statement—These actions are taken after a match occurs but no actions are specified for the **route-filter** or **source-address-filter** option.

The **upto** and **prefix-length-range** match types are similar in that both specify the most-significant bits and provide a range of prefix lengths that can match. The difference is that **upto** allows you to specify an upper limit only for the prefix length range, whereas **prefix-length-range** allows you to specify both lower and upper limits.

For more examples of these route filter match types, see "Route Filter Examples" on page 224.

Table 16: Route Filter Match Types for a Prefix List

Match Type	Match Criteria
address-mask *netmask-value*	All of the following are true: • The bit-wise logical AND of the ***netmask-value*** pattern and the incoming IPv4 or IPv6 route address and the bit-wise logical AND of the ***netmask-value*** pattern and the ***destination-prefix*** address are the same. The bits set in the ***netmask-value*** pattern do not need to be contiguous. • The ***prefix-length*** component of the incoming IPv4 or IPv6 route address and the ***prefix-length*** component of the ***destination-prefix*** address are the same. NOTE: The **address-mask** routing policy match type is valid only for matching an incoming IPv4 (**family inet**) or IPv6 (**family inet6**) route address to a list of destination match prefixes specified in a **route-filter** statement. The **address-mask** routing policy match type enables you to match an incoming IPv4 or IPv6 route address on a configured netmask address in addition to the length of a configured destination match prefix. The length of the route address must match exactly with the length of the configured destination match prefix, as the **address-mask** match type does not support prefix length variations for a range of prefix lengths. When the longest-match lookup is performed on a route filter, the lookup evaluates an **address-mask** match type differently from other routing policy match types. The lookup does not consider the length of the destination match prefix. Instead, the lookup considers the number of contiguous high-order bits set in the netmask value. For more information about this route filter match type, see "How an Address Mask Match Type Is Evaluated" on page 223. For example configurations showing route filters that contain the **address-mask** match type, see the following topics: • "Accepting Incoming IPv4 Routes by Applying an Address Mask to the Route Address and the Destination Match Prefix" on page 228. • "Accepting Incoming IPv4 Routes with Similar Patterns But Different Prefix Lengths" on page 230. • "Evaluation of an Address Mask Match Type with Longest-Match Lookup" on page 230.
exact	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The ***prefix-length*** component of the match prefix is equal to the route's prefix length.

Table 16: Route Filter Match Types for a Prefix List *(continued)*

Match Type	Match Criteria
longer	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The route's prefix length is greater than the ***prefix-length*** component of the match prefix.
orlonger	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or the ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The route's prefix length is equal to or greater than the ***prefix-length*** component of the configured match prefix.
prefix-length-range *prefix-length2-prefix-length3*	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The route's prefix length falls between ***prefix-length2*** and ***prefix-length3***, inclusive.
through {*destination-prefix2* | *source-prefix2*}	All of the following are true: • The route address shares the same most-significant bits as the first match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the first match prefix. • The route address shares the same most-significant bits as the second match prefix (***destination-prefix2*** or ***source-prefix2***). The number of significant bits is described by the ***prefix-length*** component of the second match prefix. • The route's prefix length is less than or equal to the ***prefix-length*** component of the second match prefix. You do not use the **through** match type in most routing policy configurations. For an example, see "Rejecting Routes from Specific Hosts" on page 226.
upto *prefix-length2*	All of the following are true: • The route address shares the same most-significant bits as the match prefix (***destination-prefix*** or ***source-prefix***). The number of significant bits is described by the ***prefix-length*** component of the match prefix. • The route's prefix length falls between the ***prefix-length*** component of the first match prefix and ***prefix-length2***.

Figure 23 on page 219 shows the detailed radix tree for the route 192.168.0.0/16.

Figure 23: Portion of the Radix Tree

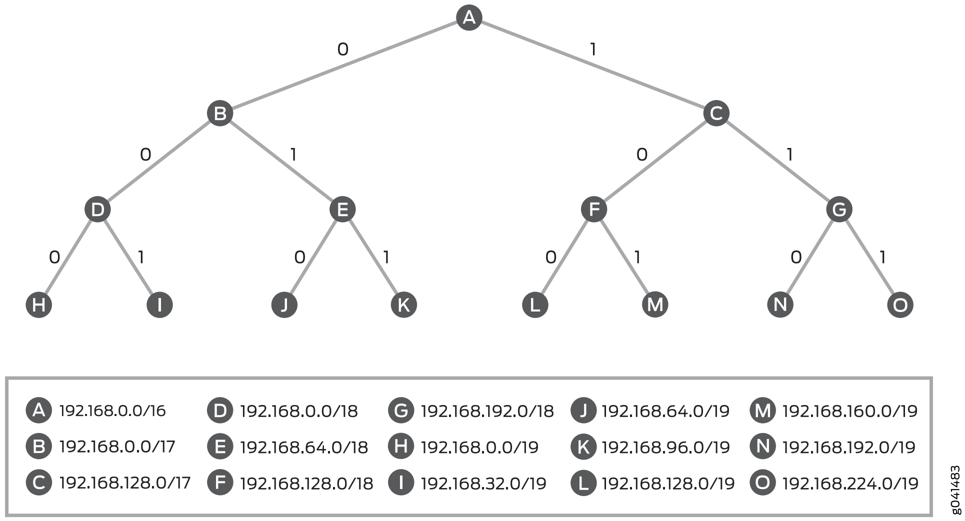


Figure 24 on page 220 and Table 17 on page 220 demonstrate the operation of the various route filter match types.

Figure 24: Route Filter Match Types

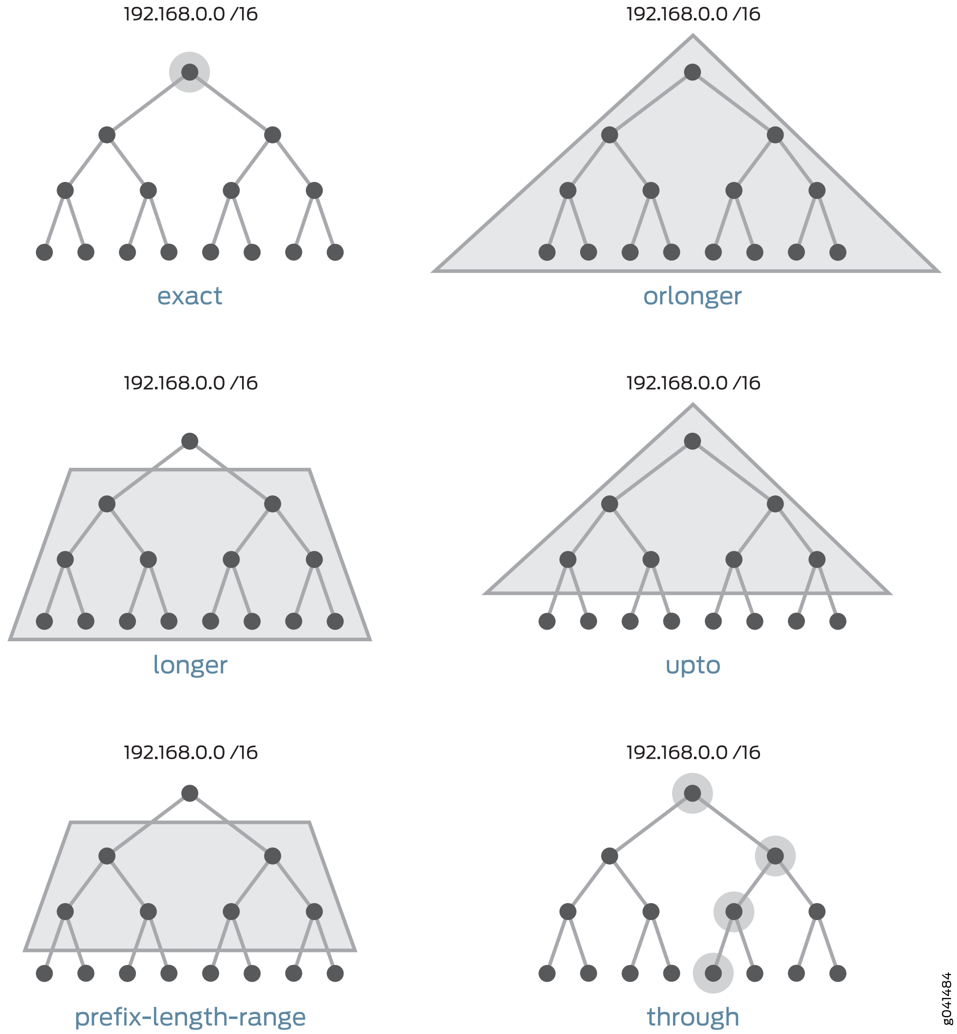


Table 17: Match Type Examples

Prefix	192.168/16 exact	192.168/16 longer	192.168/16 orlonger	192.168/16 upto /24	192.168/16 prefix-length-range /18-/20	192.168/16 through 192.168.16/20	192.168/19 address-mask 255.255.0.0
10.0.0.0/8	–	–	–	–	–	–	–
192.168.0.0/16	Match	–	Match	Match	–	Match	–
192.168.0.0/17	–	Match	Match	Match	–	Match	–
192.168.0.0/18	–	Match	Match	Match	Match	Match	–
192.168.0.0/19	–	Match	Match	Match	Match	Match	Match
192.168.4.0/24	–	Match	Match	Match	–	–	–

Table 17: Match Type Examples *(continued)*

Prefix	192.168/16 exact	192.168/16 longer	192.168/16 orlonger	192.168/16 upto /24	192.168/16 prefix-length-range /18–/20	192.168/16 through 192.168.16/20	192.168/19 address-mask 255.255.0.0
192.168.5.4/30	–	Match	Match	–	–	–	–
192.168.12.4/30	–	Match	Match	–	–	–	–
192.168.12.128/32	–	Match	Match	–	–	–	–
192.168.16.0/20	–	Match	Match	Match	Match	Match	–
192.168.192.0/18	–	Match	Match	Match	Match	–	–
192.168.224.0/19	–	Match	Match	Match	Match	–	Match
10.169.1.0/24	–	–	–	–	–	–	–
10.170.0.0/16	–	–	–	–	–	–	–

How Route Filters Are Evaluated in Routing Policy Match Conditions

During route filter evaluation, the policy framework software compares each route's source address with the destination prefixes in the route filter. The evaluation occurs in two steps:

1. The policy framework software performs a *longest-match lookup*, which means that the software searches for the prefix in the list with the longest length.

   The longest-match lookup considers the ***prefix*** and ***prefix-length*** components of the configured match prefix only, and not the ***match-type*** component. The following sample route filter illustrates this point:

   ```
   from {
      route-filter 192.168.0.0/14 upto /24 reject;
      route-filter 192.168.0.0/15 exact;
   }
   then accept;
   ```

   The longest match for the candidate route 192.168.1.0/24 is the second route-filter, 192.168.0.0/15, which is based on prefix and prefix length only.

2. When an incoming route matches a prefix (longest first), the following actions occur:

   1. The route filter stops evaluating other prefixes, even if the match type fails.
   2. The software examines the match type and action associated with that prefix.

NOTE: When a route source address is evaluated against a match criteria that uses the **address-mask** match type, both steps of the evaluation include the configured netmask value. For more information, see "How an Address Mask Match Type Is Evaluated" on page 223.

In Step 1, if route 192.168.1.0/24 were evaluated, it would fail to match. It matches the longest prefix of 192.168.0.0/15, but it does not match **exact**. The route filter is finished because it matched a prefix, but the result is a failed match because the match type failed.

If a match occurs, the action specified with the prefix is taken. If an action is not specified with the prefix, the action in the **then** statement is taken. If neither action is specified, the software evaluates the next term or routing policy, if present, or takes the **accept** or **reject** action specified by the default policy. For more information about the default routing policies, see "Default Routing Policies" on page 30.

NOTE: If you specify multiple prefixes in the route filter, only one prefix needs to match for a match to occur. The route filter matching is effectively a logical OR operation.

If a match does not occur, the software evaluates the next term or routing policy, if present, or takes the **accept** or **reject** action specified by the default policy.

For example, compare the prefix 192.168.254.0/24 against the following route filter:

```
route-filter 192.168.0.0/16 orlonger;
route-filter 192.168.254.0/23 exact;
```

The prefix 192.168.254.0/23 is determined to be the longest prefix. When the software evaluates 192.168.254.0/24 against the longest prefix, a match occurs (192.168.254.0/24 is a subset of 192.168.254.0/23). Because of the match between 192.168.254.0/24 and the longest prefix, the evaluation continues. However, when the software evaluates the match type, a match does not occur between 192.168.254.0/24 and 192.168.254.0/23 exact. The software concludes that the term does not match and goes on to the next term or routing policy, if present, or takes the **accept** or **reject** action specified by the default policy.

NOTE: The walkup feature allows terms with multiple route filters to "walk-up" the evaluation process to include less-specific routes as well as the longest match. In other words, enabling walkup changes the default behavior from "if one fails, then the term fails" to "if one matches, then the term matches." For more information about the walkup feature, see "Walkup for Route Filters Overview" on page 234.

How Prefix Order Affects Route Filter Evaluation

The order in which the prefixes are specified (from top to bottom) typically does not matter, because the policy framework software scans the route filter looking for the longest prefix during evaluation. An exception to this rule is when you use the same destination prefix multiple times in a list. In this case, the order of the prefixes is important, because the list of identical prefixes is scanned from top to bottom, and the first match type that matches the route applies.

NOTE: The walkup feature allows terms with multiple route filters to "walk-up" the evaluation process to include less-specific routes as well as the longest match. In other words, enabling walkup changes the default behavior from "if one fails, then the term fails" to "if one matches, then the term matches." For more information about the walkup feature, see "Walkup for Route Filters Overview" on page 234.

In the following example, different match types are specified for the same prefix. The route 0.0.0.0/0 would be rejected, the route 0.0.0.0/8 would be marked with **next-hop self**, and the route 0.0.0.0/25 would be rejected.

```
route-filter 0.0.0.0/0 upto /7 reject;
route-filter 0.0.0.0/0 upto /24 next-hop self;
route-filter 0.0.0.0/0 orlonger reject;
```

How an Address Mask Match Type Is Evaluated

The **address-mask** routing policy match type enables you to match incoming IPv4 or IPv6 route addresses on a configured netmask value in addition to the length of a configured destination match prefix. During route filter evaluation, an **address-mask** match type is processed differently from other routing policy match types, taking into consideration the configured netmask value:

- When a longest-match lookup evaluates an **address-mask** routing policy match type, the ***prefix-length*** component of the configured match prefix is not considered. Instead, the lookup considers the number of contiguous high-order bits set in the configured netmask value.
- When an incoming IPv4 or IPv6 route address is evaluated against a route filter match criteria that uses the **address-mask** routing policy match type, the match succeeds if the following values are identical:
  - The bit-wise logical AND of the configured netmask value and the incoming IPv4 or IPv6 route address
  - The bit-wise logical AND of the configured netmask value and the configured destination match prefix

For an example configuration of a route filter that contains two **address-mask** match types, see "Evaluation of an Address Mask Match Type with Longest-Match Lookup" on page 230.

Common Configuration Problem with the Longest-Match Lookup

A common problem when defining a route filter is including a shorter prefix that you want to match with a longer, similar prefix in the same list. For example, imagine that the prefix 192.168.254.0/24 is compared against the following route filter:

```
route-filter 192.168.0.0/16 orlonger;
route-filter 192.168.254.0/23 exact;
```

Because the policy framework software performs longest-match lookup, the prefix 192.168.254.0/23 is determined to be the longest prefix. An exact match does not occur between 192.168.254.0/24 and 192.168.254.0/23 exact. The software determines that the term does not match and goes on to the next term or routing policy, if present, or takes the **accept** or **reject** action specified by the default policy. (For more information about the default routing policies, see "Default Routing Policies" on page 30.) The shorter prefix 192.168.0.0/16 orlonger that you wanted to match is inadvertently ignored.

One solution to this problem is to remove the prefix 192.168.0.0/16 orlonger from the route filter in this term and move it to another term where it is the only prefix or the longest prefix in the list.

Another solution is to enable the **walkup** feature. See "Walkup for Route Filters Overview" on page 234 for details.

Route Filter Examples

The examples in this section show only fragments of routing policies. Normally, you would combine these fragments with other terms or routing policies.

In all examples, remember that the following actions apply to nonmatching routes:

- Evaluate next term, if present.
- Evaluate next policy, if present.
- Take the **accept** or **reject** action specified by the default policy. For more information about the default routing policies, see "Default Routing Policies" on page 30.

The following examples show how to configure route filters for various purposes:

Rejecting Routes with Specific Destination Prefixes and Mask Lengths

Reject routes with a destination prefix of 0.0.0.0 and a mask length from 0 through 8, and accept all other routes:

```
[edit]
policy-options {
  policy-statement policy-statement from-hall2 {
    term 1 {
      from {
        route-filter 0.0.0.0/0 upto /8 reject;
      }
    }
    then accept;
  }
}
```

Rejecting Routes with a Mask Length Greater than Eight

Reject routes with a mask of /8 and greater (that is, /8, /9, /10, and so on) that have the first 8 bits set to 0 and accept routes less than 8 bits in length:

```
[edit]
policy-options {
  policy-statement from-hall3 {
    term term1 {
      from {
        route-filter 0/0 upto /7 accept;
        route-filter 0/8 orlonger;
      }
      then reject;
    }
  }
}
```

Rejecting Routes with Mask Length Between 26 and 29

Reject routes with the destination prefix of 192.168.10/24 and a mask between /26 and /29 and accept all other routes:

```
[edit]
policy-options {
  policy-statement from-customer-a {
    term term1 {
      from {
        route-filter 192.168.10/24 prefix-length-range /26–/29 reject;
      }
      then accept;
    }
```



```
    }
}
```

Rejecting Routes from Specific Hosts

Reject a range of routes from specific hosts, and accept all other routes:

```
[edit]
policy-options {
    policy-statement hosts-only {
        from {
            route-filter 10.125.0.0/16 upto /31 reject;
            route-filter 0/0;
        }
        then accept;
    }
}
```

You do not use the **through** match type in most routing policy configurations. You should think of **through** as a tool to group a contiguous set of exact matches. For example, instead of specifying four exact matches:

```
from route-filter 0.0.0.0/1 exact
from route-filter 0.0.0.0/2 exact
from route-filter 0.0.0.0/3 exact
from route-filter 0.0.0.0/4 exact
```

You could represent them with the following single match:

```
from route-filter 0.0.0.0/1 through 0.0.0.0/4
```

Accepting Routes with a Defined Set of Prefixes

Explicitly accept a limited set of prefixes (in the first term) and reject all others (in the second term):

```
policy-options {
policy-statement internet-in {
    term 1 {
        from {
            route-filter 192.168.231.0/24 exact accept;
            route-filter 192.168.244.0/24 exact accept;
            route-filter 192.168.198.0/24 exact accept;
            route-filter 192.168.160.0/24 exact accept;
            route-filter 192.168.59.0/24 exact accept;
        }
    }
    term 2 {
        then {
            reject;
        }
    }
}
```

Rejecting Routes with a Defined Set of Prefixes

Reject a few groups of prefixes, and accept the remaining prefixes:


```
[edit policy-options]
policy-statement drop-routes {
  term 1{
    from { # first, reject a number of prefixes:
      route-filter default exact reject; # reject 0.0.0.0/0 exact
      route-filter 0.0.0.0/8 orlonger reject; # reject prefix 0, mask /8 or longer
      route-filter 10.0.0.0/8 orlonger reject; # reject loopback addresses
      }
      route-filter 10.105.0.0/16 exact { # accept 10.105.0.0/16
        as-path-prepend "1 2 3";
        accept;
      }
      route-filter 192.0.2.0/24 orlonger reject; # reject test network packets
      route-filter 172.16.233.0/3 orlonger reject; # reject multicast and higher
      route-filter 0.0.0.0/0 upto /24 accept; # accept everything up to /24
      route-filter 0.0.0.0/0 orlonger accept; # accept everything else
      }
    }
  }
}
```

Rejecting Routes with Prefixes Longer than 24 Bits

Reject all prefixes longer than 24 bits. You would install this routing policy in a sequence of routing policies in an **export** statement. The first term in this filter passes on all routes with a prefix length of up to 24 bits. The second, unnamed term rejects everything else.

```
[edit policy-options]
policy-statement 24bit-filter {
  term acl20 {
    from {
      route-filter 0.0.0.0/0 upto /24;
    }
    then next policy;
  }
  then reject;
}
```

If, in this example, you were to specify **route-filter 0.0.0.0/0 upto /24 accept**, matching prefixes would be accepted immediately and the next routing policy in the **export** statement would never get evaluated.

If you were to include the **then reject** statement in the term **acl20**, prefixes greater than 24 bits would never get rejected because the policy framework software, when evaluating the term, would move on to evaluating the next statement before reaching the **then reject** statement.

Rejecting PIM Multicast Traffic Joins

Configure a routing policy for rejecting Protocol Independent Multicast (PIM) multicast traffic joins for a source destination prefix from a neighbor:

```
[edit]
policy-options {
  policy-statement join-filter {
    from {
```

```
            neighbor 10.14.12.20;
            source-address-filter 10.83.0.0/16 orlonger;
        }
        then reject;
    }
}
```

Rejecting PIM Traffic

Configure a routing policy for rejecting PIM traffic for a source destination prefix from an interface:

```
[edit]
policy-options {
    policy-statement join-filter {
        from {
            interface so-1/0/0.0;
            source-address-filter 10.83.0.0/16 orlonger;
        }
        then reject;
    }
}
```

The following routing policy qualifiers apply to PIM:

- **interface**—Interface over which a join is received
- **neighbor**—Source from which a join originates
- **route-filter**—Group address
- **source-address-filter**—Source address for which to reject a join

For more information about importing a PIM join filter in a PIM protocol definition, see the *Multicast Protocols Feature Guide*.

Accepting Incoming IPv4 Routes by Applying an Address Mask to the Route Address and the Destination Match Prefix

Accept incoming IPv4 routes with a destination prefix of 10.1.0.0/24 and the third byte an even number from 0 to 14, inclusive:

```
[edit]
policy-options {
    policy-statement from_customer_a {
        term term_1 {
            from {
                route-filter 10.1.0.0/24 address-mask 255.255.241.0;
            }
            then {
                ...
                reject;
            }
        }
    }
}
```


The route filter in routing policy term **term_1** matches the following incoming IPv4 route addresses:

- 10.1.0.0/24
- 10.1.2.0/24
- 10.1.4.0/24
- 10.1.6.0/24
- 10.1.8.0/24
- 10.1.10.0/24
- 10.1.12.0/24
- 10.1.14.0/24

The bit-wise logical AND of the netmask value and the candidate route address must match the bit-wise logical AND of the netmask value and the match prefix address. That is, where the netmask bit pattern 255.255.241.0 contains a set bit, the incoming IPv4 route address being evaluated must match the value of the corresponding bit in the destination prefix address 10.1.0.0/24.

- The first two bytes of the netmask value are binary 1111 1111   1111 1111, which means that a candidate route address will fail the match if the first two bytes are not 10.1.
- The third byte of the netmask value is binary 1111 0001, which means that a candidate route address will fail the match if the third byte is greater than 15 (decimal), an odd number, or both.
- The prefix length of the match prefix address is 24 (decimal), which means that a candidate route address will fail the match if its prefix length is not exactly 24.

As an example, suppose that the candidate route address being tested in the policy is 10.1.8.0/24 (binary 0000 1010   0000 0001   0000 1000).

- When the netmask value is applied to this candidate route address, the result is binary 0000 1010   0000 0001   0000 0000.
- When the netmask value is applied to the configured destination prefix address, the result is also binary 0000 1010   0000 0001   0000 0000.
- Because the results of both AND operations are the same, the match continues to the second match criteria.
- Because the prefix lengths of the candidate address and the configured destination prefix address are the same (24 bits), the match succeeds.

As another example, suppose that the candidate route address being tested in the policy is 10.1.3.0/24 (binary 0000 1010   0000 0001   0000 0011).

- When the netmask value is applied to this candidate route address, the result is binary 0000 1010   0000 0001   0000 0001.

- However, when the netmask value is applied to the configured destination prefix address, the result is binary 0000 1010   0000 0001   0000 0000.
- Because the results of the two AND operations are different (in the third byte), the match fails.

Accepting Incoming IPv4 Routes with Similar Patterns But Different Prefix Lengths

Accept incoming IPv4 route addresses of the form 10.*.1/24 or 10.*.1.*/32:

```
[edit]
policy-options {
  policy-statement from_customer_b {
    term term_2 {
      from {
        route-filter 10.0.1.0/24 address-mask 255.0.255.0;
        route-filter 10.0.1.0/32 address-mask 255.0.255.0;
      }
      then {
        ...
        reject;
      }
    }
  }
}
```

The route filter match criteria **10.0.1.0/24 address-mask 255.0.255.0** matches an incoming IPv4 route address of the form 10.*.1/24. The route's prefix length must be exactly 24 bits long, and any value is acceptable in the second byte.

The route filter match criteria **10.0.1.0/32 address-mask 255.0.255.0** matches an incoming IPv4 route address of the form 10.*.1.*/32. The route's prefix length must be exactly 32 bits long, and any value is acceptable in the second byte and the fourth byte.

Evaluation of an Address Mask Match Type with Longest-Match Lookup

This example illustrates how a longest-match lookup evaluates a route filter that contains two **address-mask** match types. Consider the route filter configured in the routing policy term **term_3** below:

```
[edit]
policy-options {
  policy-statement from_customer_c {
    term term_3 {
      from {
        route-filter 10.0.1.0/24 address-mask 255.0.255.0;
        route-filter 10.0.2.0/24 address-mask 255.240.255.0;
      }
      then {
        ...
      }
    }
  }
}
```


Suppose that the incoming IPv4 route source address 10.1.1.0/24 is tested against the route filter configured in the policy term **term_3**:

1. The longest-match lookup tree for routing policy term **term_3** contains two match prefixes: one prefix for **10.0.1.0/24 address-mask 255.0.255.0** and one prefix for **10.0.2.0/24 address-mask 255.240.255.0**. When searching the tree for the longest-prefix match for a candidate, the longest-match lookup considers the number of contiguous high-order bits in the configured ***netmask-value*** instead of the length of the configured ***destination-prefix***:

   - For the first route filter match criteria, the longest-match lookup entry is 10.0.0.0/8 because the netmask value contains 8 contiguous high-order bits.
   - For second route filter match criteria, the longest-match lookup entry is 10.0.0.0/12 because the netmask value contains 12 contiguous high-order bits.

   For the candidate route address 10.1.1.0/24, the longest-match lookup returns the tree entry 10.0.0.0/12, which is corresponds to the route filter match criteria **10.0.2.0/24 address-mask 255.240.255.0**.

2. Now that the longest-match prefix in **term_3** has been identified for the candidate route address, the candidate route address is evaluated against the route filter match criteria **10.0.2.0/24 address-mask 255.240.255.0**:

   a. To test the incoming IPv4 route address 10.1.1.0/24, the netmask value 255.240.255.0 is applied to 10.1.1.0/24. The result is 10.0.1.0.

   b. To test the configured destination prefix address 10.0.2.0/24, the netmask value 255.240.255.0 is applied to 10.0.2.0/24. The result is 10.0.2.0.

   c. Because the results are different, the route filter match fails. No actions, whether specified with the match criteria or with the **then** statement, are taken. The incoming IPv4 route address is not evaluated against any other match criteria.

Related Documentation

- Walkup for Route Filters Overview on page 234
- Example: Configuring Policy Chains and Route Filters on page 186
- Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF on page 258
- Example: Configuring the MED Using Route Filters on page 262

Understanding Route Filter and Source Address Filter Lists for Use in Routing Policy Match Conditions

Existing route filters and source address filters are configured and processed inline within the term of the policy statement. When route policies are changed, the entire policy is purged and rebuilt during the configuration parsing operation. When this happens on routing policies that include hundreds or even thousands of route filters and source address filters, a significant amount of time is added to the rebuild of the policy.

In order to speed the parsing operation, the **route-filter-list** and **source-address-filter-list** statements are available as another means of configuring route filters and source address filters. These statements maintain all the capabilities of the **route-filter** and **source-address-filter** statements, including consideration of the prefix length and match type of the individual prefixes in the list.

Route filters and route filter lists are typically used to match an incoming route address to destination match prefixes of any type except for unicast source addresses.

Source address filters and source address filter lists are typically used to match an incoming route address to unicast source addresses in Multiprotocol BGP (MBGP) and Multicast Source Discovery Protocol (MSDP) environments.

Multiple route filter lists and source address filter lists can be used within the same policy statements. Route filter lists and source address filter lists can also be used in conjunction with route filters and source address filters.

Related Documentation

- route-filter-list on page 1236
- Understanding Route Filters for Use in Routing Policy Match Conditions on page 213

Understanding Load Balancing Using Source or Destination IP Only

In deep packet inspection (DPI) networks with per-subscriber awareness or transparent caches, all of the PE routers in the service provider network should route all traffic to and from a particular subscriber through the specific content server that maintains subscriber state for that subscriber. To reach the same server consistently, the traffic must be hashed onto the same link towards that specific server for traffic in both directions.

In order to accomplish this consistency, certain MX Series routers can be configured to make load-balancing decisions based solely on the source IP address or the destination IP address of the traffic. From a service provider perspective, using only the source IP for inbound traffic, and the destination IP for outbound traffic limits the criteria used in hashing, making it more likely that a particular link will be chosen to forward the traffic.

NOTE: This feature will only work on IP-based traffic. In the case of L3VPN traffic, only MPLS lookup will be performed on the PE routers when the default label assignment scheme is used. In order to use source-or-destination only load-balancing with L3VPN, you can either configure **vrf-table-label** or add a vt- interface in the routing instance.

Related Documentation

- Configuring Load Balancing Using Source or Destination IP Only on page 233

- *vrf-table-label*
- *interface*

Configuring Load Balancing Using Source or Destination IP Only

In equal-cost multipath, (ECMP) per-subscriber aware environments such as content service providers who service residential customers, traffic in both directions within the service provider network should always pass through the content servers that maintain the subscriber state information for a given subscriber. This is accomplished by calculating the load balancing hash based solely on source address for traffic coming into the service provider network and calculating the load balancing hash based solely on the destination address for traffic leaving the service provider network.

Source and destination only load balancing is generally configured in an ECMP or aggregated ethernet (AE) environment on an service provider network. It is usually applied to all of the PE routers. It is only supported for IPv4 (**inet**) and IPv6 (**inet6**) traffic.

You do not need any special configuration in place before starting this configuration.

NOTE: This feature will only work on IP-based traffic. In the case of L3VPN traffic, only MPLS lookup will be performed on the PE routers when the default label assignment scheme is used. In order to use source-or-destination only load-balancing with L3VPN, you can either configure **vrf-table-label** or add a vt- interface in the routing instance.

To configure load balancing using source or destination IP only, you first configure system-wide forwarding options with a prefix-length to use when calculating the hash-key. Then, you configure a policy action of either **load-balance source-ip-only** or **load-balance destination-ip-only** within a policy statement.

1. To configure system-wide prefix length for use with source and destination IP only load balancing, insert the **source-destination-only-load-balancing** configuration statement at the [**edit forwarding-options enhanced-hash-key**] hierarchy level and add a prefix length:

```
[edit forwarding-options enhanced-hash-key]
source-destination-only-load-balancing {
  family inet {
    prefix-length prefix-length;
  }
  family inet6 {
    prefix-length prefix-length;
  }
}
```

2. To configure routing policy to use load balancing based on source or destination IP only, insert either the **source-ip-only** or **destination-ip-only** as an action statement within a policy statement at the [**edit policy-options policy-statement *policy-name***] hierarchy level:

```
[edit policy-options policy-statement policy-name]
term term-name {
```

```
    from {
      route-filter filter-spec’
    }
    then {
      load-balance (source-ip-only | destination-ip-only);
    }
  }
```

NOTE: The **source-ip-only** and **destination-ip-only** configuration elements cannot be used together in the same term. This is because of the directional nature of the traffic that we are load balancing. To use the two elements in the same policy statement, you create two separate terms, each using a route filter specification that addresses the same traffic. Then use **source-ip-only** for the inbound traffic and **destination-ip-only** for the outbound traffic.

NOTE:

Related Documentation

- *Configuring VPLS Load Balancing on MX Series 3D Universal Edge Routers*
- Understanding Load Balancing Using Source or Destination IP Only on page 232
- *Configuring Stateful Load Balancing on Aggregated Ethernet Interfaces*

Walkup for Route Filters Overview

Use the walkup feature if you have concerns about policy performance because of split route filters across multiple policy terms. The walkup feature enables the consolidation of route filters under one policy term.

By default, Junos evaluates multiple route filters in a policy statement term by first finding the longest match prefix and then evaluating the conditions attached to the route filter, such as prefix range. If the route filter condition is false (for example, the prefix is not in the specified range), then the whole term is false, even if there are potentially true shorter route filter prefixes. Due to this behavior, there can be performance issues if route filters are split into individual policy statement terms. The walkup feature changes the default route filter behavior.

Some automated policy tools — for example, those used for autonomous system border routers in the Border Gateway Protocol (BGP) — break up route filters into multiple terms because of the default route filter behavior. Route filters are also used in routing protocols other than BGP; the walkup feature is not limited to BGP route filters.

NOTE: **Technically, BGP does not deal with routes in the same way as OSPF or IS-IS. BGP "routes" are more properly called network layer reachability information (NLRI) updates. However, the term "route" is used in most documentation and is used here.**

Route filters consist of three major parts:

1. A prefix and prefix length (for example, **10.0.0.0/8**)
2. A match condition (for example, **exact**)
3. An action that is carried out if both previous parts — the prefix and match condition — both evaluate to true (for example, **accept**)

So the **10.0.0.0/8 exact accept** route filter succeeds if and only if the prefix considered is **10.0.0.0/8** exactly. This route filter rejects routes with all other longer prefixes, such as **10.0.0.0/10**, although there might be other route filter terms in the policy chain that accept the **10.0.0.0/10** route.

NOTE: **Although the 10.0.0.0/8 route and variations are not specifically reserved for documentation, the private RFC 1918 10.0.0.0/8 address space is used in this topic because of the flexibility and realistic scenarios that this address spaces provides.**

Route filters can be combined in a single policy statement term. In that case, evaluation becomes more complex. Consider the following routing policy:

```
[edit policy-options]
policy-statement RouteFilter-A {
  term RouteFilter-1 {
    from {
      route-filter 10.0.0.0/16 prefix-length-range /22-/24;
      route-filter 10.0.0.0/8 orlonger;
    }
    then accept;
  }
  term default {
    then reject;
  }
}
```

Note that the **10.0.0.0/8 orlonger** filter includes the **10.0.0.0/16 prefix-length-range /22-/24** filter in its scope. That is, any **10.0.0.0** route with a prefix of 8 bits or longer could also be a route with a prefix in the range between 22 and 24 bits.

By default, evaluation of a policy statement term with multiple route filters is a two-step process:

1. The policy framework software performs a longest-match lookup on the list based on prefix and prefix-length values.

2. The software considers the route filter condition (**orlonger**, **exact**, and so on). The route either fulfills the route filter condition (success) or does not match the route filter condition (failure).

Based on the results of these two steps, the action determined by the match or failure is applied to the route. In **Route-Filter-A**, this means that any route that is “true” is accepted and any route that is “false” in the **RouteFilter-1** term is rejected. This route becomes a hidden (filtered) route.

For example, consider what happens when the route **10.0.0.0/18** is evaluated by the policy statement **RouteFilter-A**:

First, the **10.0.0.0/18** route is evaluated by the **RouteFilter-1** term. Because **10.0.0.0/16** is longer than **10.0.0.0/8**, the **10.0.0.0/18** route matches the longer and more specific route prefix. Next, the match fails because the **10.0.0.0/18** route does not match the **prefix-length-range /22-/24** condition. So the route match fails in the **RouteFilter-1** term, and the policy examines the next term, the default term. The **10.0.0.0/18** route is rejected by the default term.

As a result, the **10.0.0.0/18** route is hidden (filtered). (The **10.0.0.0/18** route can still be found with the **show route hidden** command.)

The issue is that the user might actually want the **10.0.0.0/18** route to be accepted, not rejected. Naturally, a route filter with a **10.0.0.0/18 exact** configuration could be added. But in a backbone routing table with 100,000 or more entries, it is not possible to configure a route filter tuned to every possible route or every possible new route added to the network.

The default workaround to achieve the proper behavior from the example routing policy is to configure a separate term for each route filter. This is frequently done, as follows:

```
[edit policy-options]
policy-statement RouteFilter-A {
   term RouteFilter-1 {
      from {
         route-filter 10.0.0.0/16 prefix-length-range /22-/24;
      }
      then accept;
   }
   term RouteFilter-2 {
      from {
         route-filter 10.0.0.0/8 orlonger;
      }
      then accept;
   }
   term default {
      then reject;
   }
}
```

Now the **10.0.0.0/18** route is accepted because, although it still fails the **RouteFilter-1** match condition, it matches the new **RouteFilter-2** term (**10.0.0.0/8** is the longest match, and the **orlonger** condition is true). The problem with this approach is that the complete routing policy now takes more time to evaluate than when multiple route filters are grouped. This method also makes maintenance more complex.

The issues with the one-term-per-route-filters approach are solved with the walkup statement and feature. Walkup alters the default behavior of route filter evaluation globally or on a per-policy basis.

The walkup feature allows terms with multiple route filters to “walk-up” the evaluation process to include less-specific routes as well as the longest match. In other words, the walkup knob changes the default behavior from “if one fails, then the term fails” to if “one matches, then the term matches.”

Consider the application of the walkup feature to the example policy statement (you can also apply walk-up globally to all policies configured):

```
[edit policy-options]
policy-statement RouteFilter-A {
  defaults {
    route-filter walkup;
  }
  term RouteFilter-1 {
    from {
      route-filter 10.0.0.0/16 prefix-length-range /22-/24;
      route-filter 10.0.0.0/8 orlonger;
    }
    then accept;
  }
  term default {
    then reject;
  }
}
```

This is what happens when the route prefix **10.0.0.0/18** is evaluated by the policy statement **RouteFilter-A**:

The default behavior is altered by the walkup knob. As before, the **10.0.0.0/18** route matches the longer and more specific route prefix because **10.0.0.0/16** is longer than **10.0.0.0/8**. As before, this match fails because the **10.0.0.0/18** route does not match the **prefix-length-range /22-/24** condition. However, this time the process continues by a “walk up” and examines the less specific **10.0.0.0/8** route filter. The route condition of **orlonger** matches this filter and therefore the route is accepted by the **RouteFilter-1** term.

This can be verified (for a BGP route) by the **show route protocol bgp 10.0.0.0/18** command. This time, the route is not hidden.

If you enable the walkup feature globally, you can override it locally on a per-policy basis with the **[edit policy-options policy-statements policy-statement-name defaults route-filter no-walkup]** statement.

Related Documentation

- Example: Configuring Walkup for Route Filters Globally to Improve Operational Efficiency on page 247

- Example: Configuring Walkup for Route Filters Locally to Improve Operational Efficiency on page 252
- Configuring Walkup for Route Filters to Improve Operational Efficiency on page 238
- Route Filter Match Conditions on page 53

- *BGP Configuration Overview*
- *Verify That a Particular BGP Route Is Received on Your Router*
- *Example: Configuring BGP Route Advertisement*

Configuring Walkup for Route Filters to Improve Operational Efficiency

Use the walkup feature if you have concerns about policy performance because of split route filters across multiple policy terms. The walkup feature enables the consolidation of route filters under one policy term.

If policy statements have been split into multiple terms because of the default route filter behavior, the route filter walkup feature allows you to consolidate multiple route filters into one policy statement term. By default, Junos OS evaluates multiple route filters in a policy statement term by first finding the longest match prefix and then evaluating the conditions attached to the route filter, such as the prefix range. If the route filter condition is false (for example, the prefix is not in the specified range), then the whole term is false, even if there are potentially true shorter route filter prefixes. The walkup feature alters this default behavior, locally or globally.

The route filter walkup feature is used anywhere multiple route filters are used in a policy statement. The walkup option is supported in the main routing instance at the **[edit policy-options]** hierarchy level and in logical systems at the **[edit logical-systems policy-options]** hierarchy level.

Before you begin configuring route filter walkup, be sure you have:

- A properly configured routing policy or set of routing policies
- A need to consolidate multiple route filter terms into fewer routing policy terms

Route filter walkup can be configured in two different ways. You can configure the **walkup** option globally at the **[edit policy-options default route-filter]** hierarchy level or in logical systems at the **[edit logical-systems policy-options default route-filter]** hierarchy level. When you configure the **walkup** option globally, you alter the policy route filter behavior in every policy statement. Instead of the default policy statement behavior (if the longest match route filter is false, then the term is false), the **walkup** option changes this behavior globally (to “walk up” from the longest match route filter to less specific, and if any is true, then the term is true).

If you configure the **walkup** option globally, you can still override it locally on a per-routing-policy basis. So if you have enabled **walkup** globally, you can override it in a routing policy by configuring the **no-walkup** option statement at the **[edit policy-options**

policy-statement default route-filter] hierarchy level. The **no-walkup** option restores the default route filter behavior locally for this policy statement.

NOTE: At the **[edit policy-options default route-filter]** global level, the only option is the **walkup** statement because the default behavior globally is "no walkup." However, for an individual policy statement at the **[edit policy-options policy-statement default route-filter]** hierarchy level, you can configure either the **walkup** or **no-walkup** option statement. In this way, at the local level, you can control whether the policy statement performs a walkup (with the **walkup** statement configured) or no walkup (with the **no-walkup** statement configured. This gives the user maximum control over the **walkup** option

You configure the walkup feature globally with:

user@host> set policy-options defaults route-filter walkup

Alternatively, configure the walkup feature globally in a logical system with:

user@host> set logical-systems *logical-system-name* policy-options defaults route-filter walkup

You configure the walkup or no-walkup feature locally in a policy statement with:

user@host> set policy-options policy-statement *policy-statement-name* defaults route-filter [no-walkup | walkup]

Alternatively, configure the walkup feature locally in a logical system with:

user@host> set logical-systems *logical-system-name* policy-options policy-statement policy-statement-name defaults route-filter [no-walkup | walkup]

Route filter walkup behavior can be complex when the statements are configured at the global and local level at the same time. Table 18 on page 239 shows the behavior of a policy statement with all six possible combinations of the walkup option when you configure the feature both globally and locally.

Table 18: Route Filter Walkup and Policy Statements

Case:	Global Configuration	Local Configuration	Result
1	(none)	(none)	The device does not perform a walkup for any policy (default operation).
2	(none)	**walkup**	The device performs a walkup for this policy.
3	(none)	**no-walkup**	The device does not perform a walkup for any policy (default operation).
4	**walkup**	(none)	The device performs a walkup for all policies.

Table 18: Route Filter Walkup and Policy Statements *(continued)*

Case:	Global Configuration	Local Configuration	Result
5	**walkup**	**walkup**	The device performs a walkup for all policies.
6	**walkup**	**no-walkup**	The device does not perform a walkup for this policy only.

Each row forms a possible use case numbered 1 through 6. Each walkup case is configured as follows:

- Case #1: This is a trivial configuration for backward compatibility. No route filter walkup is enabled either globally or locally. The device behaves exactly as it did before the feature was introduced. No route filter walkup occurs in any policy.
- Case #2: Route filter walkup is not enabled globally, but is enabled locally for a specific policy named **RouteFilter-Case2**. Route filter walkup occurs in this policy.

  To configure the route filter walkup locally for a specific policy:

  1. Enable the walkup feature locally for this policy statement.

     ```
     [edit policy-options]
     user@host# set policy-statement RouteFilter-Case2 defaults route-filter walkup
     ```

  2. Configure policy terms locally (walkup applies to all terms in this policy).

     ```
     [edit policy-options]
     user@host# set policy-statement RouteFilter-Case2 term ...
     ```

  3. Apply the policy statement to a routing protocol.

- Case #3: Route filter **walkup** is not enabled globally, but **no-walkup** is enabled locally for a specific policy named **RouteFilter-Case3**. (This case is not particularly helpful, because no walkup takes place in all policies by default, but does make local behavior explicit, even if walkup is enabled globally in the future.)

  To configure the route filter no-walkup locally for a specific policy:

  1. Enable the **no-walkup** feature locally for this policy statement.

     ```
     [edit policy-options]
     user@host# set policy-statement RouteFilter-Case3 defaults route-filter no-walkup
     ```

  2. Configure policy terms locally (**no-walkup** applies to this policy).

     ```
     [edit policy-options]
     user@host# set policy-statement RouteFilter-Case3 term ...
     ```

  3. Apply the policy statement to a routing protocol.

- Case #4: Route filter **walkup** is enabled globally, but not enabled locally for a specific policy named **RouteFilter-Case4**. Because of the global configuration, route filter **walkup** occurs in this policy.

  To configure the route filter walkup globally for a device:

  1. Enable the walkup feature globally for this device.

     ```
     [edit policy-options]
     user@host# set defaults route-filter walkup
     ```

     NOTE: Global **walkup**, in contrast to the **walkup** or **no-walkup** statements configured locally in a policy statement, is configured at the **[edit policy-options defaults]** or **[edit logical-systems** ***logical-system-name*** **policy-options defaults]** hierarchy level and applies to all policies.

  2. Configure policy statement **RouteFilter-Case4** and terms locally (walkup applies to this policy).

     ```
     [edit policy-options]
     user@host# set policy-statement RouteFilter-Case4 term ...
     ```

  3. Apply the policy statement to a routing protocol.

- Case #5: Route filter **walkup** is enabled globally, and enabled locally for a specific policy named **RouteFilter-Case5**. Although this configuration might appear redundant (walkup enabled globally as well as locally), this ensures that route filter walkup occurs in this policy even if route filter walkup is deleted at the global level.

  To configure the route filter walkup globally for a device and locally for a specific policy:

  1. Enable the **walkup** feature globally for this device.

     ```
     [edit policy-options]
     user@host# set defaults route-filter walkup
     ```

     NOTE: Global walkup is configured at the **[edit policy-options defaults]** or **[edit logical-systems** ***logical-system-name*** **policy-options defaults]** hierarchy level and applies to all policies.

  2. Configure policy statement **RouteFilter-Case5** and enable **walkup** locally (**walkup** applies to this policy).

     ```
     [edit policy-options]
     user@host# set policy-statement Route-Filter-Case5 defaults route-filter walkup
     ```

  3. Configure policy statement **RouteFilter-Case5** and terms locally (walkup applies to this policy).


```
[edit policy-options]
user@host# set policy-statement RouteFilter-Case5 term ...
```

4. Apply the policy statement to a routing protocol.

- Case #6: Route filter **walkup** is enabled globally, but overridden locally with **no-walkup** for a specific policy named **RouteFilter-Case6**. Because of the local configuration, no route filter walkup occurs in this policy. This case is useful to make sure that a local policy still functions exactly as before global walkup was enabled.

  To configure the route filter walkup globally for a device and the no-walkup feature locally for a specific policy:

  1. Enable the walkup feature globally for this device.

     ```
     [edit policy-options]
     user@host# set defaults route-filter walkup
     ```

     NOTE: Global walkup is configured at the **[edit policy-options defaults]** or **[edit logical-systems *logical-system-name* policy-options defaults]** hierarchy level and applies to all policies.

  2. Configure policy statement **RouteFilter-Case6** and disable walkup locally with the **no-walkup** statement (no walkup is performed in this policy).

     ```
     [edit policy-options]
     user@host# set policy-statement Route-Filter-Case6 defaults route-filter walkup
     ```

  3. Configure policy statement **RouteFilter-Case6** and terms locally.

     ```
     [edit policy-options]
     user@host# set policy-statement RouteFilter-Case6 term ...
     ```

  4. Apply the policy statement to a routing protocol.

NOTE: Keep in mind that a policy statement does nothing until it is applied as an import or export policy for the routing protocol itself. For BGP, this can be done at the global, group or neighbor level.

Related Documentation

- Walkup for Route Filters Overview on page 234
- Example: Configuring Walkup for Route Filters Globally to Improve Operational Efficiency on page 247
- Example: Configuring Walkup for Route Filters Locally to Improve Operational Efficiency on page 252

- Route Filter Match Conditions on page 53
- *Verify That a Particular BGP Route Is Received on Your Router*
- *Example: Configuring BGP Route Advertisement*

Example: Configuring Route Filter Lists

Junos OS has long supported route filters for use in policy statements. Whenever policies are changed, the route filters have to be processed inline with the policy. Policies that contain large numbers of route filters take time to load.

This example shows how to create a route filter list and use that list in a policy statement. Route filter lists reduce the amount of time needed to reload a given policy.

NOTE: **There is no speed benefit to using route filter lists in place of individual route filter entries when there are only a few route filters to process. The speed benefit is seen mainly in environments where there are hundreds or thousands of route filters listed within the policies.**

Requirements

- A router configured with a routing protocol such as BGP or OSPF that is actively exchanging route information with its peers.
- The router that is configured with route filter lists must be running Junos OS Release 15.2 or later.

Overview

The **route-filter-list** statement allows for the creation of a pre-defined list of route filters for use in routing policies. You configure the list at the **[edit policy-options]** hierarchy level. The configured route filter list is then referenced as a match condition in the **from** section of a policy statement at the **[edit policy-options policy-statement *policy-statement-name* term *term-name* from]** hierarchy level.

In this example, the router that you are configuring is receiving some routes from its BGP neighbor 192.0.2.1. This is shown in the output of the **show route receive-protocol bgp 192.0.2.1** operational command.

```
user@router> show route receive-protocol bgp 192.0.2.1
inet.0: 17 destinations, 18 routes (16 active, 0 holddown, 1 hidden)
  Prefix            Nexthop          MED     Lclpref      AS path
* 198.151.100.0/29         192.0.2.1                            103 I
```

```
* 198.151.100.8/29        192.0.2.1                           103 I
* 203.0.113.0/29          192.0.2.1                           103 I
* 203.0.113.8/29          192.0.2.1                           103 I
* 203.0.113.16/29         192.0.2.1                           103 I
```

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set policy-options route-filter-list rf-list-1 203.0.113.0/29 exact
set policy-options route-filter-list rf-list-1 203.0.113.8/29 exact
set policy-options route-filter-list rf-list-1 203.0.113.16/29 orlonger accept
set policy-options policy-statement rf-test-policy term term2 from route-filter
  198.51.100.0/29 upto 198.51.100.0/30
set policy-options policy-statement rf-test-policy term term2 from route-filter
  198.51.100.8/29 upto 198.51.100.8/30 accept
set policy-options policy-statement rf-test-policy term term2 from route-filter-list rf-list-1
set policy-options policy-statement rf-test-policy then reject
set protocols bgp group test-group import rf-test-policy
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

The following step-by-step procedure will lead you through the steps needed to:

- Configure a route filter list named **rf-list-1** and populate the list for later use in a route policy.
- Configure a routing policy statement named **rf-test-policy** that uses route filters and the configured route filter list.
- Configure BGP to use **rf-test-policy** as an import filter.

1. Configure a route filter list named **rf-list-1** for later use in a route policy.

   ```
   [edit policy-options]
   user@router# set route-filter-list rf-list-1
   ```

2. Populate the list **rf-list-1**.

   Note that one of the statements in the list has an action configured. This action will be carried out immediately upon a match with a received destination prefix.

   ```
   [edit policy-options]
   user@router# set route-filter-list rf-list-1 203.0.113.0/29 exact
   user@router# set route-filter-list rf-list-1 203.0.113.8/29 exact
   user@router# set route-filter-list rf-list-1 203.0.113.16/29 orlonger accept
   ```


3. Configure a routing policy statement named **rf-test-policy** that uses route filters and the configured route filter list.

   The overall action for this policy is **reject.** There are individual route filters and elements of the route filter list that have a configured action of **accept**. The actions configured in the individual route filter statements and elements of the route filter list are carried out immediately upon matching a received destination prefix.

   ```
   [edit policy-options]
   user@router# set policy-statement rf-test-policy term term2 from route-filter
     198.51.100.0/29 upto 198.51.100.0/30
   user@router# set policy-statement rf-test-policy term term2 from route-filter
     198.51.100.8/29 upto 198.51.100.8/30 accept
   user@router# set policy-statement rf-test-policy term term2 from route-filter-list
     rf-list-1
   user@router# set policy-statement rf-test-policy then reject
   ```

4. Configure BGP to use the configured policy as an import filter to selectively allow some routes and reject other routes from being added to the routing table.

   ```
   [edit protocols bgp group test-group]
   user@router# set import rf-test-policy
   ```

Verification

Verifying the Configured Route Filter List

Purpose To confirm that the route filter list is properly configured, issue the **show policy-options route-filter-list** ***route-filter-list-name*** command at the **[edit]** hierarchy level.

Action

```
[edit]
user@routershow policy-options route-filter-list rf-list-1
203.0.113.0/29 exact;
203.0.113.8/29 exact;
203.0.113.16/29 orlonger accept;
```

Meaning The output shows that the stored configuration is correct.

Verifying the Configured Policy Statement

Purpose To confirm that the policy statement is properly configured, issue the **show policy-options policy-statement** ***policy-statement-name*** command at the **[edit]** hierarchy level.

Action

```
[edit]
user@router# show policy-options policy-statement rf-test-policy
from {
  route-filter 198.51.100.0/29 upto 198.51.100.0/30;
  route-filter 198.51.100.8/29 upto 198.51.100.8/30 accept;
  route-filter-list rf-list-1;
}
then reject;
```

Meaning The output confirms that the stored configuration is correct.

Verifying That the Policy Statement Is Applied as an Import Policy in the BGP Protocol

Purpose To confirm that the configured policy statement is applied as an import policy in the BGP Protocol, issue the **show protocols bgp import** command at the **[edit]** hierarchy level.

Action

```
[edit]
user@router# show protocols bgp import
import rf-test-policy;
```

Meaning The outptut confirms that the stored configuration is correct.

If you have not already done so, you can issue the **commit** command at the **[edit]** hierarchy level so that the configuration is made active.

Verifying That the Route Filter List Is Operating as Expected

Purpose Now that the configuration has been verified and committed, confirm the operation of the route filter list by issuing the **show route receive-protocol bgp 192.0.2.1** operational command.

Action If you compare this output with the output of the same command issued prior to configuring the route filter list and policy statement, you see that some routes are no longer installed in the routing table.

```
user@router> show route receive-protocol bgp 192.0.2.1
inet.0: 14 destinations, 15 routes (13 active, 0 holddown, 1 hidden)
  Prefix             Nexthop         MED     Lclpref    AS path
* 198.151.100.8/29          192.0.2.1                           103 I
* 203.0.113.16/29           192.0.2.1                           103 I
```

Meaning The output shows that three of the five previously installed BGP routes have been rejected by the policy statement **rf-test-policy**. The only routes that remain from the previous list are the two that had **accept** actions listed as part of the filter definition. The other routes were rejected by the action of the **policy-statement**.

Related Documentation

- **route-filter-list on page 1236**
- Understanding Route Filter and Source Address Filter Lists for Use in Routing Policy Match Conditions on page 231

Example: Configuring Walkup for Route Filters Globally to Improve Operational Efficiency

Use the walkup feature if you have concerns about policy performance because of split route filters across multiple policy terms. The walkup feature enables the consolidation of route filters under one policy term.

This example shows how to configure the route filter walkup feature globally for policy statements with route filters. When configured at the global level, the route filter walkup option applies to all policy statements. This example changes the default behavior of policy terms with multiple route filters globally, so that any reversion to the default "no walkup" behavior must be established locally.

Requirements

This example uses the following hardware and software components:

- A Juniper Networks router
- A Junos operating system from 13.3 or above

Before you configure route filter walkup locally, be sure you have:

- A properly configured routing policy or set of routing policies
- A need to consolidate multiple route filter terms into fewer routing policy terms

Overview

Routing protocols exchange information with other routers running the same routing protocols. In many cases, route filters are used in routing policy statements to filter prefixes for import or export. In some cases, when route filters are split into many separate terms, performance is impacted. The route filter walkup feature allows consolidation of policy statement terms for operational efficiency.

This example uses BGP, but the same walkup feature applies to any routing protocol that supports route filtering of input or output.

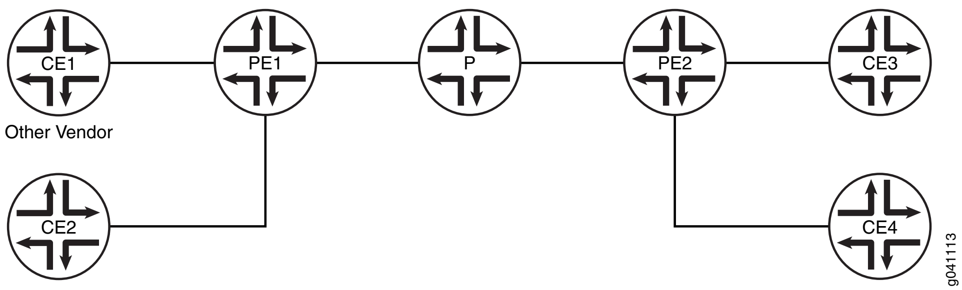
You can configure a Juniper Networks router to change the default operation of a term in a policy statement with route filters. By default, only a single longest match attempt is made for all route filters in a term. The walkup feature allows the router to "walk up" the route filters in a term from longest match to less specific in search of a true condition. This allows consolidation of multiple terms in a policy statement and corresponding operational efficiency.

This example changes the default behavior globally, for all policy statements. You can still configure **no-walkup** for an individual policy.

Topology

In the sample network in Figure 25 on page 248, the router CE1 is a router from another vendor. The rest are Juniper Networks routers. The walkup feature can be configured on any router in the figure, except for router CE1. The vendor of router CE1 might or not might support a similar feature.

Figure 25: Topology for the Global Walkup Example



In the example, the following addresses are used:

- 10.0.0.0/16
- 10.0.0.0/8

NOTE: Although the 10.0.0.0/8 address space is not specifically reserved for documentation, the private RFC 1918 10.0.0.0/8 address space is used in this topic because of the flexibility and realistic scenarios that this address spaces provides.

Configuring Route Filter Walkup Globally

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details such as addresses and interfaces to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device PE1

```
set policy-options defaults route-filter walkup
set policy-options policy-statement routeset1-import term prefixes1 from route-filter
  10.0.0.0/16 prefix-length-range /22-/24
set policy-options policy-statement routeset1-import term prefixes1 from route-filter
  10.0.0.0/8 orlonger
set policy-options policy-statement routeset1-import term prefixes1 then accept
```

```
set policy-options policy-statement routeset1-import term reject-the-rest then reject
set policy-options policy-statement import-route-filter-a term import-routes from protocol
  bgp
set policy-options policy-statement import-route-filter-a term import-routes from policy
  routeset1-import
set policy-options policy-statement import-route-filter-a term import-routes then next
  policy
set policy-options policy-statement import-route-filter-a term all-others then reject
set policy-options policy-statement route-filter-a-export term all then reject
set protocols bgp group routeset1 type external
set protocols bgp group routeset1 neighbor 10.0.10.13 import import-route-filter-a
set protocols bgp group routeset1 neighbor 10.0.10.13 family inet unicast
set protocols bgp group routeset1 neighbor 10.0.10.13 export route-filter-a-export
set protocols bgp group routeset1 neighbor 10.0.10.13 peer-as 64506
```

Step-by-Step Procedure

The following example requires that you navigate to various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*

To configure router PE1 to perform walkup globally and combine multiple route filters in one term:

1. Configure the walkup feature globally.

```
[edit policy-options defaults]
user@PE1# set route-filter walkup
```

2. Configure the policy statements for an import policy named **routeset1-import**.

```
[edit policy-options]
user@PE1# set policy-statement routeset1-import term prefixes1 from route-filter
  10.0.0.0/16 prefix-length-range /22-/24
user@PE1# set policy-statement routeset1-import term prefixes1 from route-filter
  10.0.0.0/8 orlonger
user@PE1# set policy-statement routeset1-import term prefixes1 then accept
user@PE1# set policy-statement routeset1-import term reject-the-rest then reject
```

3. Configure the policy options for the import and export policy statements.

```
[edit policy-options]
user@PE1# set policy-statement import-route-filter-a term import-routes from
  protocol bgp
user@PE1# set policy-statement import-route-filter-a term import-routes from
  policy routeset1-import
user@PE1# set policy-statement import-route-filter-a term import-routes then next
  policy
user@PE1# set policy-statement route-filter-a-export term all-others then reject
```

4. Apply the import and export policies to a BGP neighbor.

```
[edit protocols bgp]
user@PE1# set group routeset1 type external
user@PE1# set group routeset1 neighbor 10.0.10.13 import import-route-filter-a
user@PE1# set group routeset1 neighbor 10.0.10.13 family inet unicast
```

```
user@PE1# set group routeset1 neighbor 10.0.10.13 export route-filter-a-export
user@PE1# set group routeset1 neighbor 10.0.10.13 peer-as 64506
```

Results

From configuration mode, confirm your configuration by entering the **show protocols** and **show policy-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options
  defaults {
    route-filter walkup;
  }
  policy-statement routeset1-import {
  term prefixes1 {
    from {
      route-filter 10.0.0.0/16 prefix-length-range /22-/24;
      route-filter 10.0.0.0/8 orlonger;
    }
    then accept;
  }
  term reject-the-rest {
    then reject;
  }
}

policy-statement import-route-filter-a {
  term import-routes {
    from {
      protocol bgp;
      policy routeset1-import;
    }
    then next policy;
  }
  term all-others {
    then reject;
  }
}
policy-statement route-filter-a-export {
  term all {
    then reject;
  }
}

user@PE!# show protocols bgp
group routeset1 {
  type external;
  neighbor 10.0.10.13 {
    import import-route-filter-a;
    family inet {
      unicast;
    }
    export router-filter-a-export;
```

```
        peer-as 64506;
      }
    }
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Verifying Route Filter Operation

Purpose Display expected information about the routes to confirm the route filters are working as expected.

Notice that the **10.0.0.0/8 orlonger** filter includes the **10.0.0.0/16 prefix-length-range /22-/24** filter in its scope. That is, any **10.0.0.0** route with a prefix of 8 bits or longer could also be a route with a prefix in the range between 22 and 24 bits. Without the walkup feature enabled, a route such as **10.0.0.0/16** would be rejected and become a hidden route. If the walkup feature is working as expected, then a route such as **10.0.0.0/16** would be accepted by the policy.

Action From operational mode, enter the show route protocol**bgp 10.0.0.0/16** command. Make sure that **10.0.0.0/16** is not a hidden route.

```
user@PE1>show route protocol bgp 10.0.0.0/16
inet.0: 520762 destinations, 520764 routes (520760 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/16       *[BGP/170] 01:07:37, localpref 100
                       AS path: 64506, I, validation-state:  unverified
                     > to 10.0.100.13 via xe-0/2/0.0
```

As a further check, make sure that no routes that should be accepted are hidden routes. From operational mode, enter the show route protocol **bgp** ***ip-address-prefix*** **hidden** command to verify this.

Meaning The presence of routes that are not the longest match in the configured policy route filter term shows that the walkup feature is functioning globally.

Troubleshooting

To troubleshoot route filter walkup globally:

- Troubleshooting BGP on page 251
- Troubleshooting Policy Statements on page 252
- Troubleshooting Route Filters on page 252

Troubleshooting BGP

Problem BGP is not functioning as expected.

Solution See the *BGP Configuration Overview* topic, examples, and troubleshooting.

Troubleshooting Policy Statements

Problem The policy statements are not functioning as expected.

Solution See the *Verify That a Particular BGP Route Is Received on Your Router* and *Example: Configuring BGP Route Advertisement* topics, related examples, and troubleshooting.

Troubleshooting Route Filters

Problem The route filters are not functioning as expected.

Solution See the "Route Filter Match Conditions" on page 53 topic, examples, and troubleshooting.

Related Documentation

- Example: Configuring Walkup for Route Filters Locally to Improve Operational Efficiency on page 252
- Walkup for Route Filters Overview on page 234
- Configuring Walkup for Route Filters to Improve Operational Efficiency on page 238
- Route Filter Match Conditions on page 53

- *BGP Configuration Overview*
- *Verify That a Particular BGP Route Is Received on Your Router*
- *Example: Configuring BGP Route Advertisement*

Example: Configuring Walkup for Route Filters Locally to Improve Operational Efficiency

Use the walkup feature if you have concerns about policy performance because of split route filters across multiple policy terms. The walkup feature enables the consolidation of route filters under one policy term.

This example shows how to configure the route filter walkup feature locally for policy statements with route filters. When configured at the local level, the route filter walkup option applies only to the policy statement in which it is configured. This example does *not* change the default behavior of policy terms with route filters globally. This example establishes route filter walkup locally.

Requirements

This example uses the following hardware and software components:

- A Juniper Networks router
- A Junos operating system from 13.3 or above

Before you configure route filter walkup globally, be sure you have:

- A properly configured routing policy or set of routing policies
- A need to consolidate multiple route filter terms into fewer routing policy terms

Overview

Routing protocols exchange information with other routers running the same routing protocols. In many cases, route filters are used in routing policy statements to filter prefixes for import or export. In some cases, when route filters are split into many separate terms, performance is impacted. The route filter walkup feature allows consolidation of policy statement terms for operational efficiency.

This example uses BGP, but the same walkup feature applies to any routing protocol that supports route filtering of input or output.

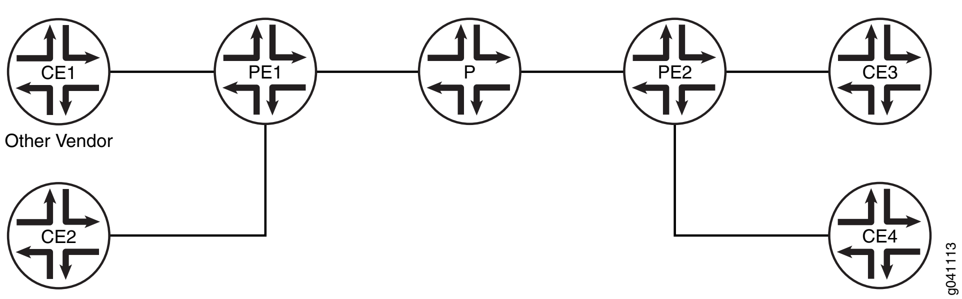
You can configure a Juniper Networks router to change the default operation of a term in a policy statement with route filters. By default, only a single longest match attempt is made for all route filters in a term. The walkup feature allows the router to “walk up” the route filters in a term from longest match to less specific in search of a true condition. This allows consolidation of multiple terms in a policy statement and corresponding operational efficiency.

This example changes the default behavior locally in a single policy statement. It does not affect the behavior of other policy statements.

Topology

In the sample network in Figure 25 on page 248, the router CE1 is a router from another vendor. The rest are Juniper Networks routers. The walkup feature can be configured on any router in the figure, except for router CE1. The vendor of router CE1 might or not might support a similar feature.

Figure 26: Topology for the Local Walkup Example



In the example, the following addresses are used:

- 10.0.0.0/16
- 10.0.0.0/8

NOTE: Although the **10.0.0.0/8** address space is not specifically reserved for documentation, the private RFC 1918 **10.0.0.0/8** address space is used in this topic because of the flexibility and realistic scenarios that this address spaces provides.

Configuring Route Filter Walkup Locally

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details such as addresses and interfaces to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device PE1

```
set policy-options policy-statement routeset1-import defaults route-filter walkup
set policy-options policy-statement routeset1-import term prefixes1 from route-filter
   10.0.0.0/16 prefix-length-range /22-/24
set policy-options policy-statement routeset1-import term prefixes1 from route-filter
   10.0.0.0/8 orlonger
set policy-options policy-statement routeset1-import term prefixes1 then accept
set policy-options policy-statement routeset1-import term reject-the-rest then reject
set policy-options policy-statement import-route-filter-a term import-routes from protocol
   bgp
set policy-options policy-statement import-route-filter-a term import-routes from policy
   routeset1-import
set policy-options policy-statement import-route-filter-a term import-routes then next
   policy
set policy-options policy-statement import-route-filter-a term all-others then reject
set policy-options policy-statement route-filter-a-export term all then reject
set protocols bgp group routeset1 type external
set protocols bgp group routeset1 neighbor 10.0.10.13 import import-route-filter-a
set protocols bgp group routeset1 neighbor 10.0.10.13 family inet unicast
set protocols bgp group routeset1 neighbor 10.0.10.13 export route-filter-a-export
set protocols bgp group routeset1 neighbor 10.0.10.13 peer-as 64506
```

Step-by-Step Procedure

The following example requires that you navigate to various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*

To configure router PE1 to perform walkup locally for multiple route filters in one term:

1. Configure the walkup feature locally in a policy named **routeset1-import**.

   ```
   [edit policy-options policy-statement routeset1-import defaults]
   user@PE1# set route-filter walkup
   ```

2. Configure the policy statements for an import policy named **routeset1-import**.


```
[edit policy-options ]
user@PE1# set policy-statement routeset1-import term prefixes1 from route-filter
  10.0.0.0/16 prefix-length-range /22-/24
user@PE1# set policy-statement routeset1-import term prefixes1 from route-filter
  10.0.0.0/8 orlonger
user@PE1# set policy-statement routeset1-import term prefixes1 then accept
user@PE1# set policy-statement routeset1-import term reject-the-rest then reject
```

3. Configure the policy options for the import and export policy statements.

```
[edit policy-options]
user@PE1# set policy-statement import-route-filter-a term import-routes from
  protocol bgp
user@PE1# set policy-statement import-route-filter-a term import-routes from
  policy routeset1-import
user@PE1# set policy-statement import-route-filter-a term import-routes then next
  policy
user@PE1# set policy-statement route-filter-a-export term all-others then reject
```

4. Apply the import and export policies to a BGP neighbor.

```
[edit protocols bgp]
user@PE1# set group routeset1 type external
user@PE1# set group routeset1 neighbor 10.0.10.13 import import-route-filter-a
user@PE1# set group routeset1 neighbor 10.0.10.13 family inet unicast
user@PE1# set group routeset1 neighbor 10.0.10.13 export route-filter-a-export
user@PE1# set group routeset1 neighbor 10.0.10.13 peer-as 64506
```

Results

From configuration mode, confirm your configuration by entering the **show protocols** and **show policy-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show policy-options
policy-statement routeset1-import {
  defaults {
    route-filter walkup;
  }
  term prefixes1 {
    from {
      route-filter 10.0.0.0/16 prefix-length-range /22-/24;
      route-filter 10.0.0.0/8 orlonger;
    }
    then accept;
  }
  term reject-the-rest {
    then reject;
  }
}

policy-statement import-route-filter-a {
  term import-routes {
```

```
        from {
            protocol bgp;
            policy routeset1-import;
        }
        then next policy;
    }
    term all-others {
        then reject;
    }
}
policy-statement route-filter-a-export {
    term all {
        then reject;
    }
}

user@PE!# show protocols bgp
group routeset1 {
    type external;
    neighbor 10.0.10.13 {
        import import-route-filter-a;
        family inet {
            unicast;
        }
        export router-filter-a-export;
        peer-as 64506;
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Verifying Route Filter Operation

Purpose Display expected information about the routes to confirm the route filters are working as expected.

Notice that the **10.0.0.0/8 orlonger** filter includes the **10.0.0.0/16 prefix-length-range /22-/24** filter in its scope. That is, any **10.0.0.0** route with a prefix of 8 bits or longer could also be a route with a prefix in the range between 22 and 24 bits. Without the walkup feature enabled in the policy example given, a route such as **10.0.0.0/16** would be rejected and become a hidden route. If the walkup feature is working as expected, then a route such as **10.0.0.0/16** would be accepted by the policy.

Action From operational mode, enter the show route protocol**bgp 10.0.0.0/16** command. Make sure that **10.0.0.0/16** is not a hidden route.

```
user@PE1>show route protocol bgp 10.0.0.0/16
inet.0: 520762 destinations, 520764 routes (520760 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/16       *[BGP/170] 01:07:37, localpref 100
                       AS path: 64506, I, validation-state:  unverified
                     > to 10.0.100.13 via xe-0/2/0.0
```

As a further check, make sure that no routes that should be accepted are hidden routes. From operational mode, enter the show route protocol **bgp *ip-address-prefix* hidden** command to verify this.

Meaning The presence of routes that are not the longest match in the configured policy route filter term shows that the walkup feature is functioning locally.

Troubleshooting

To troubleshoot route filter walkup locally:

Troubleshooting BGP

Problem BGP is not functioning as expected.

Solution See the *BGP Configuration Overview* topic, examples, and troubleshooting.

Troubleshooting Policy Statements

Problem The policy statements are not functioning as expected.

Solution See the *Verify That a Particular BGP Route Is Received on Your Router* and *Example: Configuring BGP Route Advertisement* topics, related examples, and troubleshooting.

Troubleshooting Route Filters

Problem The route filters are not functioning as expected.

Solution See the "Route Filter Match Conditions" on page 53 topic, examples, and troubleshooting.

Related Documentation

- Example: Configuring Walkup for Route Filters Globally to Improve Operational Efficiency on page 247
- Walkup for Route Filters Overview on page 234
- Configuring Walkup for Route Filters to Improve Operational Efficiency on page 238
- Route Filter Match Conditions on page 53
- *BGP Configuration Overview*
- *Verify That a Particular BGP Route Is Received on Your Router*
- *Example: Configuring BGP Route Advertisement*

Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF

This example shows how to create an OSPF import policy that prioritizes specific prefixes learned through OSPF.

- Requirements on page 258
- Overview on page 258
- Configuration on page 259
- Verification on page 262

Requirements

Before you begin:

- Configure the device interfaces. See the *Interfaces Feature Guide for Security Devices*.
- Configure the router identifiers for the devices in your OSPF network. See *Example: Configuring an OSPF Router Identifier*.
- Control OSPF designated router election See *Example: Controlling OSPF Designated Router Election*
- Configure a single-area OSPF network. See *Example: Configuring a Single-Area OSPF Network* .
- Configure a multiarea OSPF network. See *Example: Configuring a Multiarea OSPF Network*.

Overview

In a network with a large number of OSPF routes, it can be useful to control the order in which routes are updated in response to a network topology change. In Junos OS Release 9.3 and later, you can specify a priority of high, medium, or low for prefixes included in an OSPF import policy. In the event of an OSPF topology change, high priority prefixes are updated in the routing table first, followed by medium and then low priority prefixes.

OSPF import policy can only be used to set priority or to filter OSPF external routes. If an OSPF import policy is applied that results in a **reject** terminating action for a nonexternal route, then the **reject** action is ignored and the route is accepted anyway. By default, such a route is now installed in the routing table with a priority of low. This behavior prevents traffic black holes, that is, silently discarded traffic, by ensuring consistent routing within the OSPF domain.

In general, OSPF routes that are not explicitly assigned a priority are treated as priority medium, except for the following:

- Summary discard routes have a default priority of low.
- Local routes that are not added to the routing table are assigned a priority of low.
- External routes that are rejected by import policy and thus not added to the routing table are assigned a priority of low.

Any available match criteria applicable to OSPF routes can be used to determine the priority. Two of the most commonly used match criteria for OSPF are the **route-filter** and **tag** statements.

In this example, the routing device is in area 0.0.0.0, with interfaces **fe-0/1/0** and **fe-1/1/0** connecting to neighboring devices. You configure an import routing policy named ospf-import to specify a priority for prefixes learned through OSPF. Routes associated with these prefixes are installed in the routing table in the order of the prefixes' specified priority. Routes matching **192.0.2.0/24 orlonger** are installed first because they have a priority of **high**. Routes matching **198.51.100.0/24 orlonger** are installed next because they have a priority of **medium**. Routes matching **203.0.113.0/24 orlonger** are installed last because they have a priority of **low**. You then apply the import policy to OSPF.

NOTE: **The priority value takes effect when a new route is installed, or when there is a change to an existing route.**

Configuration

CLI Quick Configuration

To quickly configure an OSPF import policy that prioritizes specific prefixes learned through OSPF, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the [edit] hierarchy level, and then enter **commit** from configuration mode.

```
[edit]
set interfaces fe-0/1/0 unit 0 family inet address 192.168.8.4/30
set interfaces fe-0/1/0 unit 0 family inet address 192.168.8.5/30
set policy-options policy-statement ospf-import term t1 from route-filter 203.0.113.0/24
  orlonger
set policy-options policy-statement ospf-import term t1 then priority low
set policy-options policy-statement ospf-import term t1 then accept
set policy-options policy-statement ospf-import term t2 from route-filter 198.51.100.0/24
  orlonger
set policy-options policy-statement ospf-import term t2 then priority medium
```

```
set policy-options policy-statement ospf-import term t2 then accept
set policy-options policy-statement ospf-import term t3 from route-filter 192.0.2.0/24
  orlonger
set policy-options policy-statement ospf-import term t3 then priority high
set policy-options policy-statement ospf-import term t3 then accept
set protocols ospf import ospf-import
set protocols ospf area 0.0.0.0 interface fe-0/1/0
set protocols ospf area 0.0.0.0 interface fe-1/1/0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Modifying the Junos OS Configuration* in the*CLI User Guide*.

To configure an OSPF import policy that prioritizes specific prefixes:

1. Configure the interfaces.

   ```
   [edit]
   user@host# set interfaces fe-0/1/0 unit 0 family inet address 192.168.8.4/30
   user@host# set interfaces fe-0/2/0 unit 0 family inet address 192.168.8.5/30
   ```

2. Enable OSPF on the interfaces.

   NOTE: For OSPFv3, include the **ospf3** statement at the **[edit protocols]** hierarchy level.

   ```
   [edit]
   user@host# set protocols ospf area 0.0.0.0 interface fe-0/1/0
   user@host# set protocols ospf area 0.0.0.0 interface fe-0/2/0
   ```

3. Configure the policy to specify the priority for prefixes learned through OSPF.

   ```
   [edit ]
   user@host# set policy-options policy-statement ospf-import term t1 from route-filter
     203.0.113.0/24 orlonger
   user@host# set policy-options policy-statement ospf-import term t1 then priority
     low
   user@host# set policy-options policy-statement ospf-import term t1 then accept
   user@host# set policy-options policy-statement ospf-import term t2 from route-filter
     198.51.100.0/24 orlonger
   user@host# set policy-options policy-statement ospf-import term t2 then priority
     medium
   user@host# set policy-options policy-statement ospf-import term t2 then accept
   user@host# set policy-options policy-statement ospf-import term t3 from route-filter
     192.0.2.0/24 orlonger
   user@host# set policy-options policy-statement ospf-import term t3 then priority
     high
   user@host# set policy-options policy-statement ospf-import term t3 then accept
   ```

4. Apply the policy to OSPF.


```
[edit]
user@host# set protocols ospf import ospf-import
```

5. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Results Confirm your configuration by entering the **show interfaces**, **show policy-options**, and the **show protocols ospf** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show interfaces
fe-0/1/0 {
  unit 0 {
    family inet {
      address 192.168.8.4/30;
    }
  }
}
  fe-0/2/0 {
    unit 0 {
      family inet {
        address 192.168.8.5/30;
      }
    }
  }
```

```
user@host# show protocols ospf
import ospf-import;
area 0.0.0.0 {
  interface fe-0/1/0.0;
  interface fe-0/2/0.0;
}
```

```
user@host# show policy-options
policy-statement ospf-import {
  term t1 {
    from {
      route-filter 203.0.113.0/24 orlonger;
    }
    then {
      priority low;
      accept;
    }
  }
  term t2 {
    from {
      route-filter 198.51.100.0/24 orlonger;
    }
    then {
      priority medium;
      accept;
    }
```

```
    }
    term t3 {
      from {
        route-filter 192.0.2.0/24 orlonger;
      }
      then {
        priority high;
        accept;
      }
    }
  }
```

```
user@host# show protocols ospf
import ospf-import;
area 0.0.0.0 {
  interface fe-0/1/0.0;
  interface fe-0/2/0.0;
}
```

To confirm your OSPFv3 configuration, enter the **show interfaces**, **show policy-options**, and **show protocols ospf3** commands.

Verification

Confirm that the configuration is working properly.

Verifying the Prefix Priority in the OSPF Routing Table

Purpose Verify the priority assigned to the prefix in the OSPF routing table.

Action From operational mode, enter the **show ospf route detail** for OSPFv2, and enter the **show ospf3 route detail** command for OSPFv3.

Related Documentation

- Understanding Route Filters for Use in Routing Policy Match Conditions on page 213
- *OSPF Routing Policy Overview*

Example: Configuring the MED Using Route Filters

This example shows how to configure a policy that uses route filters to modify the multiple exit discriminator (MED) metric to advertise in BGP update messages.

Requirements

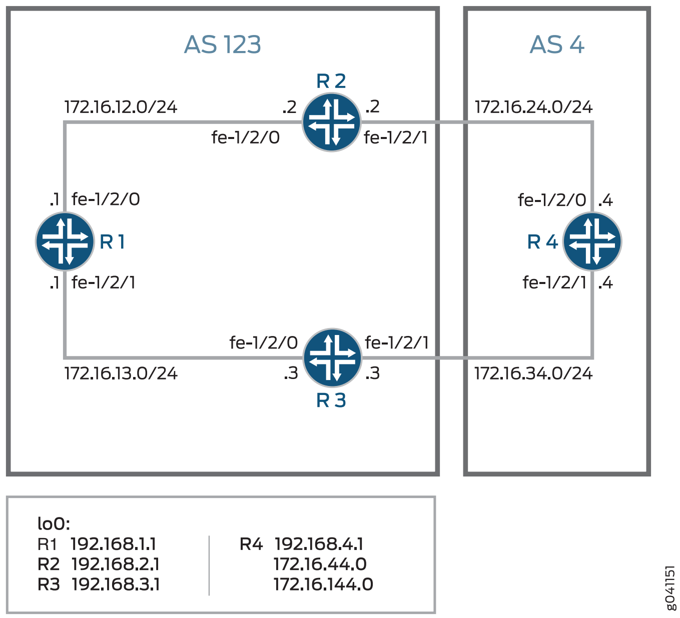
No special configuration beyond device initialization is required before you configure this example.

Overview

To configure a route-filter policy that modifies the advertised MED metric in BGP update messages, include the **metric** statement in the policy action.

Figure 27 on page 263 shows a typical network with internal peer sessions and multiple exit points to a neighboring autonomous system (AS).

Figure 27: Typical Network with IBGP Sessions and Multiple Exit Points



Device R4 has multiple loopback interfaces configured to simulate advertised prefixes. The extra loopback interface addresses are 172.16.44.0/32 and 172.16.144.0/32. This example shows how to configure Device R4 to advertise a MED value of 30 to Device R3 for all routes except 172.16.144.0. For 172.16.144.0, a MED value of 10 is advertised to Device 3. A MED value of 20 is advertised to Device R2, regardless of the route prefix.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 1 family inet address 172.16.12.1/24
set interfaces fe-1/2/1 unit 2 family inet address 172.16.13.1/24
set interfaces lo0 unit 1 family inet address 192.168.1.1/32
set protocols bgp group internal type internal
```

```
set protocols bgp group internal local-address 192.168.1.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.2.1
set protocols bgp group internal neighbor 192.168.3.1
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.1
set protocols ospf area 0.0.0.0 interface fe-1/2/1.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.1.1
```

Device R2

```
set interfaces fe-1/2/0 unit 3 family inet address 172.16.12.2/24
set interfaces fe-1/2/1 unit 4 family inet address 172.16.24.2/24
set interfaces lo0 unit 2 family inet address 192.168.2.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.2.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.1.1
set protocols bgp group internal neighbor 192.168.3.1
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 4
set protocols bgp group external neighbor 172.16.24.4
set protocols ospf area 0.0.0.0 interface lo0.2 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/1.4
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.2.1
```

Device R3

```
set interfaces fe-1/2/0 unit 5 family inet address 172.16.13.3/24
set interfaces fe-1/2/1 unit 6 family inet address 172.16.34.3/24
set interfaces lo0 unit 3 family inet address 192.168.3.1/32
set protocols bgp group internal type internal
set protocols bgp group internal local-address 192.168.3.1
set protocols bgp group internal export send-direct
set protocols bgp group internal neighbor 192.168.1.1
set protocols bgp group internal neighbor 192.168.2.1
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 4
set protocols bgp group external neighbor 172.16.34.4
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.5
set protocols ospf area 0.0.0.0 interface fe-1/2/1.6
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 123
set routing-options router-id 192.168.3.1
```

Device R4

```
set interfaces fe-1/2/0 unit 7 family inet address 172.16.24.4/24
set interfaces fe-1/2/1 unit 8 family inet address 172.16.34.4/24
```



```
set interfaces lo0 unit 4 family inet address 192.168.4.1/32
set interfaces lo0 unit 4 family inet address 172.16.44.0/32
set interfaces lo0 unit 4 family inet address 172.16.144.0/32
set protocols bgp group external type external
set protocols bgp group external export send-direct
set protocols bgp group external peer-as 123
set protocols bgp group external neighbor 172.16.34.3 export med-10
set protocols bgp group external neighbor 172.16.34.3 export med-30
set protocols bgp group external neighbor 172.16.24.2 metric-out 20
set policy-options policy-statement med-10 from route-filter 172.16.144.0/32 exact
set policy-options policy-statement med-10 then metric 10
set policy-options policy-statement med-10 then accept
set policy-options policy-statement med-30 from route-filter 0.0.0.0/0 longer
set policy-options policy-statement med-30 then metric 30
set policy-options policy-statement med-30 then accept
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 4
set routing-options router-id 192.168.4.1
```

Configuring Device R1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces fe-1/2/0 unit 1]
user@R1# set family inet address 172.16.12.1/24

[edit interfaces fe-1/2/1 unit 2]
user@R1# set family inet address 172.16.13.1/24

[edit interfaces lo0 unit 1]
user@R1# set family inet address 192.168.1.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R1# set type internal
user@R1# set local-address 192.168.1.1
user@R1# set export send-direct
user@R1# set neighbor 192.168.2.1
user@R1# set neighbor 192.168.3.1
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface lo0.1 passive
user@R1# set interface fe-1/2/0.1
user@R1# set interface fe-1/2/1.2
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

   ```
   [edit policy-options policy-statement send-direct term 1]
   user@R1# set from protocol direct
   user@R1# set then accept
   ```

5. Configure the router ID and autonomous system (AS) number.

   ```
   [edit routing-options]
   user@R1# set autonomous-system 123
   user@R1# set router-id 192.168.1.1
   ```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
fe-1/2/0 {
  unit 1 {
    family inet {
      address 172.16.12.1/24;
    }
  }
}
fe-1/2/1 {
  unit 2 {
    family inet {
      address 172.16.13.1/24;
    }
  }
}
lo0 {
  unit 1 {
    family inet {
      address 192.168.1.1/32;
    }
  }
}
```

```
user@R1# show protocols
bgp {
  group internal {
    type internal;
    local-address 192.168.1.1;
    export send-direct;
    neighbor 192.168.2.1;
    neighbor 192.168.3.1;
  }
}
ospf {
```



```
    area 0.0.0.0 {
        interface lo0.1 {
            passive;
        }
        interface fe-1/2/0.1;
        interface fe-1/2/1.2;
    }
}
```

```
user@R1# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R1# show routing-options
autonomous-system 123;
router-id 192.168.1.1;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Device R2

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces fe-1/2/0 unit 3]
user@R2# set family inet address 172.16.12.21/24
```

```
[edit interfaces fe-1/2/1 unit 4]
user@R2# set family inet address 172.16.24.2/24
```

```
[edit interfaces lo0 unit 2]
user@R2# set family inet address 192.168.2.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R2# set type internal
user@R2# set local-address 192.168.2.1
user@R2# set export send-direct
user@R2# set neighbor 192.168.1.1
user@R2# set neighbor 192.168.3.1
```

```
[edit protocols bgp group external]
user@R2# set type external
user@R2# set export send-direct
user@R2# set peer-as 4
```

```
user@R2# set neighbor 172.16.24.4
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R2# set interface lo0.2 passive
user@R2# set interface fe-1/2/0.3
user@R2# set interface fe-1/2/1.4
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

5. Configure the router ID and autonomous system (AS) number.

```
[edit routing-options]
user@R2# set autonomous-system 123
user@R2# set router-id 192.168.2.1
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
fe-1/2/0 {
  unit 3 {
    family inet {
      address 172.16.12.2/24;
    }
  }
}
fe-1/2/1 {
  unit 4 {
    family inet {
      address 172.16.24.2/24;
    }
  }
}
lo0 {
  unit 2 {
    family inet {
      address 192.168.2.1/32;
    }
  }
}
```

```
user@R2# show protocols
bgp {
```



```
    group internal {
      type internal;
      local-address 192.168.2.1;
      export send-direct;
      neighbor 192.168.1.1;
      neighbor 192.168.3.1;
    }
    group external {
      type external;
      export send-direct;
      peer-as 4;
      neighbor 172.16.24.4;
    }
  }
  ospf {
    area 0.0.0.0 {
      interface lo0.2 {
        passive;
      }
      interface fe-1/2/0.3;
      interface fe-1/2/1.4;
    }
  }
```

```
  user@R2# show policy-options
  policy-statement send-direct {
    term 1 {
      from protocol direct;
      then accept;
    }
  }
```

```
  user@R2# show routing-options
  autonomous-system 123;
  router-id 192.168.2.1;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Device R3

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R3:

1. Configure the device interfaces.

```
[edit interfaces fe-1/2/0 unit 5]
user@R3# set family inet address 172.16.13.3/24
```

```
[edit interfaces fe-1/2/1 unit 6]
user@R3# set family inet address 172.16.34.3/24
```

```
[edit interfaces lo0 unit 3]
```

```
user@R3# set family inet address 192.168.3.1/32
```

2. Configure BGP.

```
[edit protocols bgp group internal]
user@R3# set type internal
user@R3# set local-address 192.168.3.1
user@R3# set export send-direct
user@R3# set neighbor 192.168.1.1
user@R3# set neighbor 192.168.2.1

[edit protocols bgp group external]
user@R3# set type external
user@R3# set export send-direct
user@R3# set peer-as 4
user@R3# set neighbor 172.16.34.4
```

3. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R3# set interface lo0.3 passive
user@R3# set interface fe-1/2/0.5
user@R3# set interface fe-1/2/1.6
```

4. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R3# set from protocol direct
user@R3# set then accept
```

5. Configure the router ID and autonomous system (AS) number.

```
[edit routing-options]
user@R3# set autonomous-system 123
user@R3# set router-id 192.168.3.1
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R3# show interfaces
fe-1/2/0 {
  unit 5 {
    family inet {
      address 172.16.13.3/24;
    }
  }
}
fe-1/2/1 {
```



```
    unit 6 {
        family inet {
            address 172.16.34.3/24;
        }
    }
}
lo0 {
    unit 3 {
        family inet {
            address 192.168.3.1/32;
        }
    }
}
```

```
user@R3# show protocols
bgp {
    group internal {
        type internal;
        local-address 192.168.3.1;
        export send-direct;
        neighbor 192.168.1.1;
        neighbor 192.168.2.1;
    }
    group external {
        type external;
        export send-direct;
        peer-as 4;
        neighbor 172.16.34.4;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.3 {
            passive;
        }
        interface fe-1/2/0.5;
        interface fe-1/2/1.6;
    }
}
```

```
user@R3# show policy-options
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R3# show routing-options
autonomous-system 123;
router-id 192.168.3.1;
```

If you are done configuring the device, enter **commit** from configuration mode.

Configuring Device R4

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R4:

1. Configure the device interfaces.

   ```
   [edit interfaces fe-1/2/0 unit 7]
   user@R4# set family inet address 172.16.24.4/24
   ```

   ```
   [edit interfaces fe-1/2/1 unit 8]
   user@R4# set family inet address 172.16.34.4/24
   ```

   ```
   [edit interfaces lo0 unit 4]
   user@R4# set family inet address 192.168.4.1/32
   user@R4# set family inet address 172.16.44.0/32
   user@R4# set family inet address 172.16.144.0/32
   ```

   Device R4 has multiple loopback interface addresses to simulate advertised prefixes.

2. Configure a policy that accepts direct routes.

   Other useful options for this scenario might be to accept routes learned through OSPF or local routes.

   ```
   [edit policy-options policy-statement send-direct term 1]
   user@R4# set from protocol direct
   user@R4# set then accept
   ```

3. Configure BGP.

   ```
   [edit protocols bgp group external]
   user@R4# set type external
   user@R4# set export send-direct
   user@R4# set peer-as 123
   ```

4. Configure the two MED policies.

   ```
   [edit policy-options]
   set policy-statement med-10 from route-filter 172.16.144.0/32 exact
   set policy-statement med-10 then metric 10
   set policy-statement med-10 then accept

   set policy-statement med-30 from route-filter 0.0.0.0/0 longer
   set policy-statement med-30 then metric 30
   set policy-statement med-30 then accept
   ```

5. Configure the two EBGP neighbors, applying the two MED policies to Device R3, and a MED value of 20 to Device R2.

```
[edit protocols bgp group external]
user@R4# set neighbor 172.16.34.3 export med-10
user@R4# set neighbor 172.16.34.3 export med-30
user@R4# set neighbor 172.16.24.2 metric-out 20
```

6. Configure the router ID and autonomous system (AS) number.

```
[edit routing-options]
user@R4# set autonomous-system 4
user@R4# set router-id 192.168.4.1
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R4# show interfaces
fe-1/2/0 {
    unit 7 {
        family inet {
            address 172.16.24.4/24;
        }
    }
}
fe-1/2/1 {
    unit 8 {
        family inet {
            address 172.16.34.4/24;
        }
    }
}
lo0 {
    unit 4 {
        family inet {
            address 192.168.4.1/32;
            address 172.16.44.0/32;
            address 172.16.144.0/32;
        }
    }
}
```

```
user@R4# show protocols
bgp {
    group external {
        type external;
        export send-direct;
        peer-as 123;
        neighbor 172.16.24.2 {
            metric-out 20;
        }
        neighbor 172.16.34.3 {
            export [ med-10 med-30 ];
        }
    }
```

```
}
user@R4# show policy-options
policy-statement med-10 {
  from {
    route-filter 172.16.144.0/32 exact;
  }
  then {
    metric 10;
    accept;
  }
}
policy-statement med-30 {
  from {
    route-filter 0.0.0.0/0 longer;
  }
  then {
    metric 30;
    accept;
  }
}
policy-statement send-direct {
  term 1 {
    from protocol direct;
    then accept;
  }
}
user@R4# show routing-options
autonomous-system 4;
router-id 192.168.4.1;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Checking the Active Path from Device R1 to Device R4 on page 274
- Verifying That Device R4 Is Sending Its Routes Correctly on page 275

Checking the Active Path from Device R1 to Device R4

Purpose Verify that the active path goes through Device R2.

Action From operational mode, enter the **show route protocol bgp** command.

```
user@R1> show route protocol bgp
inet.0: 13 destinations, 19 routes (13 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.12.0/24       [BGP/170] 4d 01:13:32, localpref 100, from 192.168.2.1
                      AS path: I
                    > to 172.16.12.2 via fe-1/2/0.1
172.16.13.0/24       [BGP/170] 3d 05:36:10, localpref 100, from 192.168.3.1
```



```
                        AS path: I
                      > to 172.16.13.3 via fe-1/2/1.2
172.16.24.0/24         [BGP/170] 4d 01:13:32, localpref 100, from 192.168.2.1
                        AS path: I
                      > to 172.16.12.2 via fe-1/2/0.1
172.16.34.0/24         [BGP/170] 3d 05:36:10, localpref 100, from 192.168.3.1
                        AS path: I
                      > to 172.16.13.3 via fe-1/2/1.2
172.16.44.0/32       *[BGP/170] 00:06:03, MED 20, localpref 100, from 192.168.2.1
                        AS path: 4 I
                      > to 172.16.12.2 via fe-1/2/0.1
172.16.144.0/32 *[BGP/170] 00:06:03, MED 10, localpref 100, from 192.168.3.1
                        AS path: 4 I
                      > to 172.16.13.3 via fe-1/2/1.2
192.168.2.1/32        [BGP/170] 4d 01:13:32, localpref 100, from 192.168.2.1
                        AS path: I
                      > to 172.16.12.2 via fe-1/2/0.1
192.168.3.1/32        [BGP/170] 3d 05:36:10, localpref 100, from 192.168.3.1
                        AS path: I
                      > to 172.16.13.3 via fe-1/2/1.2
192.168.4.1/32       *[BGP/170] 00:06:03, MED 20, localpref 100, from 192.168.2.1
                        AS path: 4 I
                      > to 172.16.12.2 via fe-1/2/0.1
```

Meaning The output shows that the preferred path to the routes advertised by Device R4 is through Device R2 for all routes except 172.16.144.0/32. For 172.16.144.0/32, the preferred path is through Device R3.

Verifying That Device R4 Is Sending Its Routes Correctly

Purpose Make sure that Device R4 is sending update messages with a value of 20 to Device R2 and a value of 30 to Device R3.

Action From operational mode, enter the **show route advertising-protocol bgp** command.

```
user@R4> show route advertising-protocol bgp 172.16.24.2
inet.0: 11 destinations, 13 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.24.0/24           Self                 20                  I
* 172.16.34.0/24           Self                 20                  I
* 172.16.44.0/32          Self                 20                  I
* 172.16.144.0/32      Self                 20                  I
* 192.168.4.1/32          Self                 20                  I

user@R4> show route advertising-protocol bgp 172.16.34.3
inet.0: 11 destinations, 13 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.24.0/24           Self                 30                  I
* 172.16.34.0/24           Self                 30                  I
* 172.16.44.0/32          Self                 30                  I
* 172.16.144.0/32      Self                 10                  I
* 192.168.4.1/32          Self                 30                  I
```

Meaning The MED column shows that Device R4 is sending the correct MED values to its two EBGP neighbors.

Related Documentation

- *Example: Associating the MED Path Attribute with the IGP Metric and Delaying MED Updates*
- Understanding Route Filters for Use in Routing Policy Match Conditions on page 213
- *Understanding BGP Path Selection*
- *Understanding External BGP Peering Sessions*

Example: Configuring Layer 3 VPN Protocol Family Qualifiers for Route Filters

This example shows how to control the scope of BGP import policies by configuring a family qualifier for the BGP import policy. The family qualifier specifies routes of type **inet**, **inet6**, **inet-vpn**, or **inet6-vpn**.

Requirements

This example uses Junos OS Release 10.0 or later.

Before you begin:

- Configure the device interfaces.
- Configure an interior gateway protocol. See the *Junos OS Routing Protocols Library*.
- Configure a BGP session for multiple route types. For example, configure the session for both family **inet** routes and family **inet-vpn** routes. See *Configuring IBGP Sessions Between PE Routers in VPNs* and *Configuring Layer 3 VPNs to Carry IPv6 Traffic*.

Overview

Family qualifiers cause a route filter to match only one specific family. When you configure an IPv4 route filter without a family qualifier, as shown here, the route filter matches **inet** and **inet-vpn** routes.

route-filter *ipv4-address/mask*;

Likewise, when you configure an IPv6 route filter without a family qualifier, as shown here, the route filter matches **inet6** and **inet6-vpn** routes.

route-filter *ipv6-address/mask*;

Consider the case in which a BGP session has been configured for both family **inet** routes and family **inet-vpn** routes, and an import policy has been configured for this BGP session.

This means that both family **inet** and family **inet-vpn** routes, when received, share the same import policy. The policy term might look as follows:

```
from {
  route-filter 0.0.0.0/0 exact;
}
then {
  next-hop self;
  accept;
}
```

This route-filter logic matches an **inet** route of 0.0.0.0 and an **inet-vpn** route whose IPv4 address portion is 0.0.0.0. The 8-byte route distinguisher portion of the **inet-vpn** route is not considered in the route-filter matching. This is a change in Junos OS behavior that was introduced in Junos OS Release 10.0.

If you do not want your policy to match both types of routes, add a family qualifier to your policy. To have the route-filter match only **inet** routes, add the family **inet** policy qualifier. To have the route-filter match only **inet-vpn** routes, add the family **inet-vpn** policy qualifier.

The family qualifier is evaluated before the route-filter is evaluated. Thus, the route-filter is not evaluated if the family match fails. The same logic applies to family **inet6** and family **inet6-vpn**. The route-filter used in the **inet6** example must use an IPv6 address. There is a potential efficiency gain in using a family qualifier because the family qualifier is tested before most other qualifiers, quickly eliminating routes from undesired families.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

inet Example

```
set policy-options policy-statement specific-family from family inet
set policy-options policy-statement specific-family from route-filter 0.0.0.0/0 exact
set policy-options policy-statement specific-family then next-hop self
set policy-options policy-statement specific-family then accept
set protocols bgp import specific-family
```

Inet-vpn Example

```
set policy-options policy-statement specific-family from family inet-vpn
set policy-options policy-statement specific-family from route-filter 0.0.0.0/0 exact
set policy-options policy-statement specific-family then next-hop self
set policy-options policy-statement specific-family then accept
set protocols bgp import specific-family
```

inet6 Example

```
set policy-options policy-statement specific-family from family inet6
set policy-options policy-statement specific-family from route-filter 0::0/0 exact
set policy-options policy-statement specific-family then next-hop self
set policy-options policy-statement specific-family then accept
set protocols bgp import specific-family
```


Inet6-vpn Example

```
set policy-options policy-statement specific-family from family inet6-vpn
set policy-options policy-statement specific-family from route-filter 0::0/0 exact
set policy-options policy-statement specific-family then next-hop self
set policy-options policy-statement specific-family then accept
set protocols bgp import specific-family
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure a flow map:

1. Configure the family qualifier.

   ```
   [edit policy-options]
   user@host# set policy-statement specific-family from family inet
   ```

2. Configure the route filter.

   ```
   [edit policy-options]
   user@host# set policy-statement specific-family from route-filter 0.0.0.0/0 exact
   ```

3. Configure the policy actions.

   ```
   [edit policy-options]
   user@host# set policy-statement specific-family then next-hop self
   user@host# set policy-statement specific-family then accept
   ```

4. Apply the policy.

   ```
   [edit protocols bgp]
   user@host# set import specific-family
   ```

Results

From configuration mode, confirm your configuration by issuing the **show protocols** and **show policy-options** command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show protocols
bgp {
  import specific-family;
}
user@host# show policy-options
policy-statement specific-family {
  from {
    family inet;
    route-filter 0.0.0.0/0 exact;
  }
  then {
    next-hop self;
    accept;
```



```
    }
  }
```

If you are done configuring the device, enter **commit** from configuration mode.

Repeat the procedure for every protocol family for which you need a specific route-filter policy.

Verification

To verify the configuration, run the following commands:

- **show route advertising-protocol bgp *neighbor* detail**
- **show route instance *instance-name* detail**

Related Documentation

- Understanding Route Filters for Use in Routing Policy Match Conditions on page 213
- Route Filter Match Conditions on page 53
- Example: Configuring Policy Chains and Route Filters on page 186
- Example: Configuring the MED Using Route Filters on page 262
- Example: Configuring a Route Filter Policy to Specify Priority for Prefixes Learned Through OSPF on page 258

Understanding Prefix Lists for Use in Routing Policy Match Conditions

A *prefix list* is a named list of IP addresses. You can specify an exact match with incoming routes and apply a common action to all matching prefixes in the list.

Suppose, for example, that you configure the following prefix list:

```
prefix-list bgp179 {
  apply-path "protocols bgp group <*> neighbor <*>";
}
```

This works well when all neighbors on the device are in the same address family.

When the neighbors are in different address families, for example when both IPv4 and IPv6 neighbors are configured, you can use a prefix list as follows:

```
prefix-list IPV4-BGP-NEIGHBORS {
  apply-path "protocols bgp group <*> neighbor <*.*.*.*>";
}
prefix-list IPV6-BGP-NEIGHBORS {
  apply-path "protocols bgp group <*> neighbor <*:*:*>";
}
```

One prefix list matches IPv4 addresses. The other matches IPv6 addresses. You can run the **show configuration policy-options prefix-list prefix-list name | display inheritance** command to verify the configuration.

A prefix list functions like a route list that contains multiple instances of the **exact** match type only. The differences between these two extended match conditions are summarized in Table 19 on page 280.

Table 19: Prefix List and Route List Differences

Feature	Prefix List	Route Lists
Action	Can specify action in a **then** statement only. These actions are applied to all prefixes that match the term.	Can specify action that is applied to a particular prefix in a **route-filter** match condition in a **from** statement, or to all prefixes in the list using a **then** statement.

For information about configuring route lists, see "Understanding Route Filters for Use in Routing Policy Match Conditions" on page 213.

This section includes the following information:

Configuring Prefix Lists

You can create a named prefix list and include it in a routing policy with the **prefix-list** match condition (described in "Routing Policy Match Conditions" on page 44).

To define a prefix list, include the **prefix-list** statement:

```
[edit policy-options]
  prefix-list prefix-list-name {
  apply-path path;
  ip-addresses;
}
```

You can use the **apply-path** statement to include all prefixes (and their associated network mask) pointed to by a defined path, or you can specify one or more addresses, or both.

To include a prefix list in a routing policy, specify the **prefix-list** match condition in the **from** statement at the **[edit policy-options policy-statement *policy-name* term *term-name*]** hierarchy level:

```
[edit policy-options policy-statement policy-name term term-name]
from {
  prefix-list prefix-list-name;
}
then actions;
```

name identifies the prefix list. It can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose the entire name in quotation marks (" ").

ip-addresses are the IPv4 or IP version 6 (IPv6) prefixes specified as ***prefix/prefix-length***. If you omit ***prefix-length*** for an IPv4 prefix, the default is /32***prefix-length***. If you omit ***prefix-length*** for an IPv6 prefix, the default is /128. Prefixes specified in a **from** statement must be either all IPv4 addresses or all IPv6 addresses.

NOTE: You cannot apply actions to individual prefixes in the list.

You can specify the same prefix list in the **from** statement of multiple routing policies or firewall filters. For information about firewall filters, see "Guidelines for Configuring Firewall Filters" on page 576 and "Guidelines for Applying Standard Firewall Filters" on page 581.

Use the **apply-path** statement to configure a prefix list comprising all IP prefixes pointed to by a defined path. This eliminates most of the effort required to maintain a group prefix list.

The path consists of elements separated by spaces. Each element matches a configuration keyword or an identifier, and you can use wildcards to match more than one identifier. Wildcards must be enclosed in angle brackets, for example, <*>.

NOTE: You cannot add a path element, including wildcards, after a leaf statement in the **apply-path** statement. Path elements, including wildcards, can only be used after a container statement.

NOTE: When you use **apply-path** to define a prefix list, you can also use the same prefix list in a policy statement.

For examples of configuring a prefix list, see "Example: Configuring Routing Policy Prefix Lists" on page 282.

How Prefix Lists Are Evaluated in Routing Policy Match Conditions

During prefix list evaluation, the policy framework software performs a *longest-match lookup*, which means that the software searches for the prefix in the list with the longest length. The order in which you specify the prefixes, from top to bottom, does not matter. The software then compares a route's source address to the longest prefix.

You can use prefix list qualifiers for prefixes contained in a prefix list by configuring a prefix list filter. For more information, see *Configuring Prefix Lists for Use in Routing Policy Match Conditions*.

If a match occurs, the evaluation of the current term continues. If a match does not occur, the evaluation of the current term ends.

NOTE: If you specify multiple prefixes in the prefix list, only one prefix must match for a match to occur. The prefix list matching is effectively a logical OR operation.

Configuring Prefix List Filters

A prefix list filter allows you to apply prefix list qualifiers to a list of prefixes within a prefix list. The prefixes within the list are evaluated using the specified qualifiers. You can configure multiple prefix list filters under the same policy term.

To configure a prefix list filter, include the **prefix-list-filter** statement at the **[edit policy-options policy-statement *policy-name* from]** hierarchy level:

```
[edit policy-options policy-statement policy-name]
from {
  prefix-list-filter prefix-list-name match-type actions;
}
```

The ***prefix-list-name*** option is the name of the prefix list to be used for evaluation. You can specify only one prefix list.

The ***match-type*** option is the type of match to apply to the prefixes in the prefix list. It can be one of the match types listed in Table 20 on page 282.

The ***actions*** option is the action to take if the prefix list matches. It can be one or more of the actions listed in "Configuring Flow Control Actions" on page 56 and "Configuring Actions That Manipulate Route Characteristics" on page 57.

Table 20: Route List Match Types for a Prefix List Filter

Match Type	Match Condition
exact	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is equal to the route's prefix length.
longer	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is greater than the route's prefix length.
orlonger	The route shares the same most-significant bits (described by ***prefix-length***), and ***prefix-length*** is equal to or greater than the route's prefix length.

Related Documentation

- Example: Configuring Routing Policy Prefix Lists on page 282
- Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List on page 687

Example: Configuring Routing Policy Prefix Lists

In Junos OS, prefix lists provide one method of defining a set of routes. Junos OS provides other methods of accomplishing the same task, such as route filters. A prefix list is a listing of IP prefixes that represent a set of routes that are used as match criteria in an applied policy. Such a list might be useful for representing a list of customer routes in your autonomous system (AS). A prefix list is given a name and is configured within the **[edit policy-options]** configuration hierarchy.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

Prefix lists are similar to a list of route filters. The functional difference between route filters and prefix lists is that you cannot specify a range using a prefix list. You can simulate a range using a prefix list by including additional prefixes in the list, or by using two prefix lists, one shorter and one longer, setting one to accept and the other to reject. You can also filter a prefix list using the **prefix-list-filter** match condition. Your choices are **exact**, **longer**, and **orlonger**.

The benefit of a prefix list over a list of route filters is seen when the prefixes are referenced in several different locations. For instance, a prefix list can be referenced in a BGP import policy, an export policy, an RPF policy, in firewall filters, in loopback filters, in setting a multicast scope, and so on.

When your list of prefixes changes, rather than trying to remember the many different locations prefixes are configured, you can instead update the prefix list, changing the prefix one time instead of multiple times. This helps to reduce the likelihood of configuration errors, such as mistyping the address in a location or forgetting to update one or more locations.

Prefix lists also help when managing a large number of devices. You can write the various filters and policies as generically as possible, referencing prefix lists instead of specific IP addresses. The more complex logic in the filters and policies has to be written only one time, with minimal per-device and per-site customizations.

As shown in Figure 28 on page 285, each router in AS 64510 has customer routes. Device R1 assigns customer routes within the 172.16.1.0/24 subnet. Device R2 and Device R3 assign customer routes within the 172.16.2.0/24 and 172.16.3.0/24 subnets, respectively. Device R1 has been designated the central point in AS 64510 to maintain a complete list of customer routes. Device R1 has a prefix list called **customers**, as follows:

```
user@R1# show policy-options
prefix-list customers {
  172.16.1.16/28;
  172.16.1.32/28;
  172.16.1.48/28;
```



```
    172.16.1.64/28;
    172.16.2.16/28;
    172.16.2.32/28;
    172.16.2.48/28;
    172.16.2.64/28;
    172.16.3.16/28;
    172.16.3.32/28;
    172.16.3.48/28;
    172.16.3.64/28;
}
```

As you can see, the prefix list does not contain a match type for each route (as you would see with a route filter). This is an important point when using a prefix list in a policy. Routes match only if they exactly match one of the prefixes in the list. In other words, each route in the list must appear in the routing table exactly as it is configured in the prefix list.

You reference the prefix list as a match criterion within a policy like this:

```
user@R1# show policy-options
policy-statement customer-routes {
    term get-routes {
        from {
            prefix-list customers;
        }
        then accept;
    }
    term others {
        then reject;
    }
}
```

In this example, all the routes in the **customers** prefix list appear in the routing table on Device R1. Device R2 and Device R3 export to Device R1 static routes to their customers.

As previously mentioned, you can use the **prefix-list-filter** match condition with the **exact**, **longer**, or **orlonger** match type. This provides a way to avoid the prefix list exact-match limitation of prefix lists. For example:

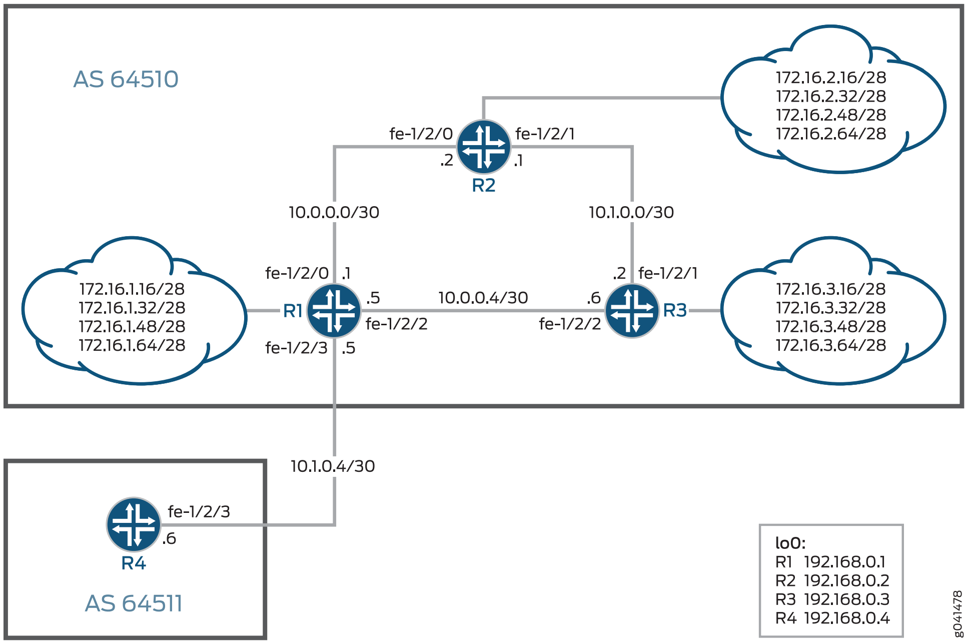
```
user@R1# show policy-options
policy-statement customer-routes {
    term get-routes {
        from {
            prefix-list-filter customers orlonger;
        }
        then accept;
    }
    term others {
        then reject;
    }
}
```

The example demonstrates the effects of both the **prefix-list** match condition and the **prefix-list-filter** match condition.

Topology

Figure 28 on page 285 shows the sample network.

Figure 28: BGP Topology for Policy Prefix Lists



"CLI Quick Configuration" on page 285 shows the configuration for all of the devices in Figure 28 on page 285.

The section "Step-by-Step Procedure" on page 287 describes the steps on Device R1.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 description to_R2
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/2 unit 0 description to_R3
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30
set interfaces fe-1/2/3 unit 0 description to_R4
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.1
set protocols bgp group int neighbor 192.168.0.2
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group to_64511 type external
set protocols bgp group to_64511 neighbor 10.1.0.6 peer-as 64511
set protocols bgp group to_64511 export customer-routes
```

```
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options prefix-list 64510-customers 172.16.1.16/28
set policy-options prefix-list 64510-customers 172.16.1.32/28
set policy-options prefix-list 64510-customers 172.16.1.48/28
set policy-options prefix-list 64510-customers 172.16.1.64/28
set policy-options prefix-list 64510-customers 172.16.2.16/28
set policy-options prefix-list 64510-customers 172.16.2.32/28
set policy-options prefix-list 64510-customers 172.16.2.48/28
set policy-options prefix-list 64510-customers 172.16.2.64/28
set policy-options prefix-list 64510-customers 172.16.3.16/28
set policy-options prefix-list 64510-customers 172.16.3.32/28
set policy-options prefix-list 64510-customers 172.16.3.48/28
set policy-options prefix-list 64510-customers 172.16.3.64/28
set policy-options policy-statement customer-routes term get-routes from prefix-list
   64510-customers
set policy-options policy-statement customer-routes term get-routes then accept
set policy-options policy-statement customer-routes term others then reject
set routing-options static route 172.16.1.16/28 discard
set routing-options static route 172.16.1.32/28 discard
set routing-options static route 172.16.1.48/28 discard
set routing-options static route 172.16.1.64/28 discard
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device R2

```
set interfaces fe-1/2/0 unit 0 description to_R1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/2 unit 0 description to_R3
set interfaces fe-1/2/2 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int neighbor 192.168.0.1 export send-static
set protocols bgp group int neighbor 192.168.0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.16.2.16/28 discard
set routing-options static route 172.16.2.32/28 discard
set routing-options static route 172.16.2.48/28 discard
set routing-options static route 172.16.2.64/28 discard
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/1 unit 0 description to_R2
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/2 unit 0 description to_R1
set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.3
```



```
set protocols bgp group int neighbor 192.168.0.1 export send-static
set protocols bgp group int neighbor 192.168.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.16.3.16/28 discard
set routing-options static route 172.16.3.32/28 discard
set routing-options static route 172.16.3.48/28 discard
set routing-options static route 172.16.3.64/28 discard
set routing-options static route 172.16.3.1/32 discard
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64510
```

Device R4

```
set interfaces fe-1/2/3 unit 0 description to_R1
set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.6/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.1.0.5
set routing-options autonomous-system 64511
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set interfaces fe-1/2/0 unit 0 description to_R2
user@R1# set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30

user@R1# set interfaces fe-1/2/2 unit 0 description to_R3
user@R1# set interfaces fe-1/2/2 unit 0 family inet address 10.0.0.5/30

user@R1# set interfaces fe-1/2/3 unit 0 description to_R4
user@R1# set interfaces fe-1/2/3 unit 0 family inet address 10.1.0.5/30

user@R1# set interfaces lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure the internal BGP (IBGP) connections to Device R2 and Device R3.

```
[edit protocols bgp group int]
user@R1# set type internal
user@R1# set local-address 192.168.0.1
user@R1# set neighbor 192.168.0.2
user@R1# set neighbor 192.168.0.3
```

3. Configure the EBGP connection to Device R4.

```
[edit protocols bgp group to_64511]
user@R1# set type external
user@R1# set neighbor 10.1.0.6 peer-as 64511
user@R1# set export customer-routes
```

4. Configure OSPF connections to Device R2 and Device R3.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.0
user@R1# set interface fe-1/2/2.0
user@R1# set interface lo0.0 passive
```

5. Configure the prefix list.

```
[edit policy-options prefix-list 64510-customers]
user@R1# set 172.16.1.16/28
user@R1# set 172.16.1.32/28
user@R1# set 172.16.1.48/28
user@R1# set 172.16.1.64/28
user@R1# set 172.16.2.16/28
user@R1# set 172.16.2.32/28
user@R1# set 172.16.2.48/28
user@R1# set 172.16.2.64/28
user@R1# set 172.16.3.16/28
user@R1# set 172.16.3.32/28
user@R1# set 172.16.3.48/28
user@R1# set 172.16.3.64/28
```

6. Configure the routing policy that references the prefix list as a match criterion.

```
[edit policy-options policy-statement customer-routes term get-routes]
user@R1# set from prefix-list 64510-customers
user@R1# set then accept

[edit policy-options policy-statement customer-routes term others]
user@R1# set then reject
```

7. Configure the static route to the 172.16.5.0/24 network.

```
[edit routing-options static]
user@R1# set route 172.16.1.16/28 discard
user@R1# set route 172.16.1.32/28 discard
user@R1# set route 172.16.1.48/28 discard
user@R1# set route 172.16.1.64/28 discard
```

8. Configure the autonomous system (AS) number and router ID.

```
[edit routing-options]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 64510
```


Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
fe-1/2/0 {
   unit 0 {
      description to_R2;
      family inet {
         address 10.0.0.1/30;
      }
   }
}
fe-1/2/2 {
   unit 0 {
      description to_R3;
      family inet {
         address 10.0.0.5/30;
      }
   }
}
fe-1/2/3 {
   unit 0 {
      description to_R4;
      family inet {
         address 10.1.0.5/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.1/32;
      }
   }
}
```

```
user@R1# show protocols
bgp {
   group int {
      type internal;
      local-address 192.168.0.1;
      neighbor 192.168.0.2;
      neighbor 192.168.0.3;
   }
   group to_64511 {
      type external;
      export customer-routes;
      neighbor 10.1.0.6 {
         peer-as 64511;
      }
   }
}
ospf {
   area 0.0.0.0 {
```

```
        interface fe-1/2/0.0;
        interface fe-1/2/2.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

user@R1# **show policy-options**

```
prefix-list 64510-customers {
    172.16.1.16/28;
    172.16.1.32/28;
    172.16.1.48/28;
    172.16.1.64/28;
    172.16.2.16/28;
    172.16.2.32/28;
    172.16.2.48/28;
    172.16.2.64/28;
    172.16.3.16/28;
    172.16.3.32/28;
    172.16.3.48/28;
    172.16.3.64/28;
}
policy-statement customer-routes {
    term get-routes {
        from {
            prefix-list 64510-customers;
        }
        then accept;
    }
    term others {
        then reject;
    }
}
```

user@R1# **show routing-options**

```
static {
    route 172.16.1.16/28 discard;
    route 172.16.1.32/28 discard;
    route 172.16.1.48/28 discard;
    route 172.16.1.64/28 discard;
}
router-id 192.168.0.1;
autonomous-system 64510;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Verifying the Routes on Device R1 on page 291
- Verifying the Route Advertisement to Device R4 on page 291
- Experimenting with the prefix-list-filter Statement on page 292

Verifying the Routes on Device R1

Purpose On Device R1, check the routes in the routing table.

Action

```
user@R1> show route terse 172.16/16

inet.0: 26 destinations, 26 routes (26 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 172.16.1.16/28     S   5                        Discard
* ? 172.16.1.32/28     S   5                        Discard
* ? 172.16.1.48/28     S   5                        Discard
* ? 172.16.1.64/28     S   5                        Discard
* ? 172.16.2.1/32      B 170        100                             I
  unverified                                       >10.0.0.2
* ? 172.16.2.16/28     B 170        100                             I
  unverified                                       >10.0.0.2
* ? 172.16.2.32/28     B 170        100                             I
  unverified                                       >10.0.0.2
* ? 172.16.2.48/28     B 170        100                             I
  unverified                                       >10.0.0.2
* ? 172.16.2.64/28     B 170        100                             I
  unverified                                       >10.0.0.2
* ? 172.16.2.96/32     B 170        100                             I
  unverified                                       >10.0.0.2
* ? 172.16.3.1/32      B 170        100                             I
  unverified                                       >10.0.0.6
* ? 172.16.3.16/28     B 170        100                             I
  unverified                                       >10.0.0.6
* ? 172.16.3.32/28     B 170        100                             I
  unverified                                       >10.0.0.6
* ? 172.16.3.48/28     B 170        100                             I
  unverified                                       >10.0.0.6
* ? 172.16.3.64/28     B 170        100                             I
  unverified                                       >10.0.0.6
```

Meaning Device R1 has learned its own static routes (S) and the BGP routes from Devices R2 and R3 (B).

Verifying the Route Advertisement to Device R4

Purpose On Device R1, make sure that the customer routes are advertised to Device R4.

Action

user@R1> **show route advertising-protocol bgp 10.1.0.6**

```
inet.0: 26 destinations, 26 routes (26 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```

Meaning As expected, only the routes from the customer prefix list are advertised to Device R4.

Experimenting with the prefix-list-filter Statement

Purpose See what can happen when you use **prefix-list-filter** instead of **prefix-list**.

Action

1. On Device R2, add a static route that is longer than one of the existing static routes.

   [edit routing-options static route]
   user@R2# **set 172.16.2.65/32 discard**
   user@R2# **commit**

2. On Device R1, deactivate the prefix list and configure a prefix list filter with the **orlonger** match type.

   [edit policy-options policy-statement customer-routes term get-routes]
   user@R1# **deactivate from prefix-list 64510-customers**
   user@R1# **set from prefix-list-filter 64510-customers orlonger**
   user@R1# **commit**

3. On Device R1, check which routes are advertised to Device R4.

   user@R1> **show route advertising-protocol bgp 10.1.0.6**

```
inet.0: 27 destinations, 27 routes (27 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.16/28          Self                                    I
* 172.16.1.32/28          Self                                    I
* 172.16.1.48/28          Self                                    I
* 172.16.1.64/28          Self                                    I
* 172.16.2.16/28          Self                                    I
* 172.16.2.32/28          Self                                    I
* 172.16.2.48/28          Self                                    I
* 172.16.2.64/28          Self                                    I
* 172.16.2.65/32          Self                                    I
* 172.16.3.16/28          Self                                    I
* 172.16.3.32/28          Self                                    I
* 172.16.3.48/28          Self                                    I
* 172.16.3.64/28          Self                                    I
```


Meaning As expected, Device R1 is now advertising the 172.16.2.65/32 route to Device R4, even though 172.16.2.65/32 is not in the prefix list.

Related Documentation

- Understanding Prefix Lists for Use in Routing Policy Match Conditions on page 279
- Example: Configuring Policy Chains and Route Filters on page 186
- Example: Configuring a Policy Subroutine on page 203

Example: Configuring the Priority for Route Prefixes in the RPD Infrastructure

This example shows how to configure priority for route prefixes in the RPD infrastructure for the OSPF, LDP, and BGP protocols.

Requirements

This example uses the following hardware and software components:

- Three routers in a combination of ACX Series, M Series, MX Series, PTX Series, and T Series.
- Junos OS Release 16.1 or later running on all devices.

Before you begin:

1. Configure the device interfaces.
2. Configure the following protocols:
   - BGP
   - MPLS
   - OSPF
   - LDP

Overview

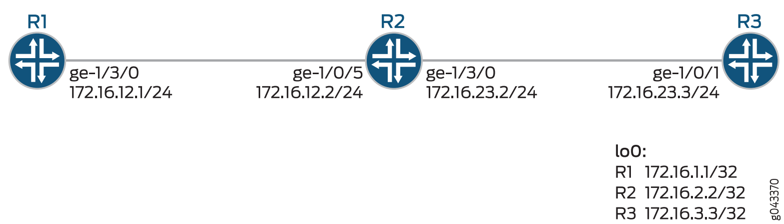
In a network with a large number of routes, it is sometimes important to control the order in which routes get updated for better convergence and to provide differentiated services. Prefix prioritization helps users to prioritize certain routes/prefixes over others and have control over the order in which routes get updated in the RIB (routing table) and the FIB (forwarding table). In Junos OS Release 16.1 and later, you can control the order in which the routes get updated from LDP/OSPF to rpd and rpd to kernel. You can specify a priority of **high** or **low** through the existing import policy in the protocols. In the event of a topology change, high priority prefixes are updated in the routing table first, followed by low priority prefixes. In general, routes that are not explicitly assigned a priority are treated as medium priority. Within the same priority level, routes will continue to be updated in lexicographic order.

In this example, the routing device is in area 0.0.0.0, with interface ge-1/3/0 connected to the neighboring device. You configure three import routing policies: next-hop-self, ospf-prio, and prio_for_bgp. The routing policy next-hop-self accepts routes from BGP. For the OSPF routing policy, routes matching 172.16.25.3/32 are installed first because they have a priority of high. LDP imports routes from OSPF. For BGP prioritization, routes matching 172.16.50.1/32 are installed first because they have a priority of high. Routes associated with these prefixes are installed in the routing table in the order of the specified priority of the prefix.

Topology

Figure 29 on page 294 shows the sample topology.

Figure 29: Priority for Route Prefixes in the rpd Infrastructure



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from the configuration mode.

R1

```
set interfaces ge-1/3/0 unit 0 family inet address 172.16.12.1/24
set interfaces ge-1/3/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.25.1/32
set protocols mpls interface ge-1/3/0.0
set protocols bgp group prio_internal type internal
set protocols bgp group prio_internal local-address 172.16.25.1
set protocols bgp group prio_internal import prio_for_bgp
set protocols bgp group prio_internal neighbor 172.16.25.3 family inet unicast
set protocols bgp group prio_internal neighbor 172.16.25.3 export next-hop-self
sset protocols ospf import ospf_prio
set protocols ospf area 0.0.0.0 interface ge-1/3/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/3/0.0
set protocols ldp interface lo0.0
set policy-options policy-statement next-hop-self term nhself from protocol bgp
set policy-options policy-statement next-hop-self term nhself then next-hop self
set policy-options policy-statement next-hop-self term nhself then accept
set policy-options policy-statement ospf_prio term ospf_ldp from protocol ospf
```



```
set policy-options policy-statement ospf_prio term ospf_ldp from route-filter 172.16.25.3/32
  exact
set policy-options policy-statement ospf_prio term ospf_ldp then priority high
set policy-options policy-statement ospf_prio term ospf_ldp then accept
set policy-options policy-statement prio_for_bgp term bgp_prio from protocol bgp
set policy-options policy-statement prio_for_bgp term bgp_prio from route-filter
  172.16.50.1/32 exact
set policy-options policy-statement prio_for_bgp term bgp_prio then priority high
set routing-options nonstop-routing
set routing-options router-id 172.16.25.1
set routing-options autonomous-system 2525
```

R2

```
set interfaces ge-1/0/5 unit 0 family inet address 172.16.12.2/24
set interfaces ge-1/0/5 unit 0 family mpls
set interfaces ge-1/3/0 unit 0 family inet address 172.16.23.2/24
set interfaces ge-1/3/0 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.25.2/32
set protocols mpls interface ge-1/0/5.0
set protocols mpls interface ge-1/3/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/5.0
set protocols ospf area 0.0.0.0 interface ge-1/3/0.0
set protocols ldp interface ge-1/0/5.0
set protocols ldp interface ge-1/3/0.0
set protocols ldp interface lo0.0
set routing-options nonstop-routing
set routing-options router-id 172.16.25.2
set routing-options autonomous-system 2525
```

R3

```
set interfaces ge-1/0/1 unit 0 family inet address 172.16.23.3/24
set interfaces ge-1/0/1 unit 0 family mpls
set interfaces lo0 unit 0 family inet address 172.16.25.3/32
set protocols mpls interface ge-1/0/1.0
set protocols bgp group prio_internal type internal
set protocols bgp group prio_internal local-address 172.16.25.3
set protocols bgp group prio_internal neighbor 172.16.25.1 family inet unicast
set protocols bgp group prio_internal neighbor 172.16.25.1 export next-hop-self
set protocols bgp group prio_internal neighbor 172.16.25.1 export static_to_bgp
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/1.0
set protocols ldp interface ge-1/0/1.0
set protocols ldp interface lo0.0
set policy-options policy-statement next-hop-self term nhself from protocol bgp
set policy-options policy-statement next-hop-self term nhself then next-hop self
set policy-options policy-statement next-hop-self term nhself then accept
set policy-options policy-statement static_to_bgp term s_to_b from protocol static
set policy-options policy-statement static_to_bgp term s_to_b from route-filter
  172.16.50.1/32 exact
set policy-options policy-statement static_to_bgp term s_to_b from route-filter
  172.16.50.2/32 exact
set policy-options policy-statement static_to_bgp term s_to_b then accept
set routing-options nonstop-routing
set routing-options static route 172.16.50.1/32 receive
set routing-options static route 172.16.50.2/32 receive
```

```
set routing-options router-id 172.16.25.3
set routing-options autonomous-system 2525
```

Configuring Device R1

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set interfaces ge-1/3/0 unit 0 family inet address 172.16.12.1/24
user@R1# set interfaces ge-1/3/0 unit 0 family mpls
user@R1# set interfaces lo0 unit 0 family inet address 172.16.25.1/32
```

2. Assign the loopback address to the device.

```
[edit lo0 unit 0 family]
user@R1# set address 172.16.25.1/32
```

3. Configure MPLS.

```
[edit protocols]
user@R1# set protocols mpls interface ge-1/3/0.0
```

4. Configure the router ID and autonomous system of Router R1.

```
[edit routing-options]
user@R1# set router-id 172.16.7.7
user@R1# set autonomous-system 100
```

5. Enable OSPF on the interfaces of Router R1.

```
[edit protocols]
user@R1# set protocols ospf import ospf_prio
user@R1# set protocols ospf area 0.0.0.0 interface ge-1/3/0.0
user@R1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

6. Configure LDP protocols on the interfaces.

```
[edit protocols]
user@R1# set protocols ldp interface ge-1/3/0.0
user@R1# set protocols ldp interface lo0.0
```

7. Configure BGP.

```
[edit protocols]
user@R1# set protocols bgp group prio_internal type internal
user@R1# set protocols bgp group prio_internal local-address 172.16.25.1
user@R1# set protocols bgp group prio_internal import prio_for_bgp
```

```
user@R1# set protocols bgp group prio_internal neighbor 172.16.25.3 family inet
  unicast
user@R1# set protocols bgp group prio_internal neighbor 172.16.25.3 export
  next-hop-self
```

8. Configure the policy options to prioritize the routes. The policy next-hop-self accepts routes from BGP. You configure three import routing policies: next-hop-self, ospf-prio, and prio_for_bgp. The routing policy next-hop-self accepts routes from BGP. For the ospf-prio routing policy, routes matching 172.16.25.3/32 are installed first because they have a priority of high. LDP imports routes from OSPF. For prio_for_bgp policy, routes matching 172.16.50.1/32 are installed first because they have a priority of high.

```
[edit policy-options policy-statement]
user@R1# set policy-options policy-statement next-hop-self term nhself from
  protocol bgp
user@R1# set policy-options policy-statement next-hop-self term nhself then
  next-hop self
user@R1# set policy-options policy-statement next-hop-self term nhself then accept

user@R1# set policy-options policy-statement ospf_prio term ospf_ldp from protocol
  ospf
user@R1# set policy-options policy-statement ospf_prio term ospf_ldp from
  route-filter 172.16.25.3/32 exact
set policy-options policy-statement ospf_prio term ospf_ldp then priority high

set policy-options policy-statement ospf_prio term ospf_ldp then accept
set policy-options policy-statement prio_for_bgp term bgp_prio from protocol bgp
set policy-options policy-statement prio_for_bgp term bgp_prio from route-filter
  172.16.50.1/32 exact
set policy-options policy-statement prio_for_bgp term bgp_prio then priority high
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show routing-options**, and **show policy-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@R1# show interfaces
ge-1/3/0 {
  unit 0 {
    family inet {
      address 172.16.12.1/24;
    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    family inet {
```

```
            address address 172.16.25.1/32;
        }
    }
}

[edit]
user@R1# show protocols
mpls {
    interface ge-1/3/0.0;
}
bgp {
    group prio_internal {
        type internal;
        local-address 172.16.25.1;
    import prio_for_bgp
        neighbor 172.16.25.3 {
            family inet {
                unicast;
                }
                export next-hop-self;
                }
                }
            }
        ospf {
            import ospf_prio;
            area 0.0.0.0 {
                interface ge-1/3/0.0;
                interface lo0.0 {
                    passive;
                    }
            }
        }
        ldp {
        interface ge-1/3/0.0;
        interface lo0.0;
        }
    }

[edit]
user@R1# show routing-options
nonstop-routing;
router-id 172.16.25.1;
autonomous-system 2525;

[edit]
user@R1# show policy-options
policy-statement next-hop-self {
    term nhself {
        from protocol bgp;
        then {
            next-hop self;
                accept;
            }
    }
}
policy-statement ospf_prio {
    term ospf_ldp {
```



```
          from {
            protocol ospf;
            route-filter 172.16.25.3/32 exact;
            }
          then {
              priority high;
              accept;
              }
            }
      }
    policy-statement prio_for_bgp {
      term bgp_prio {
          from {
            protocol bgp;
          route-filter 172.16.50.1/32 exact;
            }
            then priority high;
      }
    }
```

If you are done configuring the device, enter **commit** from the configuration mode.

Verification

Confirm that the configuration is working properly.

- Verifying the Priority for OSPF Routes on page 299
- Verifying the Priority for LDP Routes on page 300
- Verifying the Priority for BGP Routes on page 302

Verifying the Priority for OSPF Routes

Purpose Verify that the priority is set for the expected route in OSPF.

Action On Device R1, from operational mode, run the **show ospf route 172.16.25.3/32 extensive** command. A priority of high is applied to OSPF route 172.16.25.3.

```
user@R1> show ospf route 172.16.25.3/32 extensive

Topology default Route Table:

Prefix             Path  Route      NH       Metric NextHop       Nexthop
                   Type  Type       Type            Interface     Address/LSP
172.16.25.3          Intra Router     IP            2 ge-1/3/0.0    172.16.12.2
  area 0.0.0.0, origin 172.16.25.3, optional-capability 0x0
172.16.25.3/32       Intra Network    IP            2 ge-1/3/0.0    172.16.12.2
  area 0.0.0.0, origin 172.16.25.3, priority high
```

Meaning The output shows priority **high** is applied for OSPF route 172.16.25.3.

Verifying the Priority for LDP Routes

Purpose Verify if LDP inherits from OSPF.

Action From operational mode, enter the **show route 172.16.25.3** command to verify if LDP has inherited routes from OSPF.

```
user@R1> show route 172.16.25.3

  inet.0: 24 destinations, 24 routes (24 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.25.3/32       *[OSPF/10] 00:10:27, metric 2
                    > to 172.16.25.2 via ge-1/3/0.0

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.25.3/32       *[LDP/9] 00:10:24, metric 1
                    > to 172.16.25.2 via ge-1/3/0.0, Push 299824
```

From operational mode, enter the **show route 172.16.25.3 extensive** command to verify if LDP has inherited priority.

```
user@R1> show route 172.16.25.3 extensive
inet.0: 24 destinations, 24 routes (24 active, 0 holddown, 0 hidden)
172.16.25.3/32 (1 entry, 1 announced)
        State:<Flashall>
TSI:
KRT in-kernel 172.16.25.3/32 -> {172.16.12.2}
        *OSPF   Preference: 10
                Next hop type: Router, Next hop index: 549
                Address: 0xa463390
                Next-hop reference count: 6
                Next hop: 172.16.12.2 via ge-1/3/0.0, selected
                Session Id: 0x0
                State:<Active Int HighPriority>
                Local AS:  2525
                Age: 10:43      Metric: 2
                Validation State: unverified
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (4): 0-KRT 4-LDP 6-Resolve tree 2
7-Resolve_IGP_FRR task
                AS path: I


inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

172.16.25.3/32 (1 entry, 1 announced)
        State:<Flashall>
LDP    Preference: 9
                Next hop type: Router, Next hop index: 582
                Address: 0xa477810
                Next-hop reference count: 12
                Next hop: 172.16.12.2 via ge-1/3/0.0, selected
                Label operation: Push 299824
                Label TTL action: prop-ttl
                Load balance label: Label 299824: None;
                Label element ptr: 0xa17ad00
                Label parent element ptr: 0x0
                Label element references: 1
```



```
                Label element child references: 0
                Label element lsp id: 0
                Session Id: 0x0
                State:<Active Int HighPriority>
                Local AS:  2525
                Age: 10:40      Metric: 1
                Validation State: unverified
                Task: LDP
                Announcement bits (3): 2-Resolve tree 1 3-Resolve tree 2
4-Resolve_IGP_FRR task
                AS path: I
```

Meaning The output shows that LDP inherits priority **high** for route 172.16.25.3 from OSPF.

Verifying the Priority for BGP Routes

Purpose Verify that priority is set for the expected route in BGP.

Action On Device R1, from operational mode, run the **show route protocol bgp** command to display the routes learned from BGP.

```
user@R1> show route protocol bgp

inet.0: 24 destinations, 24 routes (24 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.50.1/32      *[BGP/170] 00:11:24, localpref 100, from 172.16.25.3
                      AS path: I, validation-state: unverified
                    > to 172.16.12.2 via ge-1/3/0.0, Push 299824
172.16.50.2/32      *[BGP/170] 00:11:24, localpref 100, from 172.16.25.3
                      AS path: I, validation-state: unverified
                    > to 172.16.12.2 via ge-1/3/0.0, Push 299824

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

mpls.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
```

On Device R1, from operational mode, run the **show route 172.16.50.1 extensive** command. High priority is applied for BGP route 172.16.50.1.

```
user@R1> show route 172.16.50.1 extensive

inet.0: 24 destinations, 24 routes (24 active, 0 holddown, 0 hidden)
172.16.50.1/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 172.16.50.1/32 -> {indirect(1048574)}
        *BGP    Preference: 170/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa487b10
                Next-hop reference count: 4
                Source: 172.16.25.3
                Next hop type: Router, Next hop index: 582
                Next hop: 172.16.12.2 via ge-1/3/0.0, selected
                Label operation: Push 299824
```



```
Label TTL action: prop-ttl
Load balance label: Label 299824: None;
Label element ptr: 0xa17ad00
Label parent element ptr: 0x0
Label element references: 1
Label element child references: 0
Label element lsp id: 0
Session Id: 0x0
Protocol next hop: 172.16.25.3
Indirect next hop: 0xa4a9800 1048574 INH Session ID: 0x0
State: <Active Int Ext HighPriority>
Local AS:  2525 Peer AS:  2525
Age: 11:49      Metric2: 1
Validation State: unverified
Task: BGP_2525.172.16.25.3
Announcement bits (2): 0-KRT 6-Resolve tree 2
AS path: I (Atomic)
Accepted
Localpref: 100
Router ID: 172.16.25.3
Indirect next hops: 1
        Protocol next hop: 172.16.25.3 Metric: 1
        Indirect next hop: 0xa4a9800 1048574 INH Session ID: 0x0

        Indirect path forwarding next hops: 1
                Next hop type: Router
                Next hop: 172.16.12.2 via ge-1/3/0.0
                Session Id: 0x0
        172.16.25.3/32 Originating RIB: inet.3
          Metric: 1                       Node path count: 1
          Forwarding nexthops: 1
                Nexthop: 172.16.12.2 via ge-1/3/0.0
```

Meaning The output shows that priority **high** is applied for BGP route 172.16.50.1.

Related Documentation

- Prefix Prioritization Overview on page 12
- Configuring Priority for Route Prefixes in RPD Infrastructure on page 304

Configuring Priority for Route Prefixes in RPD Infrastructure

Prefix prioritization helps users to prioritize certain routes or prefixes for better convergence and to provide differentiated services. In a network with a large number of routes, it is sometimes important to control the order in which routes get updated due to changes in the network topology. At a system level, Junos OS implements reasonable defaults based on heuristics to determine the order in which routes get updated. However, the default behavior is not always optimal. Prefix prioritization provides the user the ability to control the order in which the routes get updated from LDP or OSPF to rpd, and rpd to kernel. The Junos OS policy language is extended to let the user set relative priority (high and low) for prefixes through the existing import policy in protocols. Based on the tagged priority, the routes are placed in different priority queues. In the event of a topology change, high priority prefixes are updated in the routing table first, followed by low priority prefixes. Within the same priority level, routes will continue to be updated in lexicographic order. Routes that are not explicitly assigned a priority are treated as medium priority.

Before you begin to configure prefix prioritization in rpd for protocols such as OSPF, LDP, and BGP:

- Configure the router interfaces.
- Configure MPLS.
- Configure the OSPF, BGP, and LDP protocols.

To configure the priority **high** for the OSPF protocol:

1. Configure the policy term.

   [edit policy-options policy-statement *policy-name*]
   user@host# set term *term-name*

   For example:

   [edit policy-options policy-statement ospf-prio]
   user@host# **set term t1**

2. Configure the policy term to accept routes from OSPF.

   [edit policy-options policy-statement ospf-prio term t1]
   user@host# **set from protocol ospf**

3. Specify the desired route as a match condition for which you want to set priority **high**.

   [edit policy-options policy-statement ospf-prio term t1]
   user@host# set from route-filter *destination-prefix match-type*

   For example:

   [edit policy-options policy-statement ospf-prio term t1]
   user@host# **set from route-filter 172.16.25.3/32 exact**

4. Specify that the route is to be accepted and set priority **high** for the route if the previous conditions are matched.


```
[edit policy-options policy-statement ospf-prio term t1]
user@host# set then priority high
user@host# set then accept
```

5. Verify the configuration.

```
[edit]
user@host# show policy-options
policy-statement ospf-prio {
   term t1 {
      from {
         route-filter 172.16.25.3/32 exact;
         }
      then {
            priority high;
            accept;
            }
         }
   }
```

LDP inherits from OSPF.

To configure priority **high** for LDP:

1. Configure the policy term that imports from OSPF.

```
[edit policy-options policy-statement policy-name]
user@host# set term term-name
```

   For example:

```
[edit policy-options policy-statement ospf-import]
user@host# set term ospf_ldp
```

2. Configure the term to accept routes and priority from OSPF.

```
[edit policy-options policy-statement ospf_import term ospf_ldp]
user@host# set from protocol ospf
user@host# set from route-filter destination-prefix match-type
```

   For example:

```
[edit policy-options policy-statement ospf_import term ospf_ldp]
user@host# set from protocol ospf
user@host# set from route-filter 172.16.25.3/32 exact
```

3. Verify the configuration.

```
[edit]
user@host# show policy-options
policy-statement ospf-import {
   term ospf_ldp {
      from {
         protocol ospf ;
         route-filter 172.16.25.3/32 exact;
         }
```

```
        then {
            priority high;
            accept;
            }
          }
      }
```

To configure the priority **high** for the BGP protocol:

1. Configure the policy term.

   ```
   [edit policy-options policy-statement policy-name]
   user@host# set term term-name
   ```

   For example:

   ```
   [edit policy-options policy-statement prio-for-bgp]
   user@host# set term bgp_prio
   ```

2. Specify the desired route as a match condition.

   ```
   [edit policy-options policy-statement prio-for-bgp term bgp_prio]
   user@host# set from protocol bgp
   user@host# set from route-filter destination-prefix match-type
   ```

   For example:

   ```
   [edit policy-options policy-statement prio-for-bgp term bgp_prio]
   user@host# set from protocol bgp
   user@host# set from route-filter 172.16.50.1/32 exact
   ```

3. Specify that the route is to be accepted and set the priority **high** for the route if the previous conditions are matched.

   ```
   [edit policy-options policy-statement prio-for-bgp term bgp_prio]
   user@host# set then priority high
   user@host# set then accept
   ```

4. Verify the configuration.

   ```
   policy-statement prio_for_bgp {
     term bgp_prio {
       from {
         protocol bgp;
         route-filter 172.16.50.1/32 exact;
         }
           then {
             priority high;
             accept;
             }
     }
   }
   ```


NOTE: For BGP, you can also configure priority based on the route-distinguisher (RD) value in case of L3VPN. For example, you can configure priority for BGP with route-distinguisher 51.51.51.51:111.

To configure priority for BGP based on the route-distinguisher (RD) value:

1. Configure the policy term.

   [edit policy-options policy-statement *policy-name*]
   user@host# set term *term-name*

   For example:

   [edit policy-options policy-statement prio-for-bgp]
   user@host# **set term bgp_prio**

2. Specify the desired route as a match condition.

   [edit policy-options policy-statement prio-for-bgp term bgp_prio]
   user@host# set from rib bgp.l3vpn.0
   user@host# set from route-filter *destination-prefix match-type*
   user@host# set from route-distinguisher *route-distinguisher value*

   For example:

   [edit policy-options policy-statement prio-for-bgp term bgp_prio]
   user@host# **set from rib bgp.l3vpn.0**
   user@host# **set from route-filter 172.16.1.1/32 exact**
   user@host# **set from route-distinguisher RD1**

3. Specify that the route is to be accepted and set the priority **high** for the route if the previous conditions are matched.

   [edit policy-options policy-statement prio-for-bgp term bgp_prio]
   user@host# set then priority high
   user@host# set then accept

4. Verify the configuration.

```
policy-statement prio_for_bgp {
  term bgp_prio {
    from {
      protocol rib bgp.l3vpn.0;
      route-filter 172.16.1.1/32 exact;
      route-distinguisher RD1;
      }
        then {
          priority high;
          accept;
          }
  }
}
```

NOTE: Low priority prefixes are installed only after the high priority prefixes in the routing table. You can also configure priority **low** similarly to priority **high** for the routes you want to set to low priority.

NOTE: Priority is applied only when routes are pushed from RIB to FIB. Therefore, you cannot modify the priority of routes that are already installed. Changing the priority of routes already installed does not make sense. If you try changing the priority of routes already installed, there is a show output difference:

```
user@R1> show route 172.16.25.3 extensive | match state
 State: <FlashAll>
             State:   <Active Int HighPriority>     <=== OSPF

        Validation State: unverified
        State: <FlashAll>
             State:   <Active Int>                  <=== LDP

        Validation State: unverified
```

As the route is already installed in FIB, LDP does not show the priority as High.

Restarting the routing daemon to remove the routes and adding it again reflects the proper priority from both the OSPF and LDP protocol perspective.

```
user@R1> restart routing
Routing protocols process signalled but still running, waiting 8 seconds
 more
Routing protocols process started, pid 4512
```

```
user@R1> show route 172.16.25.3 extensive |match state
 State: <FlashAll>
             State:   <Active Int HighPriority>     <=== OSPF

        Validation State: unverified
        State: <FlashAll>
             State:   <Active Int HighPriority>     <=== LDP

        Validation State: unverified
```

Related Documentation

- Prefix Prioritization Overview on page 12
- Example: Configuring the Priority for Route Prefixes in the RPD Infrastructure on page 293

CHAPTER 6

Configuring AS Paths as Match Conditions

Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions

A BGP *AS path* is the sequence of autonomous systems that network packets traverse to get to a specified router. AS numbers are assembled in a sequence that is read from right to left. For example, for a packet to reach a destination using a route with an AS path 5 4 3 2 1, the packet first traverses AS 1 and so on until it reaches AS 5. In this case, AS 5 is the last AS before the packet destination; it is the AS that the source of the packet would peer with.

When working with AS paths and routing policy match conditions, you can use regular expressions to locate routes. To do so, create one or more match conditions based on some or all of the AS path, and then include it in a routing policy.

The following sections describe AS path regular expressions and provide configuration examples.

Configuration of AS Path Regular Expressions

You can create a named AS path regular expression and then include it in a routing policy with the **as-path** match condition (described in "Routing Policy Match Conditions" on page 44). To create a named AS path regular expression, include the **as-path** statement:

```
[edit policy-options]
as-path name regular-expression;
```

To include the AS path regular expression in a routing policy, include the **as-path** match condition in the **from** statement.

Additionally, you can create a named AS path group made up of AS path regular expressions and then include it in a routing policy with the **as-path-group** match condition. To create a named AS path group, include the **as-path-group** statement.

```
[edit policy-options]
  as-path-group group-name {
  name [ regular-expressions ];
}
```

To include the AS path regular expressions within the AS path group in a routing policy, include the **as-path-group** match condition in the **from** statement.

NOTE: **You cannot include both of the as-path and as-path-group statements in the same policy term.**

NOTE: **You can include the names of multiple AS path regular expressions in the as-path match condition in the from statement. If you do this, only one AS path regular expression needs to match for a match to occur. The AS path regular expression matching is effectively a logical OR operation.**

The AS path name identifies the regular expression. It can contain letters, numbers, and hyphens (-), and can be up to 65,536 characters. To include spaces in the name, enclose the entire name in quotation marks (“ ”).

The regular expression is used to match all or portions of the AS path. It consists of two components, which you specify in the following format:

term <operator>

- *term*—Identifies an AS. You can specify it in one of the following ways:
  - AS number—The entire AS number composes one term. You cannot reference individual characters within an AS number, which differs from regular expressions as defined in POSIX 1003.2.
  - Wildcard character—Matches any single AS number. The wildcard character is a period (.). You can specify multiple wildcard characters.
  - AS path—A single AS number or a group of AS numbers enclosed in parentheses. Grouping the regular expression in this way allows you to perform a common operation on the group as a whole and to give the group precedence. The grouped path can itself include operators.

    In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS. You can configure a value in the range from 1 through 4,294,967,295.

- ***operator***—(Optional) An operator specifying how the term must match. Most operators describe how many times the term must be found to be considered a match (for example, any number of occurrences, or zero, or one occurrence). Table 21 on page 311 lists the regular expression operators supported for AS paths. You place operators immediately after ***term*** with no intervening space, except for the pipe (|) and dash (–) operators, which you place between two terms, and parentheses, with which you enclose terms.

You can specify one or more term–operator pairs in a single regular expression.

Table 22 on page 312 shows examples of how to define regular expressions to match AS paths.

Table 21: AS Path Regular Expression Operators

Operator	Match Definition
{*m*,*n*}	At least ***m*** and at most ***n*** repetitions of ***term***. Both ***m*** and ***n*** must be positive integers, and ***m*** must be smaller than ***n***.
{*m*}	Exactly ***m*** repetitions of ***term***. ***m*** must be a positive integer.
{*m*,}	***m*** or more repetitions of ***term***. ***m*** must be a positive integer.
*	Zero or more repetitions of ***term***. This is equivalent to {0,}.
+	One or more repetitions of ***term***. This is equivalent to {1,}.
?	Zero or one repetition of ***term***. This is equivalent to {0,1}.
|	One of two terms on either side of the pipe.
–	Between a starting and ending range, inclusive.
^	A character at the beginning of a community attribute regular expression. This character is added implicitly; therefore, the use of it is optional.
$	A character at the end of a community attribute regular expression. This character is added implicitly; therefore, the use of it is optional.
( )	A group of terms that are enclosed in the parentheses. Intervening space between the parentheses and the terms is ignored. If a set of parentheses is enclosed in quotation marks with no intervening space "()", it indicates a null path.
[ ]	Set of AS numbers. One AS number from the set must match. To specify the start and end of a range, use a hyphen (-). A carrot (^) may be used to indicate that it does not match a particular AS number in the set, for example [^123].

Table 22: Examples of AS Path Regular Expressions

AS Path to Match	Regular Expression	Sample Matches
AS path is 1234	1234	1234
Zero or more occurrences of AS number 1234	1234*	1234 1234 1234 1234 1234 1234 Null AS path
Zero or one occurrence of AS number 1234	1234? or 1234{0,1}	1234 Null AS path
One through four occurrences of AS number 1234	1234{1,4}	1234 1234 1234 1234 1234 1234 1234 1234 1234 1234
One through four occurrences of AS number 12, followed by one occurrence of AS number 34	12{1,4} 34	12 34 12 12 34 12 12 12 34 12 12 12 12 34
Range of AS numbers to match a single AS number	123–125	123 124 125
	[123–125]*	Null AS path 123 124 124 125 125 125 123 124 125 123
Path whose second AS number must be 56 or 78	(. 56) | (. 78) or . (56 | 78)	1234 56 1234 78 9876 56 3857 78

Table 22: Examples of AS Path Regular Expressions *(continued)*

AS Path to Match	Regular Expression	Sample Matches
Path whose second AS number might be 56 or 78	. (56 | 78) ?	1234 56 52 34 56 1234 1234 78 39 794 78 2
Path whose first AS number is 123 and second AS number is either 56 or 78	123 (56|78)	123 56 123 78
Path of any length, except nonexistent, whose second AS number can be anything, including nonexistent	. .* or . .{0,}	1234 1234 5678 1234 5 6 7 8
AS path is 1 2 3	1 2 3	1 2 3
One occurrence of the AS numbers 1 and 2, followed by one or more occurrences of the AS number 3	1 2 3+	1 2 3 1 2 3 3 1 2 3 3 3
One or more occurrences of AS number 1, followed by one or more occurrences of AS number 2, followed by one or more occurrences of AS number 3	1+ 2+ 3+	1 2 3 1 1 2 3 1 1 2 2 3 1 1 2 2 3 3
Path of any length that begins with AS numbers 4, 5, 6	4 5 6 .*	4 5 6 4 5 6 7 8 9
Path of any length that ends with AS numbers 4, 5, 6	.* 4 5 6	4 5 6 1 2 3 4 5 6 4 9 4 5 6
AS path 5, 12, or 18	5 | 12 | 18	5 12 18

Configuring a Null AS Path

You can use AS path regular expressions to create a null AS path that matches routes (prefixes) that have originated in your AS. These routes have not been advertised to your

AS by any external peers. To create a null AS path, use the parentheses operator enclosed in quotation marks with no intervening spaces:

```
"()"
```

In the following example, locally administered AS 2 is connected to AS 1 (10.2.2.6) and AS 3. AS 3 advertises its routes to AS 2, but the administrator for AS 2 does not want to advertise AS 3 routes to AS 1 and thereby allow transit traffic from AS 1 to AS 3 through AS 2. To prevent transit traffic, the export policy **only-my-routes** is applied to AS 1. It permits advertisement of routes from AS 2 to AS 1 but prevents advertisement of routes for AS 3 (or routes for any other connected AS) to AS 1:

```
[edit policy-options]
null-as "()";
policy-statement only-my-routes {
   term just-my-as {
      from {
         protocol bgp;
         as-path null-as;
      }
      then accept;
   }
   term nothing-else {
      then reject;
   }
}
protocol {
   bgp {
      neighbor 10.2.2.6 {
         export only-my-routes;
      }
   }
}
```

How AS Path Regular Expressions Are Evaluated

AS path regular expressions implement the extended (modern) regular expressions as defined in POSIX 1003.2. They are identical to the UNIX regular expressions with the following exceptions:

- The basic unit of matching in an AS path regular expression is the AS number and not an individual character.
- A regular expression matches a route only if the AS path in the route exactly matches ***regular-expression***. The equivalent UNIX regular expression is **^*regular-expression*$**. For example, the AS path regular expression **1234** is equivalent to the UNIX regular expression **^1234$**.
- You can specify a regular expression using wildcard operators.

Examples: Configuring AS Path Regular Expressions

Exactly match routes with the AS path 1234 56 78 9 and accept them:

```
[edit]
policy-options {
```



```
    as-path wellington "1234 56 78 9";
    policy-statement from-wellington {
        term term1 {
            from as-path wellington;
        }
            then {
                preference 200;
                accept;
            }
        term term2 {
            then reject;
        }
    }
}
```

Match alternate paths to an AS and accept them after modifying the preference:

```
[edit]
policy-options {
    as-path wellington-alternate "1234{1,6} (56|47)? (78|101|112)* 9+";
    policy-statement from-wellington {
        from as-path wellington-alternate;
        }
        then {
            preference 200;
            accept;
        }
    }
}
```

Match routes with an AS path of 123, 124, or 125 and accept them after modifying the preference:

```
[edit]
policy-options {
    as-path addison "123–125";
    policy-statement from-addison {
        from as-path addison;
        }
        then {
            preference 200;
            accept;
        }
    }
}
```

Related Documentation

- Example: Using AS Path Regular Expressions on page 316
- Example: Configuring a Routing Policy to Prepend the AS Path on page 326

Example: Using AS Path Regular Expressions

An autonomous system (AS) path is a route attribute used by BGP. The AS path is used both for route selection and to prevent potential routing loops. This example shows how to use regular expressions with AS path numbers to locate a set of routes.

- Requirements on page 316
- Overview on page 316
- Configuration on page 317
- Verification on page 323

Requirements

No special configuration beyond device initialization is required before configuring this example.

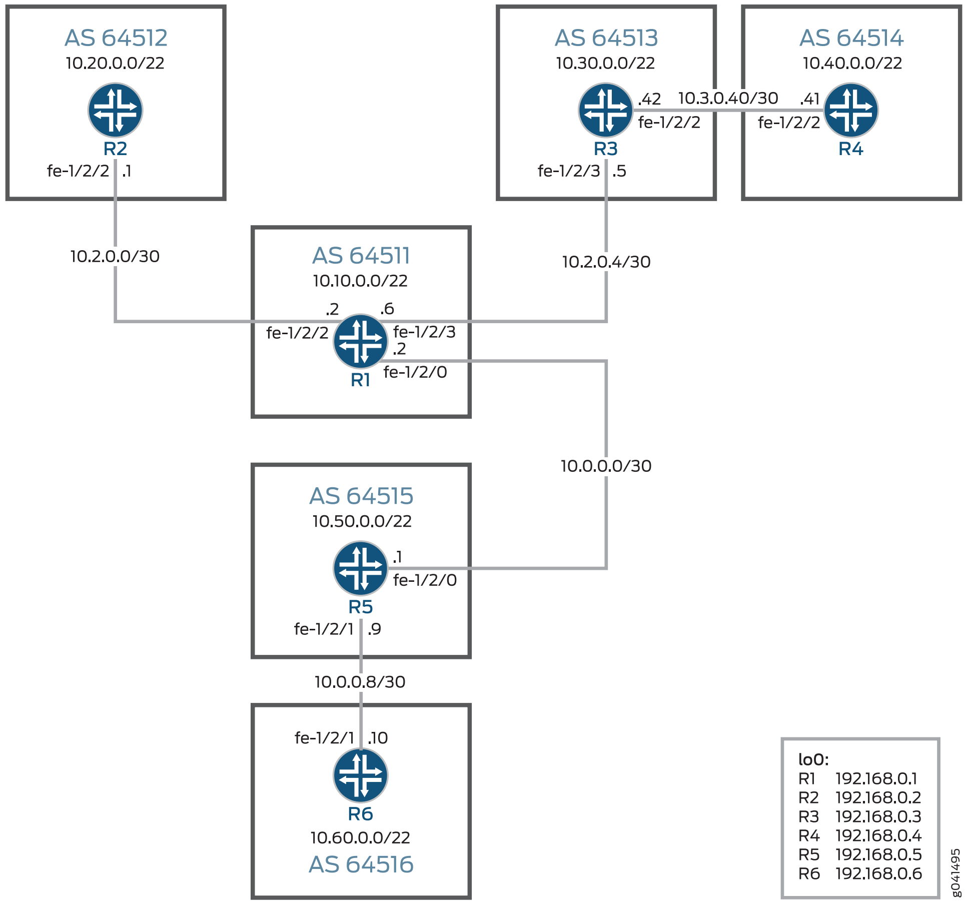
Overview

Figure 30 on page 317 shows several ASs connected through external BGP (EBGP) peering sessions. Each device is generating customer routes within its assigned address space.

Topology

Figure 30 on page 317 shows the sample network.

Figure 30: BGP Topology AS Regular Expressions



The administrators of AS 64516 want to reject all routes originating in AS 64513 and AS 64514. Two AS path regular expressions called **orig-in-64513** and **orig-in-64514** are created and referenced in a policy called **reject-some-routes**. The routing policy is then applied as an import policy on Device R6.

“CLI Quick Configuration” on page 317 shows the configuration for all of the devices in Figure 30 on page 317.

The section “Step-by-Step Procedure” on page 320 describes the steps on Device R2 and Device R6. “Verification” on page 323 shows how to use the **aspath-regex** option with the **show route** command on Device R2 to locate routes using regular expressions.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/2 unit 0 description to-R2
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.2/30
```

```
set interfaces fe-1/2/3 unit 0 description to-R3
set interfaces fe-1/2/3 unit 0 family inet address 10.2.0.6/30
set interfaces fe-1/2/0 unit 0 description to-R5
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp export send-static
set protocols bgp group 64512 type external
set protocols bgp group 64512 peer-as 64512
set protocols bgp group 64512 neighbor 10.2.0.1
set protocols bgp group 64513 type external
set protocols bgp group 64513 peer-as 64513
set protocols bgp group 64513 neighbor 10.2.0.5
set protocols bgp group 64515 type external
set protocols bgp group 64515 peer-as 64515
set protocols bgp group 64515 neighbor 10.0.0.1
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.10.1.0/24 reject
set routing-options static route 10.10.2.0/24 reject
set routing-options static route 10.10.3.0/24 reject
set routing-options autonomous-system 64511
```

Device R2

```
set interfaces fe-1/2/2 unit 0 description to-R1
set interfaces fe-1/2/2 unit 0 family inet address 10.2.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp export send-static
set protocols bgp group 64511 type external
set protocols bgp group 64511 peer-as 64511
set protocols bgp group 64511 neighbor 10.2.0.2
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.20.1.0/24 reject
set routing-options static route 10.20.2.0/24 reject
set routing-options static route 10.20.3.0/24 reject
set routing-options autonomous-system 64512
```

Device R3

```
set interfaces fe-1/2/3 unit 0 description to-R1
set interfaces fe-1/2/3 unit 0 family inet address 10.2.0.5/30
set interfaces fe-1/2/2 unit 0 description to-R4
set interfaces fe-1/2/2 unit 0 family inet address 10.3.0.42/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp export send-static
set protocols bgp group 64511 type external
set protocols bgp group 64511 peer-as 64511
set protocols bgp group 64511 neighbor 10.2.0.6
set protocols bgp group 64514 type external
set protocols bgp group 64514 peer-as 64514
set protocols bgp group 64514 neighbor 10.3.0.41
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.30.1.0/24 reject
set routing-options static route 10.30.2.0/24 reject
set routing-options static route 10.30.3.0/24 reject
set routing-options autonomous-system 64513
```


Device R4

```
set interfaces fe-1/2/2 unit 0 description to-R3
set interfaces fe-1/2/2 unit 0 family inet address 10.3.0.41/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp export send-static
set protocols bgp group 64513 type external
set protocols bgp group 64513 peer-as 64513
set protocols bgp group 64513 neighbor 10.3.0.42
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.40.1.0/24 reject
set routing-options static route 10.40.2.0/24 reject
set routing-options static route 10.40.3.0/24 reject
set routing-options autonomous-system 64514
```

Device R5

```
set interfaces fe-1/2/0 unit 0 description to-R1
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/1 unit 0 description to-R6
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.9/30
set interfaces lo0 unit 0 family inet address 192.168.0.5/32
set protocols bgp export send-static
set protocols bgp group 64511 type external
set protocols bgp group 64511 peer-as 64511
set protocols bgp group 64511 neighbor 10.0.0.2
set protocols bgp group 64516 type external
set protocols bgp group 64516 peer-as 64516
set protocols bgp group 64516 neighbor 10.0.0.10
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.50.1.0/24 reject
set routing-options static route 10.50.2.0/24 reject
set routing-options static route 10.50.3.0/24 reject
set routing-options autonomous-system 64515
```

Device R6

```
set interfaces fe-1/2/1 unit 0 description to-R5
set interfaces fe-1/2/1 unit 0 family inet address 10.0.0.10/30
set interfaces lo0 unit 0 family inet address 192.168.0.6/32
set protocols bgp export send-static
set protocols bgp group 64515 type external
set protocols bgp group 64515 import reject-some-routes
set protocols bgp group 64515 peer-as 64515
set protocols bgp group 64515 neighbor 10.0.0.9
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set policy-options policy-statement reject-some-routes term find-routes from as-path
  orig-in-64513
set policy-options policy-statement reject-some-routes term find-routes from as-path
  orig-in-64514
set policy-options policy-statement reject-some-routes term find-routes then reject
set policy-options as-path orig-in-64513 ".* 64513"
set policy-options as-path orig-in-64514 ".* 64514"
set routing-options static route 10.60.1.0/24 reject
set routing-options static route 10.60.2.0/24 reject
set routing-options static route 10.60.3.0/24 reject
set routing-options autonomous-system 64516
```


Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/2 unit 0 description to-R1
user@R2# set fe-1/2/2 unit 0 family inet address 10.2.0.1/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the EBGP connection to Device R1.

```
[edit protocols bgp]
user@R2# set export send-static
user@R2# set group 64511 type external
user@R2# set group 64511 peer-as 64511
user@R2# set group 64511 neighbor 10.2.0.2
```

3. Configure the routing policy.

```
[edit policy-options policy-statement send-static term 1]
user@R2# set from protocol static
user@R2# set then accept
```

4. Configure the static routes.

```
[edit routing-options static]
user@R2# set route 10.20.1.0/24 reject
user@R2# set route 10.20.2.0/24 reject
user@R2# set route 10.20.3.0/24 reject
```

5. Configure the AS number.

```
[edit routing-options]
user@R2# set autonomous-system 64512
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R6:

1. Configure the device interfaces.

```
[edit interfaces]
user@R6# set fe-1/2/1 unit 0 description to-R5
user@R6# set fe-1/2/1 unit 0 family inet address 10.0.0.10/30

user@R6# set lo0 unit 0 family inet address 192.168.0.6/32
```


2. Configure the EBGP connection to Device R5.

```
[edit protocols bgp]
user@R6# set export send-static
user@R6# set group 64515 type external
user@R6# set group 64515 import reject-some-routes
user@R6# set group 64515 peer-as 64515
user@R6# set group 64515 neighbor 10.0.0.9
```

3. Configure the routing policy that sends static routes.

```
[edit policy-options policy-statement send-static term 1]
user@R6# set from protocol static
user@R6# set then accept
```

4. Configure the routing policy that rejects certain routes.

```
[edit policy-options policy-statement reject-some-routes term find-routes]
user@R6# set from as-path orig-in-64513
user@R6# set from as-path orig-in-64514
user@R6# set then reject
```

```
[edit policy-options]
user@R6# set as-path orig-in-64513 ".* 64513"
user@R6# set as-path orig-in-64514 ".* 64514"
```

5. Configure the static routes.

```
[edit routing-options static]
user@R6# set route 10.60.1.0/24 reject
user@R6# set route 10.60.2.0/24 reject
user@R6# set route 10.60.3.0/24 reject
```

6. Configure the AS number.

```
[edit routing-options]
user@R6# set autonomous-system 64516
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device R2

```
user@R2# show interfaces
fe-1/2/0 {
  unit 0 {
    description to-R1;
    family inet {
      address 10.2.0.1/30;
    }
  }
}
lo0 {
```

```
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}

user@R2# show protocols
bgp {
    export send-static;
    group 64511 {
        type external;
        peer-as 64511;
        neighbor 10.2.0.2;
    }
}
user@R2# show policy-options
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
}
user@R2# show routing-options
static {
    route 10.20.1.0/24 reject;
    route 10.20.2.0/24 reject;
    route 10.20.3.0/24 reject;
}
autonomous-system 64512;
```

Device R6

```
user@R6# show interfaces
fe-1/2/0 {
    unit 0 {
        description to-R5;
        family inet {
            address 10.0.0.10/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.6/32;
        }
    }
}

user@R6# show protocols
bgp {
    export send-static;
    group 64515 {
        type external;
        import reject-some-routes;
```



```
        peer-as 64515;
        neighbor 10.0.0.9;
    }
}
```

```
user@R6# show policy-options
policy-statement reject-some-routes {
    term find-routes {
        from as-path [ orig-in-64513 orig-in-64514 ];
        then reject;
    }
}
policy-statement send-static {
    term 1 {
        from protocol static;
        then accept;
    }
}
as-path orig-in-64513 ".* 64513";
as-path orig-in-64514 ".* 64514";
```

```
user@R6# show routing-options
static {
    route 10.60.1.0/24 reject;
    route 10.60.2.0/24 reject;
    route 10.60.3.0/24 reject;
}
autonomous-system 64516;
```

If you are done configuring the devices, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Finding Routes on Device R2 on page 323
- Making Sure That Routes Are Excluded on Device R6 on page 325

Finding Routes on Device R2

Purpose On Device R2, use the **show route aspath-regex** command to locate routes using regular expressions.

Action Look for routes that are originated by Device R6 in AS 64516.

```
user@R2> show route terse aspath-regex ".* 64516"

inet.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 10.60.1.0/24       B 170        100                             64511 64515
64516 I
  unverified                                       >10.2.0.2
* ? 10.60.2.0/24       B 170        100                             64511 64515
```

```
64516 I
  unverified                                       >10.2.0.2
* ? 10.60.3.0/24       B 170        100                              64511 64515
64516 I
  unverified                                       >10.2.0.2
```

Look for routes that are originated in either AS 64514 or AS 64516.

`user@R2>` **show route terse aspath-regex ".* (64514|64516)"**

```
inet.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 10.40.1.0/24       B 170        100                              64511 64513
64514 I
  unverified                                       >10.2.0.2
* ? 10.40.2.0/24       B 170        100                              64511 64513
64514 I
  unverified                                       >10.2.0.2
* ? 10.40.3.0/24       B 170        100                              64511 64513
64514 I
  unverified                                       >10.2.0.2
* ? 10.60.1.0/24       B 170        100                              64511 64515
64516 I
  unverified                                       >10.2.0.2
* ? 10.60.2.0/24       B 170        100                              64511 64515
64516 I
  unverified                                       >10.2.0.2
* ? 10.60.3.0/24       B 170        100                              64511 64515
64516 I
  unverified                                       >10.2.0.2
```

Look for routes that use AS 64513 as a transit network.

`user@R2>` **show route terse aspath-regex ".* 64513 .+"**

```
inet.0: 21 destinations, 21 routes (21 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 10.40.1.0/24       B 170        100                              64511 64513
64514 I
  unverified                                       >10.2.0.2
* ? 10.40.2.0/24       B 170        100                              64511 64513
64514 I
  unverified                                       >10.2.0.2
* ? 10.40.3.0/24       B 170        100                              64511 64513
64514 I
  unverified
```

Meaning The output shows the routing table entries that match the specified AS path regular expressions.

Making Sure That Routes Are Excluded on Device R6

Purpose

On Device R6, use the show route and show route hidden commands to make sure that routes originating from AS 64513 and AS 64514 are excluded from Device R6's routing table.

Action

```
user@R6> show route 10.30.0/22
inet.0: 21 destinations, 21 routes (15 active, 0 holddown, 6 hidden)

user@R6> show route 10.40.0/22
inet.0: 21 destinations, 21 routes (15 active, 0 holddown, 6 hidden)

user@R6> show route hidden

inet.0: 21 destinations, 21 routes (15 active, 0 holddown, 6 hidden)
+ = Active Route, - = Last Active, * = Both

10.30.1.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 I, validation-state: unverified

                    > to 10.0.0.9 via fe-1/2/1.0
10.30.2.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 I, validation-state: unverified

                    > to 10.0.0.9 via fe-1/2/1.0
10.30.3.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 I, validation-state: unverified

                    > to 10.0.0.9 via fe-1/2/1.0
10.40.1.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 64514 I, validation-state:
unverified
                    > to 10.0.0.9 via fe-1/2/1.0
10.40.2.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 64514 I, validation-state:
unverified
                    > to 10.0.0.9 via fe-1/2/1.0
10.40.3.0/24        [BGP ] 02:24:47, localpref 100
                      AS path: 64515 64511 64513 64514 I, validation-state:
unverified
                    > to 10.0.0.9 via fe-1/2/1.0
```

Meaning

The output shows that the 10.30.0/22 and 10.40.0/22 routes are rejected on Device R6.

Related Documentation

- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions on page 309
- Example: Testing a Routing Policy with Complex Regular Expressions on page 544

Understanding Prepending AS Numbers to BGP AS Paths

You can *prepend* one or more autonomous system (AS) numbers at the beginning of an AS path. The AS numbers are added at the beginning of the path after the actual AS number from which the route originates has been added to the path. Prepending an AS path makes a shorter AS path look longer and therefore less preferable to BGP.

The BGP best path algorithm determines how the best path to an autonomous system (AS) is selected. The AS path length determines the best path when all of the following conditions are met:

- There are multiple potential routes to an AS.
- BGP has the lowest preference value (sometimes referred to as administrative distance) of the available routes.
- The local preferences of the available routes are equal.

When these conditions are met, the AS path length is used as the tie breaker in the best path algorithm. When two or more routes exist to reach a particular prefix, BGP prefers the route with the shortest AS Path length.

If you are an enterprise that has multihoming to one or more service providers, you might prefer that incoming traffic take a particular path to reach your network. Perhaps you have two connections, but one costs less than the other. Or you might have one fast connection and another, much slower connection that you only want to use as a backup if your primary connection is down. AS path prepending is an easy method that you can use to influence inbound routing to your AS.

In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS. In plain-number format, you can configure a value in the range from 1 through 4,294,967,295.

If you have a router that does not support 4-byte AS numbers in the AS path, the prependend AS number displayed in the AS path is the AS_TRANS number, AS 23456. To display the route details, use the *show route* command.

Related Documentation

- Example: Configuring a Routing Policy to Prepend the AS Path on page 326
- Example: Using AS Path Regular Expressions on page 316

- *Understanding BGP Path Selection*

Example: Configuring a Routing Policy to Prepend the AS Path

This example shows how to configure a routing policy to prepend the AS path.

- Requirements on page 327
- Overview on page 327

- Configuration on page 327
- Verification on page 328

Requirements

Before you begin, make sure your router interfaces and protocols are correctly configured.

Overview

In this example, you create a routing policy called prependpolicy1 and a term called prependterm1. The routing policy prepends the AS numbers 1111 to routes that are greater than or equal to 172.16.0.0/12, 192.168.0.0/16, and 10.0.0.0/8. The policy is applied as an import policy to all BGP routes and is evaluated when routes are imported to the routing table.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set policy-options policy-statement prependpolicy1 term prependterm1 from route-filter
  172.16.0.0/12 orlonger
set policy-options policy-statement prependpolicy1 term prependterm1 from route-filter
  192.168.0.0/16 orlonger
set policy-options policy-statement prependpolicy1 term prependterm1 from route-filter
  10.0.0.0/8 orlonger
set policy-options policy-statement prependpolicy1 term prependterm1 then
  as-path-prepend "1 1 1 1"
set policy-options policy-statement test term 1 from protocol direct
set protocols bgp import prependpolicy1
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To create a routing policy that prepends AS numbers to multiple routes:

1. Create the routing policy.

   ```
   [edit]
   user@host# edit policy-options policy-statement prependpolicy1
   ```

2. Create the routing term.

   ```
   [edit policy-options policy-statement prependpolicy1]
   user@host# edit term prependterm1
   ```

3. Specify the routes to prepend with AS numbers.

   ```
   [edit policy-options policy-statement prependpolicy1 term prependterm1]
   user@host# set from route-filter 172.16.0.0/12 orlonger
   ```

```
user@host# set from route-filter 192.168.0.0/16 orlonger
user@host# set from route-filter 10.0.0.0/8 orlonger
```

4. Specify the AS numbers to prepend.

```
[edit policy-options policy-statement prependpolicy1 term prependterm1]
user@host# set then as-path-prepend “1 1 1 1”
```

NOTE: If you enter multiple numbers, you must separate each number with a space. Enclose the numbers in double quotation marks.

5. Apply the policy as an import policy for all BGP routes.

```
[edit]
user@host# set protocols bgp import prependpolicy1
```

NOTE: You can refer to the same routing policy one or more times in the same or different import statement.

Results Confirm your configuration by entering the **show policy-options** and **show protocols bgp** commands from configuration mode. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
user@host# show policy-options
policy-statement prependpolicy1 {
    term prependterm1 {
        from {
            route-filter 172.16.0.0/12 orlonger;
            route-filter 192.168.0.0/16 orlonger;
            route-filter 10.0.0.0/8 orlonger;
        }
        then as-path-prepend "1 1 1 1";
    }
}

user@host# show protocols bgp
import prependpolicy1;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

To confirm that the configuration is working properly, perform these tasks:

- Verifying the AS Numbers to Prepend on page 329
- Verifying the Routing Policy on page 329

Verifying the AS Numbers to Prepend

Purpose Verify that the policy and term are configured on the device and that the appropriate routes are specified to prepend with AS numbers.

Action From operational mode, enter the **show policy-options** command.

Verifying the Routing Policy

Purpose Verify that the routing policy is applied to the routing protocol.

Action From operational mode, enter the **show protocols bgp** command.

Related Documentation

- *Junos OS Feature Support Reference for SRX Series and J Series Devices*

Understanding Adding AS Numbers to BGP AS Paths

You can expand or add one or more AS numbers to an AS sequence. The AS numbers are added before the local AS number has been added to the path. Expanding an AS path makes a shorter AS path look longer and therefore less preferable to BGP. The last AS number in the existing path is extracted and prepended *n* times, where *n* is a number from 1 through 32. This is similar to the AS path prepend action, except that the AS path expand action adds an arbitrary sequence of AS numbers.

For example, from AS 1 there are two equal paths (through AS 2 and AS 3) to reach AS 4. You might want packets from certain sources to use the path through AS 2. Therefore, you must make the path through AS 3 less preferable so that BGP chooses the path through AS 2. In AS 1, you can expand multiple AS numbers.

```
[edit]
policy-options {
  policy-statement as-path-expand {
    term expand {
      from {
        route-filter 192.168.0.0/16 orlonger;
        route-filter 172.16.0.0/12 orlonger;
        route-filter 10.0.0.0/8 orlonger;
      }
      then as-path-expand last-as count 4;
    }
  }
}
```

For routes from AS 2, this makes the route look like 1 2 2 2 2 2 when advertised, where 1 is from AS 1, the 2 from AS 2 is prepended four times, and the final 2 is the original 2 received from the neighbor router.

Related Documentation

- Example: Advertising Multiple Paths in BGP on page 330
- Example: Configuring a Routing Policy to Prepend the AS Path on page 326

Example: Advertising Multiple Paths in BGP

In this example, BGP routers are configured to advertise multiple paths instead of advertising only the active path. Advertising multiple paths in BGP is specified in Internet draft draft-ietf-idr-add-paths-04, *Advertisement of Multiple Paths in BGP*.

- Requirements on page 330
- Overview on page 330
- Configuration on page 331
- Verification on page 351

Requirements

This example uses the following hardware and software components:

- Eight BGP-enabled devices.
- Five of the BGP-enabled devices do not necessarily need to be routers. For example, they can be EX Series Ethernet Switches.
- Three of the BGP-enabled devices are configured to send multiple paths or receive multiple paths (or both send and receive multiple paths). These three BGP-enabled devices must be M Series Multiservice Edge Routers, MX Series 3D Universal Edge Routers, or T Series Core Routers.
- The three routers must be running Junos OS Release 11.4 or later.

Overview

The following statements are used for configuring multiple paths to a destination:

```
[edit protocols bgp group group-name family family]
add-path {
  receive;
  send {
    path-count number;
    prefix-policy [ policy-names ];
  }
}
```

In this example, Router R5, Router R6, and Router R7 redistribute static routes into BGP. Router R1 and Router R4 are route reflectors. Router R2 and Router R3 are clients to Route Reflector R1. Router R8 is a client to Route Reflector R4.

Route reflection is optional when multiple-path advertisement is enabled in BGP.

With the **add-path send path-count 6** configuration, Router R1 is configured to send up to six paths (per destination) to Router R4.

With the **add-path receive** configuration, Router R4 is configured to receive multiple paths from Router R1.

With the **add-path send path-count 6** configuration, Router R4 is configured to send up to six paths to Router R8.

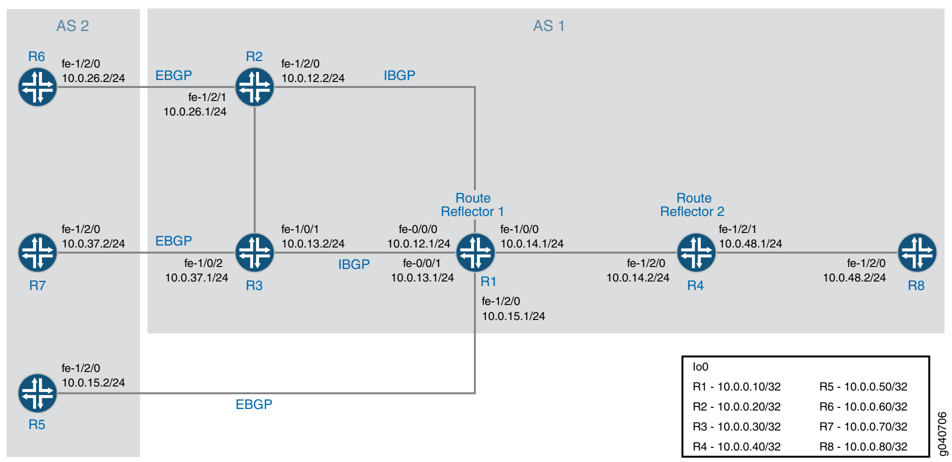
With the **add-path receive** configuration, Router R8 is configured to receive multiple paths from Router R4.

The **add-path send prefix-policy allow_199** policy configuration (along with the corresponding route filter) limits Router R4 to sending multiple paths for only the 172.16.199.1/32 route.

Topology Diagram

Figure 31 on page 331 shows the topology used in this example.

Figure 31: Advertisement of Multiple Paths in BGP



Configuration

- Configuring Router R1 on page 334
- Configuring Router R2 on page 337
- Configuring Router R3 on page 339
- Configuring Router R4 on page 341
- Configuring Router R5 on page 344
- Configuring Router R6 on page 346
- Configuring Router R7 on page 347
- Configuring Router R8 on page 349
- Results on page 350

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Router R1

```
set interfaces fe-0/0/0 unit 12 family inet address 10.0.12.1/24
set interfaces fe-0/0/1 unit 13 family inet address 10.0.13.1/24
set interfaces fe-1/0/0 unit 14 family inet address 10.0.14.1/24
set interfaces fe-1/2/0 unit 15 family inet address 10.0.15.1/24
set interfaces lo0 unit 10 family inet address 10.0.0.10/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.10
set protocols bgp group rr cluster 10.0.0.10
set protocols bgp group rr neighbor 10.0.0.20
set protocols bgp group rr neighbor 10.0.0.30
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.15.2 local-address 10.0.15.1
set protocols bgp group e1 neighbor 10.0.15.2 peer-as 2
set protocols bgp group rr_rr type internal
set protocols bgp group rr_rr local-address 10.0.0.10
set protocols bgp group rr_rr neighbor 10.0.0.40 family inet unicast add-path send
  path-count 6
set protocols ospf area 0.0.0.0 interface lo0.10 passive
set protocols ospf area 0.0.0.0 interface fe-0/0/0.12
set protocols ospf area 0.0.0.0 interface fe-0/0/1.13
set protocols ospf area 0.0.0.0 interface fe-1/0/0.14
set protocols ospf area 0.0.0.0 interface fe-1/2/0.15
set routing-options router-id 10.0.0.10
set routing-options autonomous-system 1
```

Router R2

```
set interfaces fe-1/2/0 unit 21 family inet address 10.0.12.2/24
set interfaces fe-1/2/1 unit 26 family inet address 10.0.26.1/24
set interfaces lo0 unit 20 family inet address 10.0.0.20/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.20
set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.26.2 peer-as 2
set protocols ospf area 0.0.0.0 interface lo0.20 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.21
set protocols ospf area 0.0.0.0 interface fe-1/2/1.28
set policy-options policy-statement set_nh_self then next-hop self
set routing-options autonomous-system 1
```

Router R3

```
set interfaces fe-1/0/1 unit 31 family inet address 10.0.13.2/24
set interfaces fe-1/0/2 unit 37 family inet address 10.0.37.1/24
set interfaces lo0 unit 30 family inet address 10.0.0.30/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.30
set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.37.2 peer-as 2
set protocols ospf area 0.0.0.0 interface lo0.30 passive
set protocols ospf area 0.0.0.0 interface fe-1/0/1.31
set protocols ospf area 0.0.0.0 interface fe-1/0/2.37
set policy-options policy-statement set_nh_self then next-hop self
```

```
set routing-options autonomous-system 1
```

Router R4

```
set interfaces fe-1/2/0 unit 41 family inet address 10.0.14.2/24
set interfaces fe-1/2/1 unit 48 family inet address 10.0.48.1/24
set interfaces lo0 unit 40 family inet address 10.0.0.40/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.40
set protocols bgp group rr family inet unicast add-path receive
set protocols bgp group rr neighbor 10.0.0.10
set protocols bgp group rr_client type internal
set protocols bgp group rr_client local-address 10.0.0.40
set protocols bgp group rr_client cluster 10.0.0.40
set protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send
  path-count 6
set protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast add-path send
  prefix-policy allow_199
set protocols ospf area 0.0.0.0 interface fe-1/2/0.41
set protocols ospf area 0.0.0.0 interface lo0.40 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/1.48
set policy-options policy-statement allow_199 from route-filter 172.16.199.1/32 exact
set policy-options policy-statement allow_199 term match_199 from prefix-list match_199
set policy-options policy-statement allow_199 then add-path send-count 20
set policy-options policy-statement allow_199 then accept
set routing-options autonomous-system 1
```

Router R5

```
set interfaces fe-1/2/0 unit 51 family inet address 10.0.15.2/24
set interfaces lo0 unit 50 family inet address 10.0.0.50/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.15.1 export s2b
set protocols bgp group e1 neighbor 10.0.15.1 peer-as 1
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then as-path-expand 2
set policy-options policy-statement s2b then accept
set routing-options autonomous-system 2
set routing-options static route 172.16.199.1/32 reject
set routing-options static route 172.16.198.1/32 reject
```

Router R6

```
set interfaces fe-1/2/0 unit 62 family inet address 10.0.26.2/24
set interfaces lo0 unit 60 family inet address 10.0.0.60/32
set protocols bgp group e1 type external
set protocols bgp group e1 neighbor 10.0.26.1 export s2b
set protocols bgp group e1 neighbor 10.0.26.1 peer-as 1
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then accept
set routing-options autonomous-system 2
set routing-options static route 172.16.199.1/32 reject
set routing-options static route 172.16.198.1/32 reject
```

Router R7

```
set interfaces fe-1/2/0 unit 73 family inet address 10.0.37.2/24
set interfaces lo0 unit 70 family inet address 10.0.0.70/32
set protocols bgp group e1 type external
```

```
set protocols bgp group e1 neighbor 10.0.37.1 export s2b
set protocols bgp group e1 neighbor 10.0.37.1 peer-as 1
set policy-options policy-statement s2b from protocol static
set policy-options policy-statement s2b from protocol direct
set policy-options policy-statement s2b then accept
set routing-options autonomous-system 2
set routing-options static route 172.16.199.1/32 reject
```

Router R8

```
set interfaces fe-1/2/0 unit 84 family inet address 10.0.48.2/24
set interfaces lo0 unit 80 family inet address 10.0.0.80/32
set protocols bgp group rr type internal
set protocols bgp group rr local-address 10.0.0.80
set protocols bgp group rr neighbor 10.0.0.40 family inet unicast add-path receive
set protocols ospf area 0.0.0.0 interface lo0.80 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.84
set routing-options autonomous-system 1
```

Configuring Router R1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Router R1:

1. Configure the interfaces to Router R2, Router R3, Router R4, and Router R5, and configure the loopback (lo0) interface.

```
[edit interfaces]
user@R1# set fe-0/0/0 unit 12 family inet address 10.0.12.1/24

user@R1# set fe-0/0/1 unit 13 family inet address 10.0.13.1/24

user@R1# set fe-1/0/0 unit 14 family inet address 10.0.14.1/24

user@R1# set fe-1/2/0 unit 15 family inet address 10.0.15.1/24

user@R1#set lo0 unit 10 family inet address 10.0.0.10/32
```

2. Configure BGP on the interfaces, and configure IBGP route reflection.

```
[edit protocols bgp]
user@R1# set group rr type internal
user@R1# set group rr local-address 10.0.0.10
user@R1# set group rr cluster 10.0.0.10
user@R1# set group rr neighbor 10.0.0.20
user@R1# set group rr neighbor 10.0.0.30

user@R1# set group rr_rr type internal
user@R1# set group rr_rr local-address 10.0.0.10

user@R1# set group e1 type external
```



```
user@R1# set group e1 neighbor 10.0.15.2 local-address 10.0.15.1
user@R1# set group e1 neighbor 10.0.15.2 peer-as 2
```

3. Configure Router R1 to send up to six paths to its neighbor, Router R4.

   The destination of the paths can be any destination that Router R1 can reach through multiple paths.

   ```
   [edit protocols bgp]
   user@R1# set group rr_rr neighbor 10.0.0.40 family inet unicast add-path send
     path-count 6
   ```

4. Configure OSPF on the interfaces.

   ```
   [edit protocols ospf]
   user@R1# set area 0.0.0.0 interface lo0.10 passive
   user@R1# set area 0.0.0.0 interface fe-0/0/0.12
   user@R1# set area 0.0.0.0 interface fe-0/0/1.13
   user@R1# set area 0.0.0.0 interface fe-1/0/0.14
   user@R1# set area 0.0.0.0 interface fe-1/2/0.15
   ```

5. Configure the router ID and the autonomous system number.

   ```
   [edit routing-options]
   user@R1# set router-id 10.0.0.10
   user@R1# set autonomous-system 1
   ```

6. If you are done configuring the device, commit the configuration.

   ```
   user@R1# commit
   ```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
fe-0/0/0 {
  unit 12 {
    family inet {
      address 10.0.12.1/24;
    }
  }
}
fe-0/0/1 {
  unit 13 {
    family inet {
      address 10.0.13.1/24;
    }
  }
}
fe-1/0/0 {
```

```
   unit 14 {
      family inet {
         address 10.0.14.1/24;
      }
   }
}
fe-1/2/0 {
   unit 15 {
      family inet {
         address 10.0.15.1/24;
      }
   }
}
lo0 {
   unit 10 {
      family inet {
         address 10.0.0.10/32;
      }
   }
}
```

```
user@R1# show protocols
bgp {
   group rr {
      type internal;
      local-address 10.0.0.10;
      cluster 10.0.0.10;
      neighbor 10.0.0.20;
      neighbor 10.0.0.30;
   }
   group e1 {
      type external;
      neighbor 10.0.15.2 {
         local-address 10.0.15.1;
         peer-as 2;
      }
   }
   group rr_rr {
      type internal;
      local-address 10.0.0.10;
      neighbor 10.0.0.40 {
         family inet {
            unicast {
               add-path {
                  send {
                     path-count 6;
                  }
               }
            }
         }
      }
   }
}
ospf {
   area 0.0.0.0 {
      interface lo0.10 {
```



```
            passive;
          }
          interface fe-0/0/0.12;
          interface fe-0/0/1.13;
          interface fe-1/0/0.14;
          interface fe-1/2/0.15;
        }
      }
```

```
user@R1# show routing-options
router-id 10.0.0.10;
autonomous-system 1;
```

Configuring Router R2

Step-by-Step Procedure

To configure Router R2:

1. Configure the loopback (lo0) interface and the interfaces to Router R6 and Router R1.

   ```
   [edit interfaces]
   user@R2# set fe-1/2/0 unit 21 family inet address 10.0.12.2/24

   user@R2# set fe-1/2/1 unit 26 family inet address 10.0.26.1/24

   user@R2# set lo0 unit 20 family inet address 10.0.0.20/32
   ```

2. Configure BGP and OSPF on Router R2's interfaces.

   ```
   [edit protocols]
   user@R2# set bgp group rr type internal
   user@R2# set bgp group rr local-address 10.0.0.20

   user@R2# set bgp group e1 type external
   user@R2# set bgp group e1 neighbor 10.0.26.2 peer-as 2

   user@R2# set ospf area 0.0.0.0 interface lo0.20 passive
   user@R2# set ospf area 0.0.0.0 interface fe-1/2/0.21
   user@R2# set ospf area 0.0.0.0 interface fe-1/2/1.28
   ```

3. For routes sent from Router R2 to Router R1, advertise Router R2 as the next hop, because Router R1 does not have a route to Router R6's address on the 10.0.26.0/24 network.

   ```
   [edit]
   user@R2# set policy-options policy-statement set_nh_self then next-hop self

   user@R2# set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
   ```

4. Configure the autonomous system number.

   ```
   [edit]
   user@R2# set routing-options autonomous-system 1
   ```

5. If you are done configuring the device, commit the configuration.

```
user@R2# commit
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**,and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
fe-1/2/0 {
  unit 21 {
    family inet {
      address 10.0.12.2/24;
    }
  }
}
fe-1/2/1 {
  unit 26 {
    family inet {
      address 10.0.26.1/24;
    }
  }
}
lo0 {
  unit 20 {
    family inet {
      address 10.0.0.20/32;
    }
  }
}
```

```
user@R2# show protocols
bgp {
  group rr {
    type internal;
    local-address 10.0.0.20;
    neighbor 10.0.0.10 {
      export set_nh_self;
    }
  }
  group e1 {
    type external;
    neighbor 10.0.26.2 {
      peer-as 2;
    }
  }
}
ospf {
  area 0.0.0.0 {
    interface lo0.20 {
      passive;
    }
    interface fe-1/2/0.21;
```



```
      interface fe-1/2/1.28;
    }
}
```

```
user@R2# show policy-options
policy-statement set_nh_self {
    then {
        next-hop self;
    }
}
```

```
user@R2# show routing-options
autonomous-system 1;
```

Configuring Router R3

Step-by-Step Procedure

To configure Router R3:

1. Configure the loopback (lo0) interface and the interfaces to Router R7 and Router R1.

```
[edit interfaces]
user@R3# set fe-1/0/1 unit 31 family inet address 10.0.13.2/24

user@R3# set fe-1/0/2 unit 37 family inet address 10.0.37.1/24

user@R3# set lo0 unit 30 family inet address 10.0.0.30/32
```

2. Configure BGP and OSPF on Router R3's interfaces.

```
[edit protocols]
user@R3# set bgp group rr type internal
user@R3# set bgp group rr local-address 10.0.0.30

user@R3# set bgp group e1 type external
user@R3# set bgp group e1 neighbor 10.0.37.2 peer-as 2

user@R3# set ospf area 0.0.0.0 interface lo0.30 passive
user@R3# set ospf area 0.0.0.0 interface fe-1/0/1.31
user@R3# set ospf area 0.0.0.0 interface fe-1/0/2.37
```

3. For routes sent from Router R3 to Router R1, advertise Router R3 as the next hop, because Router R1 does not have a route to Router R7's address on the 10.0.37.0/24 network.

```
[edit]
user@R3# set policy-options policy-statement set_nh_self then next-hop self

user@R3# set protocols bgp group rr neighbor 10.0.0.10 export set_nh_self
```

4. Configure the autonomous system number.

```
[edit]
```



```
user@R3# set routing-options autonomous-system 1
```

5. If you are done configuring the device, commit the configuration.

```
user@R3# commit
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R3# show interfaces
fe-1/0/1 {
  unit 31 {
    family inet {
      address 10.0.13.2/24;
    }
  }
}
fe-1/0/2 {
  unit 37 {
    family inet {
      address 10.0.37.1/24;
    }
  }
}
lo0 {
  unit 30 {
    family inet {
      address 10.0.0.30/32;
    }
  }
}
```

```
user@R3# show protocols
bgp {
  group rr {
    type internal;
    local-address 10.0.0.30;
    neighbor 10.0.0.10 {
      export set_nh_self;
    }
  }
  group e1 {
    type external;
    neighbor 10.0.37.2 {
      peer-as 2;
    }
  }
}
ospf {
  area 0.0.0.0 {
    interface lo0.30 {
      passive;
```



```
      }
      interface fe-1/0/1.31;
      interface fe-1/0/2.37;
    }
}
user@R3# show policy-options
policy-statement set_nh_self {
    then {
      next-hop self;
    }
}
user@R3# show routing-options
autonomous-system 1;
```

Configuring Router R4

Step-by-Step Procedure

To configure Router R4:

1. Configure the interfaces to Router R1 and Router R8, and configure the loopback (lo0) interface.

   ```
   [edit interfaces]
   user@R4# set fe-1/2/0 unit 41 family inet address 10.0.14.2/24

   user@R4# set fe-1/2/1 unit 48 family inet address 10.0.48.1/24

   user@R4# set lo0 unit 40 family inet address 10.0.0.40/32
   ```

2. Configure BGP on the interfaces, and configure IBGP route reflection.

   ```
   [edit protocols bgp]
   user@R4# set group rr type internal
   user@R4# set group rr local-address 10.0.0.40
   user@R4# set group rr neighbor 10.0.0.10

   user@R4# set group rr_client type internal
   user@R4# set group rr_client local-address 10.0.0.40
   user@R4# set group rr_client cluster 10.0.0.40
   ```

3. Configure Router R4 to send up to six paths to its neighbor, Router R8.

   The destination of the paths can be any destination that Router R4 can reach through multiple paths.

   ```
   [edit protocols bgp]
   user@R4# set group rr_client neighbor 10.0.0.80 family inet unicast add-path send
     path-count 6
   ```

4. Configure Router R4 to receive multiple paths from its neighbor, Router R1.

   The destination of the paths can be any destination that Router R1 can reach through multiple paths.

   ```
   [edit protocols bgp group rr family inet unicast]
   user@R4# set add-path receive
   ```

5. Configure OSPF on the interfaces.

   ```
   [edit protocols ospf area 0.0.0.0]
   user@R4# set interface fe-1/2/0.41
   user@R4# set interface lo0.40 passive
   user@R4# set interface fe-1/2/1.48
   ```

6. Configure a policy that allows Router R4 to send Router R8 multiple paths to the 172.16.199.1/32 route.

   - Router R4 receives multiple paths for the 172.16.198.1/32 route and the 172.16.199.1/32 route. However, because of this policy, Router R4 only sends multiple paths for the 172.16.199.1/32 route.

     ```
     [edit protocols bgp group rr_client neighbor 10.0.0.80 family inet unicast]
     user@R4# set add-path send prefix-policy allow_199
     [edit policy-options policy-statement allow_199]
     user@R4# set from route-filter 172.16.199.1/32 exact
     user@R4# set then accept
     ```

   - Router R4 can also be configured to send up-to 20 BGP **add-path** routes for a subset of *add-path advertised prefixes*.

     ```
     [edit policy-options policy-statement allow_199]
     user@R4# set term match_199 from prefix-list match_199
     user@R4# set then add-path send-count 20
     ```

7. Configure the autonomous system number.

   ```
   [edit routing-options]
   user@R4# set autonomous-system 1
   ```

8. If you are done configuring the device, commit the configuration.

   ```
   user@R4# commit
   ```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R4# show interfaces
fe-1/2/0 {
  unit 41 {
    family inet {
      address 10.0.14.2/24;
```



```
        }
    }
}
fe-1/2/1 {
    unit 48 {
        family inet {
            address 10.0.48.1/24;
        }
    }
}
lo0 {
    unit 40 {
        family inet {
            address 10.0.0.40/32;
        }
    }
}
```

```
user@R4# show protocols
bgp {
    group rr {
        type internal;
        local-address 10.0.0.40;
        family inet {
            unicast {
                add-path {
                    receive;
                }
            }
        }
        neighbor 10.0.0.10;
    }
    group rr_client {
        type internal;
        local-address 10.0.0.40;
        cluster 10.0.0.40;
        neighbor 10.0.0.80 {
            family inet {
                unicast {
                    add-path {
                        send {
                            path-count 6;
                            prefix-policy allow_199;
                        }
                    }
                }
            }
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.40 {
            passive;
        }
        interface fe-1/2/0.41;
```

```
        interface fe-1/2/1.48;
    }
}
```

```
user@R4# show policy-options
policy-statement allow_199 {
    from {
        route-filter 172.16.199.1/32 exact;
    }
    from term match_199 {
        prefix-list match_199;
    }
        then add-path send-count 20;
    then accept;
}
```

```
user@R4# show routing-options
autonomous-system 1;
```

Configuring Router R5

Step-by-Step Procedure

To configure Router R5:

1. Configure the loopback (lo0) interface and the interface to Router R1.

   ```
   [edit interfaces]
   user@R5# set fe-1/2/0 unit 51 family inet address 10.0.15.2/24
   ```

   ```
   user@R5# set lo0 unit 50 family inet address 10.0.0.50/32
   ```

2. Configure BGP on Router R5's interface.

   ```
   [edit protocols bgp group e1]
   user@R5# set type external
   user@R5# set neighbor 10.0.15.1 peer-as 1
   ```

3. Create static routes for redistribution into BGP.

   ```
   [edit routing-options]
   user@R5# set static route 172.16.199.1/32 reject
   user@R5# set static route 172.16.198.1/32 reject
   ```

4. Redistribute static and direct routes into BGP.

   ```
   [edit protocols bgp group e1 neighbor 10.0.15.1]
   user@R5# set export s2b
   ```

   ```
   [edit policy-options policy-statement s2b]
   user@R5# set from protocol static
   user@R5# set from protocol direct
   user@R5# set then as-path-expand 2
   user@R5# set then accept
   ```


5. Configure the autonomous system number.

```
[edit routing-options]
user@R5# set autonomous-system 2
```

6. If you are done configuring the device, commit the configuration.

```
user@R5# commit
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R5# show interfaces
fe-1/2/0 {
   unit 51 {
      family inet {
         address 10.0.15.2/24;
      }
   }
}
lo0 {
   unit 50 {
      family inet {
         address 10.0.0.50/32;
      }
   }
}
```

```
user@R5# show protocols
bgp {
   group e1 {
      type external;
      neighbor 10.0.15.1 {
         export s2b;
         peer-as 1;
      }
   }
}
```

```
user@R5# show policy-options
policy-statement s2b {
   from protocol [ static direct ];
   then {
      as-path-expand 2;
      accept;
   }
}
```

```
user@R5# show routing-options
static {
   route 172.16.198.1/32 reject;
   route 172.16.199.1/32 reject;
}
```

```
autonomous-system 2;
```

Configuring Router R6

Step-by-Step Procedure

To configure Router R6:

1. Configure the loopback (lo0) interface and the interface to Router R2.

```
[edit interfaces]
user@R6# set fe-1/2/0 unit 62 family inet address 10.0.26.2/24

user@R6# set lo0 unit 60 family inet address 10.0.0.60/32
```

2. Configure BGP on Router R6's interface.

```
[edit protocols]
user@R6# set bgp group e1 type external
user@R6# set bgp group e1 neighbor 10.0.26.1 peer-as 1
```

3. Create static routes for redistribution into BGP.

```
[edit]
user@R6# set routing-options static route 172.16.199.1/32 reject
user@R6# set routing-options static route 172.16.198.1/32 reject
```

4. Redistribute static and direct routes from Router R6's routing table into BGP.

```
[edit protocols bgp group e1 neighbor 10.0.26.1]
user@R6# set export s2b

[edit policy-options policy-statement s2b]
user@R6# set from protocol static
user@R6# set from protocol direct
user@R6# set then accept
```

5. Configure the autonomous system number.

```
[edit routing-options]
user@R6# set autonomous-system 2
```

6. If you are done configuring the device, commit the configuration.

```
user@R6# commit
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R6# show interfaces
```

```
fe-1/2/0 {
  unit 62 {
    family inet {
      address 10.0.26.2/24;
    }
  }
}
lo0 {
  unit 60 {
    family inet {
      address 10.0.0.60/32;
    }
  }
}
```

```
user@R6# show protocols
bgp {
  group e1 {
    type external;
    neighbor 10.0.26.1 {
      export s2b;
      peer-as 1;
    }
  }
}
```

```
user@R6# show policy-options
policy-statement s2b {
  from protocol [ static direct ];
  then accept;
}
```

```
user@R6# show routing-options
static {
  route 172.16.198.1/32 reject;
  route 172.16.199.1/32 reject;
}
autonomous-system 2;
```

Configuring Router R7

Step-by-Step Procedure

To configure Router R7:

1. Configure the loopback (lo0) interface and the interface to Router R3.

   ```
   [edit interfaces]
   user@R7# set fe-1/2/0 unit 73 family inet address 10.0.37.2/24
   ```

   ```
   user@R7# set lo0 unit 70 family inet address 10.0.0.70/32
   ```

2. Configure BGP on Router R7's interface.

   ```
   [edit protocols bgp group e1]
   user@R7# set type external
   user@R7# set neighbor 10.0.37.1 peer-as 1
   ```


3. Create a static route for redistribution into BGP.

```
[edit]
user@R7# set routing-options static route 172.16.199.1/32 reject
```

4. Redistribute static and direct routes from Router R7's routing table into BGP.

```
[edit protocols bgp group e1 neighbor 10.0.37.1]
user@R7# set export s2b
```

```
[edit policy-options policy-statement s2b]
user@R7# set from protocol static
user@R7# set from protocol direct
user@R7# set then accept
```

5. Configure the autonomous system number.

```
[edit routing-options]
user@R7# set autonomous-system 2
```

6. If you are done configuring the device, commit the configuration.

```
user@R7# commit
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R7# show interfaces
fe-1/2/0 {
  unit 73 {
    family inet {
      address 10.0.37.2/24;
    }
  }
}
lo0 {
  unit 70 {
    family inet {
      address 10.0.0.70/32;
    }
  }
}
```

```
user@R7# show protocols
bgp {
  group e1 {
    type external;
    neighbor 10.0.37.1 {
      export s2b;
      peer-as 1;
    }
```



```
  }
}
user@R7# show policy-options
policy-statement s2b {
  from protocol [ static direct ];
  then accept;
}
user@R7# show routing-options
static {
  route 172.16.199.1/32 reject;
}
autonomous-system 2;
```

Configuring Router R8

Step-by-Step Procedure

To configure Router R8:

1. Configure the loopback (lo0) interface and the interface to Router R4.

   ```
   [edit interfaces]
   user@R8# set fe-1/2/0 unit 84 family inet address 10.0.48.2/24

   user@R8# set lo0 unit 80 family inet address 10.0.0.80/32
   ```

2. Configure BGP and OSPF on Router R8's interface.

   ```
   [edit protocols]
   user@R8# set bgp group rr type internal
   user@R8# set bgp group rr local-address 10.0.0.80

   user@R8# set ospf area 0.0.0.0 interface lo0.80 passive
   user@R8# set ospf area 0.0.0.0 interface fe-1/2/0.84
   ```

3. Configure Router R8 to receive multiple paths from its neighbor, Router R4.

   The destination of the paths can be any destination that Router R4 can reach through multiple paths.

   ```
   [edit protocols]
   user@R8# set bgp group rr neighbor 10.0.0.40 family inet unicast add-path receive
   ```

4. Configure the autonomous system number.

   ```
   [edit]
   user@R8# set routing-options autonomous-system 1
   ```

5. If you are done configuring the device, commit the configuration.

   ```
   user@R8# commit
   ```


Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R8# show interfaces
fe-1/2/0 {
  unit 84 {
    family inet {
      address 10.0.48.2/24;
    }
  }
}
lo0 {
  unit 80 {
    family inet {
      address 10.0.0.80/32;
    }
  }
}
```

```
user@R8# show protocols
bgp {
  group rr {
    type internal;
    local-address 10.0.0.80;
    neighbor 10.0.0.40 {
      family inet {
        unicast {
          add-path {
            receive;
          }
        }
      }
    }
  }
}
ospf {
  area 0.0.0.0 {
    interface lo0.80 {
      passive;
    }
    interface fe-1/2/0.84;
  }
}
```

```
user@R8# show routing-options
autonomous-system 1;
```


Verification

Confirm that the configuration is working properly.

Verifying That the BGP Peers Have the Ability to Send and Receive Multiple Paths

Purpose Make sure that one or both of the following strings appear in the output of the **show bgp neighbor** command:

- **NLRI's for which peer can receive multiple paths: inet-unicast**
- **NLRI's for which peer can send multiple paths: inet-unicast**

Action

```
user@R1> show bgp neighbor 10.0.0.40
Peer: 10.0.0.40+179 AS 1       Local: 10.0.0.10+64227 AS 1
  Type: Internal    State: Established    Flags: <Sync>
...  NLRI's for which peer can receive multiple paths: inet-unicast
...
```

```
user@R4> show bgp neighbor 10.0.0.10
Peer: 10.0.0.10+64227 AS 1     Local: 10.0.0.40+179 AS 1
  Type: Internal    State: Established    Flags: <Sync>
...
  NLRI's for which peer can send multiple paths: inet-unicast
...
```

```
user@R4> show bgp neighbor 10.0.0.80
Peer: 10.0.0.80+55416 AS 1     Local: 10.0.0.40+179 AS 1
  Type: Internal    State: Established  (route reflector client)Flags: <Sync>
  ,,,
  NLRI's for which peer can receive multiple paths: inet-unicast
  ...
```

```
user@R8> show bgp neighbor 10.0.0.40
Peer: 10.0.0.40+179 AS 1       Local: 10.0.0.80+55416 AS 1
  Type: Internal    State: Established    Flags: <Sync>
  ...
  NLRI's for which peer can send multiple paths: inet-unicast
  ...
```

Verifying That Router R1 Is Advertising Multiple Paths

Purpose Make sure that multiple paths to the 172.16.198.1/32 destination and multiple paths to the 172.16.199.1/32 destination are advertised to Router R4.

Action

```
user@R1> show route advertising-protocol bgp 10.0.0.40
inet.0: 21 destinations, 25 routes (21 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                    100        2 2 I
* 10.0.0.60/32            10.0.0.20                    100        2 I
* 10.0.0.70/32            10.0.0.30                    100        2 I
* 172.16.198.1/32            10.0.0.20                    100        2 I
                          10.0.15.2                    100        2 2 I
* 172.16.199.1/32            10.0.0.20                    100        2 I
                          10.0.0.30                    100        2 I
                          10.0.15.2                    100        2 2 I
* 172.16.200.0/30            10.0.0.20                    100        2 I
```

Meaning When you see one prefix and more than one next hop, it means that multiple paths are advertised to Router R4.

Verifying That Router R4 Is Receiving and Advertising Multiple Paths

Purpose Make sure that multiple paths to the 172.16.199.1/32 destination are received from Router R1 and advertised to Router R8. Make sure that multiple paths to the 172.16.198.1/32 destination are received from Router R1, but only one path to this destination is advertised to Router R8.

Action

```
user@R4> show route receive-protocol bgp 10.0.0.10
inet.0: 19 destinations, 22 routes (19 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                    100        2 2 I
* 10.0.0.60/32            10.0.0.20                    100        2 I
* 10.0.0.70/32            10.0.0.30                    100        2 I
* 172.16.198.1/32            10.0.0.20                    100        2 I
                          10.0.15.2                    100        2 2 I
* 172.16.199.1/32            10.0.0.20                    100        2 I
                          10.0.0.30                    100        2 I
                          10.0.15.2                    100        2 2 I
* 172.16.200.0/30            10.0.0.20                    100        2 I
```

```
user@R4> show route advertising-protocol bgp 10.0.0.80
inet.0: 19 destinations, 22 routes (19 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                    100        2 2 I
* 10.0.0.60/32            10.0.0.20                    100        2 I
* 10.0.0.70/32            10.0.0.30                    100        2 I
* 172.16.198.1/32            10.0.0.20                    100        2 I
* 172.16.199.1/32            10.0.0.20                    100        2 I
                          10.0.0.30                    100        2 I
                          10.0.15.2                    100        2 2 I
* 172.16.200.0/30            10.0.0.20                    100        2 I
```

Meaning The **show route receive-protocol** command shows that Router R4 receives two paths to the 172.16.198.1/32 destination and three paths to the 172.16.199.1/32 destination. The **show route advertising-protocol** command shows that Router R4 advertises only one path to the 172.16.198.1/32 destination and advertises all three paths to the 172.16.199.1/32 destination.

Because of the prefix policy that is applied to Router R4, Router R4 does not advertise multiple paths to the 172.16.198.1/32 destination. Router R4 advertises only one path to the 172.16.198.1/32 destination even though it receives multiple paths to this destination.

Verifying That Router R8 Is Receiving Multiple Paths

Purpose Make sure that Router R8 receives multiple paths to the 172.16.199.1/32 destination through Router R4. Make sure that Router R8 receives only one path to the 172.16.198.1/32 destination through Router R4.

Action

```
user@R8> show route receive-protocol bgp 10.0.0.40
inet.0: 18 destinations, 20 routes (18 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.50/32            10.0.15.2                    100        2 2 I
* 10.0.0.60/32            10.0.0.20                    100        2 I
* 10.0.0.70/32            10.0.0.30                    100        2 I
* 172.16.198.1/32            10.0.0.20                    100        2 I
* 172.16.199.1/32            10.0.0.20                    100        2 I
                          10.0.0.30                    100        2 I
                          10.0.15.2                    100        2 2 I
* 200.1.1.0/30            10.0.0.20                    100        2 I
```

Checking the Path ID

Purpose On the downstream devices, Router R4 and Router R8, verify that a path ID uniquely identifies the path. Look for the **Addpath Path ID:** string.

Action `user@R4>` **show route 172.16.199.1/32 detail**

```
inet.0: 18 destinations, 20 routes (18 active, 0 holddown, 0 hidden)
172.16.199.1/32 (3 entries, 3 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 9
                Source: 10.0.0.10
                Next hop type: Router, Next hop index: 676
                Next hop: 10.0.14.1 via lt-1/2/0.41, selected
                Protocol next hop: 10.0.0.20
                Indirect next hop: 92041c8 262146
                State: <Active Int Ext>
                Local AS:     1 Peer AS:     1
                Age: 1:44:37    Metric2: 2
                Task: BGP_1.10.0.0.10+64227
                Announcement bits (3): 2-KRT 3-BGP RT Background 4-Resolve tree
1
                AS path: 2 I (Originator) Cluster list:  10.0.0.10
                AS path:  Originator ID: 10.0.0.20
                Accepted
                Localpref: 100
                Router ID: 10.0.0.10
                Addpath Path ID: 1
         BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 4
                Source: 10.0.0.10
                Next hop type: Router, Next hop index: 676
                Next hop: 10.0.14.1 via lt-1/2/0.41, selected
                Protocol next hop: 10.0.0.30
                Indirect next hop: 92042ac 262151
                State: <NotBest Int Ext>
                Inactive reason: Not Best in its group - Router ID
                Local AS:     1 Peer AS:     1
                Age: 1:44:37    Metric2: 2
                Task: BGP_1.10.0.0.10+64227
                Announcement bits (1): 3-BGP RT Background
                AS path: 2 I (Originator) Cluster list:  10.0.0.10
                AS path:  Originator ID: 10.0.0.30
                Accepted
                Localpref: 100
                Router ID: 10.0.0.10
                Addpath Path ID: 2
         BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 4
                Source: 10.0.0.10
                Next hop type: Router, Next hop index: 676
                Next hop: 10.0.14.1 via lt-1/2/0.41, selected
                Protocol next hop: 10.0.15.2
                Indirect next hop: 92040e4 262150
                State: <Int Ext>
                Inactive reason: AS path
                Local AS:     1 Peer AS:     1
                Age: 1:44:37    Metric2: 2
                Task: BGP_1.10.0.0.10+64227
                Announcement bits (1): 3-BGP RT Background
                AS path: 2 2 I
                Accepted
```



```
                Localpref: 100
                Router ID: 10.0.0.10
                Addpath Path ID: 3
```

user@R8> **show route 172.16.199.1/32 detail**

```
inet.0: 17 destinations, 19 routes (17 active, 0 holddown, 0 hidden)
172.16.199.1/32 (3 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 9
                Source: 10.0.0.40
                Next hop type: Router, Next hop index: 1045
                Next hop: 10.0.48.1 via lt-1/2/0.84, selected
                Protocol next hop: 10.0.0.20
                Indirect next hop: 91fc0e4 262148
                State: <Active Int Ext>
                Local AS:     1 Peer AS:     1
                Age: 1:56:51    Metric2: 3
                Task: BGP_1.10.0.0.40+179
                Announcement bits (2): 2-KRT 4-Resolve tree 1
                AS path: 2 I (Originator) Cluster list:  10.0.0.40 10.0.0.10
                AS path:  Originator ID: 10.0.0.20
                Accepted
                Localpref: 100
                Router ID: 10.0.0.40
                Addpath Path ID: 1
         BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 4
                Source: 10.0.0.40
                Next hop type: Router, Next hop index: 1045
                Next hop: 10.0.48.1 via lt-1/2/0.84, selected
                Protocol next hop: 10.0.0.30
                Indirect next hop: 91fc1c8 262152
                State: <NotBest Int Ext>
                Inactive reason: Not Best in its group - Router ID
                Local AS:     1 Peer AS:     1
                Age: 1:56:51    Metric2: 3
                Task: BGP_1.10.0.0.40+179
                AS path: 2 I (Originator) Cluster list:  10.0.0.40 10.0.0.10
                AS path:  Originator ID: 10.0.0.30
                Accepted
                Localpref: 100
                Router ID: 10.0.0.40
                Addpath Path ID: 2
         BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 4
                Source: 10.0.0.40
                Next hop type: Router, Next hop index: 1045
                Next hop: 10.0.48.1 via lt-1/2/0.84, selected
                Protocol next hop: 10.0.15.2
                Indirect next hop: 91fc2ac 262153
                State: <Int Ext>
                Inactive reason: AS path
                Local AS:     1 Peer AS:     1
                Age: 1:56:51    Metric2: 3
                Task: BGP_1.10.0.0.40+179
                AS path: 2 2 I (Originator) Cluster list:  10.0.0.40
```



```
AS path:  Originator ID: 10.0.0.10
Accepted
Localpref: 100
Router ID: 10.0.0.40
Addpath Path ID: 3
```

Related Documentation

- *Understanding the Advertisement of Multiple Paths to a Single Destination in BGP*
- Understanding Adding AS Numbers to BGP AS Paths on page 329

CHAPTER 7

Configuring Communities as Match Conditions

Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions

A *BGP community* is a group of destinations that share a common property. Community information is included as a path attribute in BGP update messages. This information identifies community members and enables you to perform actions on a group without having to elaborate upon each member. You can use community and extended communities attributes to trigger routing decisions, such as acceptance, rejection, preference, or redistribution.

You can assign community tags to non-BGP routes through configuration (for static, aggregate, or generated routes) or an import routing policy. These tags can then be matched when BGP exports the routes.

A community value is a 32-bit field that is divided into two main sections. The first 16 bits of the value encode the AS number of the network that originated the community, while the last 16 bits carry a unique number assigned by the AS. This system attempts to guarantee a globally unique set of community values for each AS in the Internet. Junos OS uses a notation of ***as-number:community-value***, where each value is a decimal number. The AS values of 0 and 65,535 are reserved, as are all of the community values within those AS numbers. Each community, or set of communities, is given a name within the

[edit policy-options] configuration hierarchy. The name of the community uniquely identifies it to the routing device and serves as the method by which routes are categorized. For example, a route with a community value of 64510:1111 might belong to the community named **AS64510-routes**. The community name is also used within a routing policy as a match criterion or as an action. The command syntax for creating a community is: policy-options **community *name* members [*community-ids*]**. The ***community-ids*** are either a single community value or multiple community values. When more than one value is assigned to a community name, the routing device interprets this as a logical AND of the community values. In other words, a route must have all of the configured values before being assigned the community name.

The regular community attribute is four octets. Networking enhancements, such as VPNs, have functionality requirements that can be satisfied by an attribute such as a community. However, the 4-octet community value does not provide enough expansion and flexibility to accommodate VPN requirements. This leads to the creation of extended communities. An extended community is an 8-octet value that is also divided into two main sections. The first 2 octets of the community encode a type field while the last 6 octets carry a unique set of data in a format defined by the type field. Extended communities provide a larger range for grouping or categorizing communities.

The BGP extended communities attribute format has three fields: ***type*:*administrator*:*assigned-number***. The routing device expects you to use the words **target** or **origin** to represent the type field. The administrator field uses a decimal number for the AS or an IPv4 address, while the assigned number field expects a decimal number no larger than the size of the field (65,535 for 2 octets or 4,294,967,295 for 4 octets).

When specifying community IDs for standard and extended community attributes, you can use UNIX-style regular expressions. The only exception is for VPN import policies (**vrf-import**), which do not support regular expressions for the extended communities attribute.

Regular BGP communities attributes are a variable length attribute consisting of a set of one or more 4-byte values that was split into 16 bit values. The most significant word is interpreted as an AS number and least significant word is a locally defined value assigned by the operator of the AS. Since the adoption of 4-byte ASNs, the 4-byte BGP regular community and 6-byte BGP extended community can no longer support BGP community attributes. Operators often encode AS number in the local portion of the BGP community that means that sometimes the format of the community is ASN:ASN. With the 4-byte ASN , you need 8-bytes to encode it. Although BGP extended community permits a 4-byte AS to be encoded as the global administrator filed, the local administrator field has only 2-byte of available space. Thus, 6-byte extended community attribute is also unsuitable. To overcome this, Junos OS allows you to configure optional transitive path attribute - a 12-byte BGP large community that provides the most significant 4-byte value to encode autonomous system number as the global administrator and the remaining two 4-byte assigned numbers to encode the local values as defined in RFC 8092. You can configure BGP large community at the **[edit policy-options community *community-name* members]** and **[edit routing-options static route *ip-address* community]** hierarchy levels. The BGP large community attributes format has four fields: **large**:***global administrator*:*assigned number*:*assigned number***.

NOTE: The length of the BGP large communities attribute value should be a non-zero multiple of 12.

Related Documentation

- Understanding How to Define BGP Communities and Extended Communities on page 361
- How BGP Communities and Extended Communities Are Evaluated in Routing Policy Match Conditions on page 367
- Example: Configuring a Routing Policy That Removes BGP Communities on page 413
- Example: Configuring Communities in a Routing Policy on page 372
- Example: Configuring Extended Communities in a Routing Policy on page 387
- *Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS*

Understanding How to Define BGP Communities and Extended Communities

To use a BGP community or extended community as a routing policy match condition, you define the community as described in the following sections:

- Defining BGP Communities for Use in Routing Policy Match Conditions on page 361
- Defining BGP Extended Communities for Use in Routing Policy Match Conditions on page 365

Defining BGP Communities for Use in Routing Policy Match Conditions

To create a named BGP community and define the community members, include the **community** statement:

```
[edit policy-options]
community name {
  invert-match;
  members [ community-ids ];
}
```

name identifies the community. It can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose the entire name in quotation marks (“ ”).

community-ids identifies one or more members of the community. Each community ID consists of two components, which you specify in the following format:

as-number:*community-value*;

- *as-number*—AS number of the community member. It can be a value from 0 through 65,535. You can use the following notation in specifying the AS number:

  - String of digits.
  - Asterisk (*)—A wildcard character that matches all AS numbers. (In the definition of the community attribute, the asterisk also functions as described in Table 23 on page 363.)
  - Period (.)—A wildcard character that matches any single digit in an AS number.
  - Group of AS numbers—A single AS number or a group of AS numbers enclosed in parentheses. Grouping the numbers in this way allows you to perform a common operation on the group as a whole and to give the group precedence. The grouped numbers can themselves include regular expression operators. For more information about regular expressions, see "Using UNIX Regular Expressions in Community Names" on page 362.
- *community-value*—Identifier of the community member. It can be a number from 0 through 65,535. You can use the following notation in specifying the community ID:
  - String of digits.
  - Asterisk (*)—A wildcard character that matches all community values. (In the definition of the community attribute, the asterisk also functions as described in Table 23 on page 363.)
  - Period (.)—A wildcard character that matches any single digit in a community value number.
  - Group of community value numbers—A single community value number or a group of community value numbers enclosed in parentheses. Grouping the regular expression in this way allows you to perform a common operation on the group as a whole and to give the group precedence. The grouped path can itself include regular expression operators.

You can also include one of the following well-known community names (defined in RFC 1997, *BGP Communities Attribute*) in the *community-ids* option for the **members** statement:

- no-advertise—Routes in this community name must not be advertised to other BGP peers.
- no-export—Routes in this community must not be advertised outside a BGP confederation boundary. A stand alone autonomous system that is not part of a confederation should be considered a confederation itself.
- no-export-subconfed—Routes in this community must not be advertised to external BGP peers, including peers in other members' ASs inside a BGP confederation.

Using UNIX Regular Expressions in Community Names

When specifying the members of a named BGP community (in the **members [*community-ids*]** statement), you can use UNIX-style regular expressions to specify the AS number and the member identifier. A regular expression consists of two components, which you specify in the following format:

term operator;

term identifies the string to match.

operator specifies how the term must match. Table 23 on page 363 lists the regular expression operators supported in community IDs. You place an operator immediately after *term* with no intervening space, except for the pipe (**|**) and dash (**–**) operators, which you place between two terms, and parentheses, with which you enclose terms. Table 24 on page 365 shows examples of how to define ***community-ids*** using community regular expressions. The operator is optional.

Community regular expressions are identical to the UNIX regular expressions. Both implement the extended (or modern) regular expressions as defined in POSIX 1003.2.

Community regular expressions evaluate the string specified in ***term*** on a character-by-character basis. For example, if you specify **1234:5678** as ***term***, the regular expressions see nine discrete characters, including the colon (**:**), instead of two sets of numbers (**1234** and **5678**) separated by a colon.

NOTE: In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS.

Table 23: Community Attribute Regular Expression Operators

Operator	Match Definition
{*m*,*n*}	At least ***m*** and at most ***n*** repetitions of ***term***. Both ***m*** and ***n*** must be positive integers, and ***m*** must be smaller than ***n***.
{*m*}	Exactly ***m*** repetitions of ***term***. ***m*** must be a positive integer.
{*m*,}	***m*** or more repetitions of ***term***. ***m*** must be a positive integer.
*****	Zero or more repetitions of ***term***. This is equivalent to **{0,}**.
+	One or more repetitions of ***term***. This is equivalent to **{1,}**.
?	Zero or one repetition of ***term***. This is equivalent to **{0,1}**.

Table 23: Community Attribute Regular Expression Operators *(continued)*

Operator	Match Definition
|	One of the two terms on either side of the pipe.
–	Between a starting and ending range, inclusive.
^	Character at the beginning of a community attribute regular expression. If you omit the ^ character, it is implicitly added. We recommend explicit use of this operator for the clearest interpretation of your configuration.
$	Character at the end of a community attribute regular expression. If you omit the $ character, it is implicitly added. We recommend explicit use of this operator for the clearest interpretation of your configuration.
[ ]	Set of characters. One character from the set can match. To specify the start and end of a range, use a hyphen (-). To specify a set of characters that do not match, use the caret (^) as the first character after the opening square bracket ([).
( )	Group of terms that are enclosed in parentheses. If enclosed in quotation marks with no intervening space ("()"), indicates a null. Intervening space between the parentheses and the terms is ignored.

Table 23: Community Attribute Regular Expression Operators *(continued)*

Operator	Match Definition
“ ”	Characters (such as space, tab, question mark, and bracket) that are enclosed within quotation marks in a community attribute regular expression indicate special characters.

Table 24: Examples of Community Attribute Regular Expressions

Community Attribute to Match	[illegible]	[illegible]
AS number is 56 or 78. Community value is any number.	[illegible]	[illegible]
AS number is 56. Community value is any number that starts with 2.	[illegible]	[illegible]
AS number is any number. Community value is any number that ends with 5, 7, or 9.	[illegible]	[illegible]
AS number is 56 or 78. Community value is any number that starts with 2 and ends with 2 through 8.	[illegible]	[illegible]

Defining BGP Extended Communities for Use in Routing Policy Match Conditions

To create a named BGP community and define the community members, include the **community** statement:

```
[edit policy-options]
community name {
   members [ community-ids ];
}
```

name identifies the community. It can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose the entire name in quotation marks (“ ”).

community-ids identifies one or more members of the community. Each community ID consists of three components, which you specify in the following format:

type:*administrator*:*assigned-number*

type is the type of extended community and can be either the 16-bit numerical identifier of a specific BGP extended community or one of these types:

- **bandwidth**—Sets up the bandwidth extended community. Specifying link bandwidth allows you to distribute traffic unequally among different BGP paths.

  NOTE: The link bandwidth attribute does not work concurrently with per-prefix load balancing.

- **domain-id**—Identifies the OSPF domain from which the route originated.
- **origin**—Identifies where the route originated.
- **rt-import**—Identifies the route to install in the routing table.

  NOTE: You must identify the route by an IP address, not an AS number.

- **src-as**—Identifies the AS from which the route originated. You must specify an AS number, not an IP address.

  NOTE: You must identify the AS by an AS number, not an IP address.

- **target**—Identifies the destination to which the route is going.

  NOTE: For an import policy for a VPN routing and forwarding (VRF) instance, you must include at least one route target. Additionally, you cannot use wildcard characters or regular expressions in the route target for a VRF import policy. Each value you configure for a route target for a VRF import policy must be a single value.

administrator is the administrator. It is either an AS number or an IP version 4 (IPv4) address prefix, depending on the type of extended community.

assigned-number identifies the local provider.

In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS. In plain-number format, you can configure a value in the range from 1 through 4,294,967,295. To configure a **target** or **origin** extended community that includes a 4-byte AS number in the plain-number format, append the letter “L” to the end of number. For example, a target community with the 4-byte AS number 334,324 and an assigned number of 132 is represented as **target:334324L:132**.

In Junos OS Release 9.2 and later, you can also use AS-dot notation when defining a 4-byte AS number for the **target** and **origin** extended communities. Specify two integers joined by a period: *16-bit high-order value in decimal*.*16-bit low-order value in decimal*. For example, the 4-byte AS number represented in plain-number format as 65546 is represented in AS-dot notation as 1.10.

Examples: Defining BGP Extended Communities

Configure a target community with an administrative field of **10458** and an assigned number of **20**:

```
[edit policy-options]
community test-a members [ target:10458:20 ];
```

Configure a target community with an administrative field of 10.1.1.1 and an assigned number of 20:

```
[edit policy-options]
community test-a members [ target:10.1.1.1:20 ];
```

Configure an origin community with an administrative field of 10.1.1.1 and an assigned number of 20:

```
[edit policy-options]
community test-a members [ origin:10.1.1.1:20 ];
```

Configure a target community with a 4-byte AS number in the administrative field of 100000 and an assigned number of 130:

```
[edit policy-options]
community test-b members [ target:100000L:130 ];
```

Related Documentation

- Example: Configuring Communities in a Routing Policy on page 372
- Example: Configuring Extended Communities in a Routing Policy on page 387

How BGP Communities and Extended Communities Are Evaluated in Routing Policy Match Conditions

When you use BGP communities and extended communities as match conditions in a routing policy, the policy framework software evaluates them as follows:

- Each route is evaluated against each named community in a routing policy **from** statement. If a route matches one of the named communities in the **from** statement, the evaluation of the current term continues. If a route does not match, the evaluation of the current term ends.
- The route is evaluated against each member of a named community. The evaluation of all members must be successful for the named community evaluation to be successful.
- Each member in a named community is identified by either a literal community value or a regular expression. Each member is evaluated against each community associated with the route. (Communities are an unordered property of a route. For example, 1:2 3:4

is the same as 3:4 1:2.) Only one community from the route is required to match for the member evaluation to be successful.

- Community regular expressions are evaluated on a character-by-character basis. For example, if a route contains community 1234:5678, the regular expressions see nine discrete characters, including the colon (:), instead of two sets of numbers (1234 and 5678) separated by a colon. For example:

```
[edit]
policy-options {
  policy-statement one {
    from {
      community [comm-one comm-two];
    }
  }
  community comm-one members [ 1:2 "^4:(5|6)$" ];
  community comm-two members [ 7:8 9:10 ];
}
```

  If a community member is a regular expression, a string match is made rather than a numeric match.

  For example:

```
community example1 members 100:100
community example2 members 100:1..
```

  Given a route with a community value of 1100:100, this route matches **community example2** but not **example1**.

- To match routing policy **one**, the route must match either **comm-one** or **comm-two**.
- To match **comm-one**, the route must have a community that matches 1:2 and a community that matches 4:5 or 4:6.
- To match **comm-two**, the route must have a community that matches 7:8 and a community that matches 9:10.

Multiple Matches

When multiple matches are found, label aggregation does not happen. Consider the following configuration:

```
family inet-vpn {
  unicast {
    aggregate-label {
      community community-name;
    }
  }
}

family inet-vpn {
  labeled-unicast {
    aggregate-label {
      community community-name;
    }
  }
```



```
}
```

Suppose, for instance, that two routes are received with community attributes **target:65000:1000 origin:65200:2000** and that the community name is **"5...:.*"**. In this case, both the extended community attributes, **target:65000:1000** and **origin:65200:2000** match the regular expression of the community name. In this case, label aggregation does not occur. In the following example, the **Label operation** field shows that the labels are not aggregated.

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101056
                Push 101056
                Communities: target:65000:1000 origin:65200:2000
```

You can resolve this issue in either of the following ways:

- Be more specific in the regular expression if the site-of-origin extended community attribute does not overlap with the target one.
- Specify the site of origin in the community name.

Both methods are shown in the following examples.

Be More Specific in the Regular Expression

```
user@host# set policy-options community community-name members "52..:.*"
user@host# commit
```

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
```

Specify the Site of Origin in the Community Name

```
user@host# set policy-options community community-name members "origin:65...:.*"
user@host# commit
```

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
```

Inverting Community Matches

The **community** match condition defines a regular expression and if it matches the community attribute of the received prefix, Junos OS returns a TRUE result. If not, Junos OS returns a FALSE result. The **invert-match** statement makes Junos OS behave to the contrary. If there is a match, Junos OS returns a FALSE result. If there is no match, Junos OS returns a TRUE result. To invert the results of the community expression matching, include the **invert-match** statement in the community configuration.

[edit policy-options community *name*]
invert-match;

Extended Community Type

The extended community type is not taken into account by regular expressions. Consider, for instance, the following community attributes and community name.

Communities:

- 5200:1000
- **target:65000:1000**
- **origin:65200:2000**

Community attribute:

- community-name members "5...:.*"

In this case, both extended community attribute, **5200:1000** and the extended community attribute, **origin:65200:2000**, match the regular expression of the community name. Therefore, the label aggregation does not occur, as shown here:

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: 5200:1000 target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101056
                Push 101056
                Communities: 5200:1000 target:65000:1000 origin:65200:2000
```

You can resolve this issue by using a more specific regular expression. For example, you can use the anchor character (^) to bind the location of the digits, as shown here:

```
user@host# set policy-options community community-name members "^5...:.*"
user@host# commit
```

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: 5200:1000 target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
```



```
                Label operation: Push 101040
                Push 101040
                Communities: 5200:1000 target:65000:1000 origin:65200:2000
```

Multiple Communities Are Matched with Ex-OR Logic

This differs from the AND matching logic used for non-extended communities in BGP.

If, for instance, four routes are received with two sets of community attributes, the regular expression might match both community attributes. Consider the following example:

- Communities—5200:1000 target:65000:1000
- Communities—target:65000:1000 origin:65200:2000
- Community attribute—community community-name member ["^5...:.*" origin:65.*:.*]

Both labels are aggregated, as shown here:

```
user@host> show route table VPN detail | match "^10 | Communities | Push"
10.1.1.0/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000
10.1.1.4/30 (1 entry, 1 announced)
                Label operation: Push 101040
                Push 101040
                Communities: target:65000:1000 origin:65200:2000

10.1.1.16/30 (1 entry, 1 announced)
                Label operation: Push 121104
                Push 101104
                Communities: 5200:1000 target:65000:1000
10.1.1.20/30 (1 entry, 1 announced)
                Label operation: Push 121104
                Push 101104
                Communities: 5200:1000 target:65000:1000
```

A more complete example of community values is shown here:

```
user@host> show policy-options community community-name
members [ "(^1...:*)|(^3...:*)|(^4...:*)" origin:2.*:* origin:3.*:* origin:6.*:*
 ]
```

This regular expression matches community values starting with 1, 3, or 4, and matches extended community values of type origin whose administrative value starts with 2, 3, or 6.

Including BGP Communities and Extended Communities in Routing Policy Match Conditions

To include a BGP community or extended community in a routing policy match condition, include the **community** condition in the **from** statement of a policy term:

```
from {
  community [ names ];
```

```
}
```

Additionally, you can explicitly exclude BGP community information with a static route by using the **none** option. Include this option when configuring an individual route in the **route** portion to override a community option specified in the **defaults** portion.

You can include the names of multiple communities in the **community** match condition. If you do this, only one community needs to match for a match to occur (matching is effectively a logical OR operation).

Related Documentation

- Using UNIX Regular Expressions in Community Names on page 362
- Example: Configuring Communities in a Routing Policy on page 372
- Example: Configuring Extended Communities in a Routing Policy on page 387
- Example: Configuring a Routing Policy That Removes BGP Communities on page 413
- Example: Configuring a Routing Policy Based on the Number of BGP Communities on page 405

Example: Configuring Communities in a Routing Policy

A community is a route attribute used by BGP to administratively group routes with similar properties.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

One main role of the community attribute is to be an administrative tag value used to associate routes together. Generally, these routes share some common properties, but that is not required. Communities are a flexible tool within BGP. An individual community value can be assigned to a single route or multiple routes. A route can be assigned a single community value or multiple values. Networks use the community attribute to assist in implementing administrative routing policies. A route's assigned value can allow it to be accepted into the network, or rejected from the network, or allow it to modify attributes.

Figure 32 on page 374 shows Device R1, Device R2, and Device R3 as internal BGP (IBGP) peers in autonomous system (AS) 64510. Device R4 is advertising the 172.16.0.0/21 address space from AS 64511. The specific routes received by Device R1 from Device R4 are as follows:

```
user@R1> show route receive-protocol bgp 10.0.0.13
inet.0: 20 destinations, 28 routes (20 active, 0 holddown, 8 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.0.0/24           10.0.0.13                               64511 I
* 172.16.1.0/24           10.0.0.13                               64511 I
* 172.16.2.0/24           10.0.0.13                               64511 I
* 172.16.3.0/24           10.0.0.13                               64511 I
* 172.16.4.0/24           10.0.0.13                               64511 I
* 172.16.5.0/24           10.0.0.13                               64511 I
* 172.16.6.0/24           10.0.0.13                               64511 I
* 172.16.7.0/24           10.0.0.13                               64511 I
```

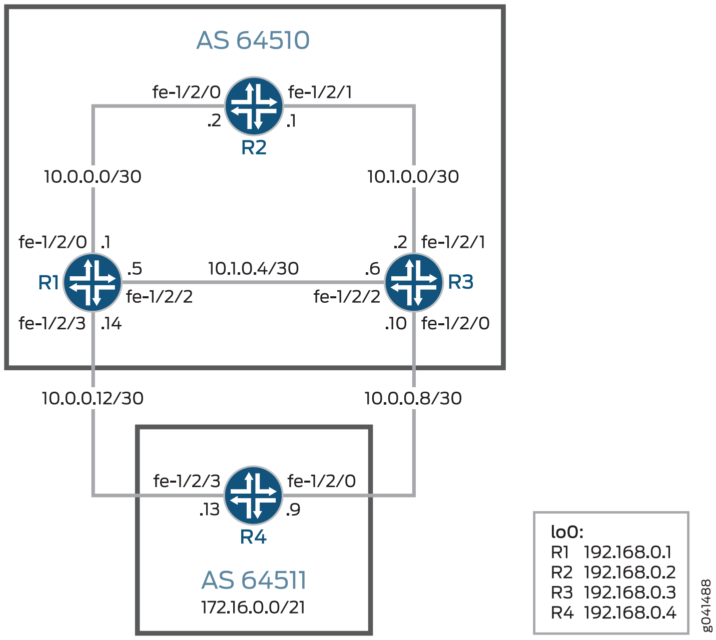
The administrators of AS 64511 want to receive certain user traffic from Device R1, and other user traffic from Device R3. To accomplish this administrative goal, Device R4 attaches the community value of 64511:1 to some routes that it sends and attaches the community value 64511:3 to other routes that it sends. Routing policies within AS 64510 are configured using a community match criterion to change the local preference of the received routes to new values that alter the BGP route selection algorithm. The route with the highest local preference value is preferred.

On Device R1, routes with the 64511:1 community value are assigned a local preference of 200, and routes with the 64511:3 community value are assigned a local preference of 50. On Device R3, the reverse is done so that routes with the 64511:3 community value are assigned a local preference of 200, and routes with the 64511:1 community value are assigned a local preference of 50. This information is then communicated through IBGP by both Device R1 and Device R3 to Device R2.

Topology

Figure 32 on page 374 shows the sample network.

Figure 32: Topology for Regular BGP Communities



"CLI Quick Configuration" on page 374 shows the configuration for all of the devices in Figure 32 on page 374.

The section "Step-by-Step Procedure" on page 377 describes the steps on Device R1 and R4.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/2 unit 0 family inet address 10.1.0.5/30
set interfaces fe-1/2/3 unit 0 family inet address 10.0.0.14/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.1
set protocols bgp group int neighbor 192.168.0.2 export send-direct
set protocols bgp group int neighbor 192.168.0.3
set protocols bgp group ext type external
set protocols bgp group ext import change-local-preference
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.13
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement change-local-preference term find-R1-routes from
  community from-R1
```

```
set policy-options policy-statement change-local-preference term find-R1-routes then
  local-preference 200
set policy-options policy-statement change-local-preference term find-R3-routes from
  community from-R3
set policy-options policy-statement change-local-preference term find-R3-routes then
  local-preference 50
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.0.0.12/30
  exact
set policy-options policy-statement send-direct term 1 then accept
set policy-options community from-R1 members 64511:1
set policy-options community from-R3 members 64511:3
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.2
set protocols bgp group int neighbor 192.168.0.1
set protocols bgp group int neighbor 192.168.0.3
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/2 unit 8 family inet address 10.1.0.6/30
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.10/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group int type internal
set protocols bgp group int local-address 192.168.0.3
set protocols bgp group int neighbor 192.168.0.1
set protocols bgp group int neighbor 192.168.0.2 export send-direct
set protocols bgp group ext type external
set protocols bgp group ext import change-local-preference
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.9
set protocols ospf area 0.0.0.0 interface fe-1/2/1.0
set protocols ospf area 0.0.0.0 interface fe-1/2/2.8
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement change-local-preference term find-R3-routes from
  community from-R3
set policy-options policy-statement change-local-preference term find-R3-routes then
  local-preference 200
set policy-options policy-statement change-local-preference term find-R1-routes from
  community from-R1
set policy-options policy-statement change-local-preference term find-R1-routes then
  local-preference 50
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.0.0.8/30 exact
set policy-options policy-statement send-direct term 1 then accept
```

```
set policy-options community from-R1 members 64511:1
set policy-options community from-R3 members 64511:3
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64510
```

Device R4

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.9/30
set interfaces fe-1/2/3 unit 0 family inet address 10.0.0.13/30
set interfaces lo0 unit 0 family inet address 192.168.0.4/32
set protocols bgp group to-R1 type external
set protocols bgp group to-R1 export send-static
set protocols bgp group to-R1 peer-as 64510
set protocols bgp group to-R1 neighbor 10.0.0.14
set protocols bgp group to-R3 type external
set protocols bgp group to-R3 export send-static
set protocols bgp group to-R3 peer-as 64510
set protocols bgp group to-R3 neighbor 10.0.0.10
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 from route-filter 172.16.0.0/24
  exact
set policy-options policy-statement send-static term 1 from route-filter 172.16.1.0/24
  exact
set policy-options policy-statement send-static term 1 from route-filter 172.16.2.0/24
  exact
set policy-options policy-statement send-static term 1 from route-filter 172.16.3.0/24
  exact
set policy-options policy-statement send-static term 1 then community add from-R1
set policy-options policy-statement send-static term 1 then accept
set policy-options policy-statement send-static term 2 from protocol static
set policy-options policy-statement send-static term 2 from route-filter 172.16.4.0/24
  exact
set policy-options policy-statement send-static term 2 from route-filter 172.16.5.0/24
  exact
set policy-options policy-statement send-static term 2 from route-filter 172.16.6.0/24
  exact
set policy-options policy-statement send-static term 2 from route-filter 172.16.7.0/24
  exact
set policy-options policy-statement send-static term 2 then community add from-R3
set policy-options policy-statement send-static term 2 then accept
set policy-options policy-statement send-static term 3 then reject
set policy-options community from-R1 members 64511:1
set policy-options community from-R3 members 64511:3
set routing-options static route 172.16.0.0/24 reject
set routing-options static route 172.16.1.0/24 reject
set routing-options static route 172.16.2.0/24 reject
set routing-options static route 172.16.3.0/24 reject
set routing-options static route 172.16.4.0/24 reject
set routing-options static route 172.16.5.0/24 reject
set routing-options static route 172.16.6.0/24 reject
set routing-options static route 172.16.7.0/24 reject
set routing-options router-id 192.168.0.4
set routing-options autonomous-system 64511
```


Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set fe-1/2/0 unit 0 family inet address 10.0.0.1/30

user@R1# set fe-1/2/2 unit 0 family inet address 10.1.0.5/30

user@R1# set fe-1/2/3 unit 0 family inet address 10.0.0.14/30

user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure internal gateway protocol (IGP) connections to Device R2 and Device R3.

```
[edit protocols ospf area 0.0.0.0]
user@R1# set interface fe-1/2/0.0
user@R1# set interface fe-1/2/2.0
user@R1# set interface lo0.0 passive
```

3. Configure the IBGP connections to Device R2 and Device R3.

```
[edit protocols bgp group int]
user@R1# set type internal
user@R1# set local-address 192.168.0.1
user@R1# set neighbor 192.168.0.2 export send-direct
user@R1# set neighbor 192.168.0.3
```

4. Configure the EBGP connection to Device R4.

```
[edit protocols bgp group ext]
user@R1# set type external
user@R1# set import change-local-preference
user@R1# set peer-as 64511
user@R1# set neighbor 10.0.0.13
```

5. Configure the policy **send-direct**.

   This policy is referenced in the IBGP connection to Device R2 and enables Device R2 to have external reachability. An alternative is to configure a **next-hop self** policy on Device R1 and Device R3.

```
[edit policy-options policy-statement send-direct term 1]
user@R1# set from protocol direct
user@R1# set from route-filter 10.0.0.12/30 exact
user@R1# set then accept
```

6. Configure the policy that changes the local preference for routes with specified community tags.

```
[edit policy-options policy-statement change-local-preference]
user@R1# set term find-R1-routes from community from-R1
user@R1# set term find-R1-routes then local-preference 200
user@R1# set term find-R3-routes from community from-R3
user@R1# set term find-R3-routes then local-preference 50

[edit policy-options ]
user@R1# set community from-R1 members 64511:1
user@R1# set community from-R3 members 64511:3
```

7. Configure the autonomous system (AS) number and router ID.

```
[edit routing-options]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 64510
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R4:

1. Configure the interfaces.

```
[edit interfaces]
user@R4# set fe-1/2/0 unit 0 family inet address 10.0.0.9/30

user@R4# set fe-1/2/3 unit 0 family inet address 10.0.0.13/30

user@R4# set lo0 unit 0 family inet address 192.168.0.4/32
```

2. Configure the EBGP connection to Device R1 and Device R3.

```
[edit protocols bgp]
user@R4# set group to-R1 type external
user@R4# set group to-R1 export send-static
user@R4# set group to-R1 peer-as 64510
user@R4# set group to-R1 neighbor 10.0.0.14

user@R4# set group to-R3 type external
user@R4# set group to-R3 export send-static
user@R4# set group to-R3 peer-as 64510
user@R4# set group to-R3 neighbor 10.0.0.10
```

3. Configure the community tags.

```
[edit policy-options ]
user@R4# set community from-R1 members 64511:1
user@R4# set community from-R3 members 64511:3
```

4. Configure the policy **send-static**.

This policy is referenced in the EBGP connections to Device R1 and Device R3. The policy attaches the 64511:1 (from-R1) community to some routes and the 64511:3 (from-R3) community to other routes.

```
[edit policy-options policy-statement send-static term 1]
user@R4# set from protocol static
user@R4# set from route-filter 172.16.0.0/24 exact
user@R4# set from route-filter 172.16.1.0/24 exact
user@R4# set from route-filter 172.16.2.0/24 exact
user@R4# set from route-filter 172.16.3.0/24 exact
user@R4# set then community add from-R1
user@R4# set then accept

[edit policy-options policy-statement send-static term 2]
user@R4# set from protocol static
user@R4# set from route-filter 172.16.4.0/24 exact
user@R4# set from route-filter 172.16.5.0/24 exact
user@R4# set from route-filter 172.16.6.0/24 exact
user@R4# set from route-filter 172.16.7.0/24 exact
user@R4# set then community add from-R3
user@R4# set then accept

[edit policy-options policy-statement send-static term 3]
user@R4# set then reject
```

5. Configure the static routes.

```
[edit routing-options static]
user@R4# set route 172.16.0.0/24 reject
user@R4# set route 172.16.1.0/24 reject
user@R4# set route 172.16.2.0/24 reject
user@R4# set route 172.16.3.0/24 reject
user@R4# set route 172.16.4.0/24 reject
user@R4# set route 172.16.5.0/24 reject
user@R4# set route 172.16.6.0/24 reject
user@R4# set route 172.16.7.0/24 reject
```

6. Configure the autonomous system (AS) number and router ID.

```
[edit routing-options]
user@R4# set router-id 192.168.0.4
user@R4# set autonomous-system 64511
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device R1

```
user@R1# show interfaces
fe-1/2/0 {
   unit 0 {
      family inet {
```

```
            address 10.0.0.1/30;
        }
    }
}
fe-1/2/2 {
    unit 0 {
        family inet {
            address 10.1.0.5/30;
        }
    }
}
fe-1/2/3 {
    unit 0 {
        family inet {
            address 10.0.0.14/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.1/32;
        }
    }
}
```

```
user@R1# show protocols
bgp {
    group int {
        type internal;
        local-address 192.168.0.1;
        neighbor 192.168.0.2 {
            export send-direct;
        }
        neighbor 192.168.0.3;
    }
    group ext {
        type external;
        import change-local-preference;
        peer-as 64511;
        neighbor 10.0.0.13;
    }
}
ospf {
    area 0.0.0.0 {
        interface fe-1/2/0.0;
        interface fe-1/2/2.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@R1# show policy-options
policy-statement change-local-preference {
    term find-R1-routes {
```



```
      from community from-R1;
      then {
        local-preference 200;
      }
    }
    term find-R3-routes {
      from community from-R3;
      then {
        local-preference 50;
      }
    }
  }
  policy-statement send-direct {
    term 1 {
      from {
        protocol direct;
        route-filter 10.0.0.12/30 exact;
      }
      then accept;
    }
  }
  community from-R1 members 64511:1;
  community from-R3 members 64511:3;
```

```
user@R1# show routing-options
router-id 192.168.0.1;
autonomous-system 64510;
```

Device R4

```
user@R4# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      address 10.0.0.9/30;
    }
  }
}
fe-1/2/3 {
  unit 0 {
    family inet {
      address 10.0.0.13/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.4/32;
    }
  }
}
```

```
user@R4# show protocols
bgp {
  group to-R1 {
    type external;
    export send-static;
```

```
        peer-as 64510;
        neighbor 10.0.0.14;
    }
    group to-R3 {
        type external;
        export send-static;
        peer-as 64510;
        neighbor 10.0.0.10;
    }
}
```

```
user@R4# show policy-options
policy-statement send-static {
    term 1 {
        from {
            protocol static;
            route-filter 172.16.0.0/24 exact;
            route-filter 172.16.1.0/24 exact;
            route-filter 172.16.2.0/24 exact;
            route-filter 172.16.3.0/24 exact;
        }
        then {
            community add from-R1;
            accept;
        }
    }
    term 2 {
        from {
            protocol static;
            route-filter 172.16.4.0/24 exact;
            route-filter 172.16.5.0/24 exact;
            route-filter 172.16.6.0/24 exact;
            route-filter 172.16.7.0/24 exact;
        }
        then {
            community add from-R3;
            accept;
        }
    }
    term 3 {
        then reject;
    }
}
community from-R1 members 64511:1;
community from-R3 members 64511:3;
```

```
user@R4# show routing-options
static {
    route 172.16.0.0/24 reject;
    route 172.16.1.0/24 reject;
    route 172.16.2.0/24 reject;
    route 172.16.3.0/24 reject;
    route 172.16.4.0/24 reject;
    route 172.16.5.0/24 reject;
    route 172.16.6.0/24 reject;
    route 172.16.7.0/24 reject;
}
```

```
router-id 192.168.0.4;
autonomous-system 64511;
```

If you are done configuring the devices, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Verifying the Routes Sent on Device R4 on page 383
- Verifying the Routes Received on Device R2 on page 385

Verifying the Routes Sent on Device R4

Purpose On Device R4, check the routes sent to Device R1 and Device R3.

Action

`user@R4>` **show route advertising-protocol bgp 10.0.0.14**

```
inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
* 172.16.0.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.1.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.2.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.3.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.4.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.5.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.6.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.7.0/24 (1 entry, 1 announced)
 BGP group to-R1 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3
```

`user@R2>` **show route advertising-protocol bgp 10.0.0.10**

```
inet.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
* 172.16.0.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1
```



```
* 172.16.1.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.2.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.3.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:1

* 172.16.4.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.5.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.6.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3

* 172.16.7.0/24 (1 entry, 1 announced)
 BGP group to-R3 type External
     Nexthop: Self
     AS path: [64511] I
     Communities: 64511:3
```

Meaning Device R4 has tagged the routes with the communities 64511:1 and 64511:3 and sent them to Device R1 and R3.

Verifying the Routes Received on Device R2

Purpose On Device R2, check the routes received from Device R1 and Device R3.

Action

```
user@R2> show route receive-protocol bgp 192.168.0.1

inet.0: 22 destinations, 30 routes (22 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 10.0.0.12/30            192.168.0.1                  100        I
* 172.16.0.0/24           10.0.0.13                    200        64511 I
* 172.16.1.0/24           10.0.0.13                    200        64511 I
* 172.16.2.0/24           10.0.0.13                    200        64511 I
* 172.16.3.0/24           10.0.0.13                    200        64511 I
  172.16.4.0/24           10.0.0.13                    50         64511 I
  172.16.5.0/24           10.0.0.13                    50         64511 I
  172.16.6.0/24           10.0.0.13                    50         64511 I
  172.16.7.0/24           10.0.0.13                    50         64511 I
```

```
user@R2> show route match-prefix 172.16.*

inet.0: 22 destinations, 30 routes (22 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.0.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
                    [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
172.16.1.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
                    [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
172.16.2.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
                    [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
172.16.3.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
                    [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
172.16.4.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.3
                      AS path: 64511 I
                      to 10.1.0.2 via fe-1/2/1.0
                    > to 10.1.0.6 via fe-1/2/0.7
                    [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.1
                      AS path: 64511 I
                      to 10.0.0.1 via fe-1/2/0.0
                    > to 10.1.0.5 via fe-1/2/0.6
172.16.5.0/24      *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.3
```



```
                              AS path: 64511 I
                              to 10.1.0.2 via fe-1/2/1.0
                            > to 10.1.0.6 via fe-1/2/0.7
                            [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.1
                              AS path: 64511 I
                              to 10.0.0.1 via fe-1/2/0.0
                            > to 10.1.0.5 via fe-1/2/0.6
172.16.6.0/24              *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.3
                              AS path: 64511 I
                              to 10.1.0.2 via fe-1/2/1.0
                            > to 10.1.0.6 via fe-1/2/0.7
                            [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.1
                              AS path: 64511 I
                              to 10.0.0.1 via fe-1/2/0.0
                            > to 10.1.0.5 via fe-1/2/0.6
172.16.7.0/24              *[BGP/170] 1d 00:47:39, localpref 200, from 192.168.0.3
                              AS path: 64511 I
                              to 10.1.0.2 via fe-1/2/1.0
                            > to 10.1.0.6 via fe-1/2/0.7
                            [BGP/170] 1d 00:47:39, localpref 50, from 192.168.0.1
                              AS path: 64511 I
                              to 10.0.0.1 via fe-1/2/0.0
                            > to 10.1.0.5 via fe-1/2/0.6
```

Meaning Device R2 has the routes with the expected local preferences and the expected active routes, as designated by the asterisks (*).

Related Documentation

- Example: Configuring Extended Communities in a Routing Policy on page 387
- Example: Configuring a Routing Policy That Removes BGP Communities on page 413
- Example: Configuring a Routing Policy Based on the Number of BGP Communities on page 405
- *Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS*

Example: Configuring Extended Communities in a Routing Policy

An extended community is similar in most ways to a regular community. Some networking implementations, such as virtual private networks (VPNs), use extended communities because the 4-octet regular community value does not provide enough expansion and flexibility. An extended community is an eight-octet value divided into two main sections.

Requirements

No special configuration beyond device initialization is required before configuring this example.

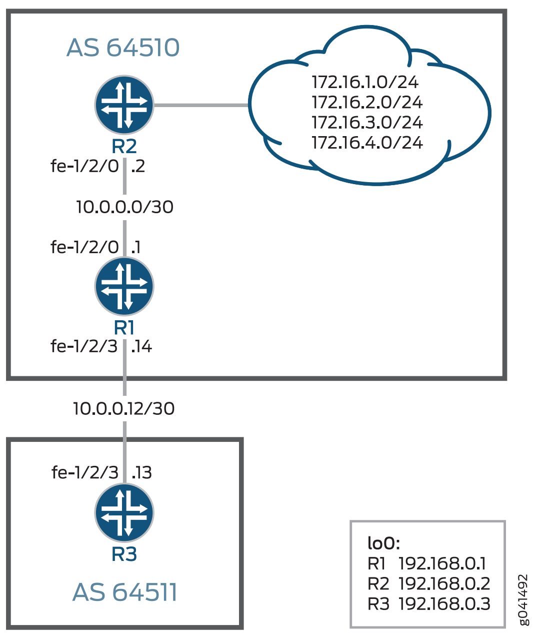
Overview

In this example, Device R1 and Device R2 are OSPF neighbors in autonomous system (AS) 64510. Device R3 has an external BGP (EBGP) connection to Device R1. Device R2 has customer networks in the 172.16/16 address space, simulated with addresses on its loopback interface (lo0). Device R1 has static routes to several 172.16.*x*/24 networks, and attaches regular community values to these routes. Device R1 then uses an export policy to advertise the routes to Device R3. Device R3 receives these routes and uses an import policy to add extended community values to the routes.

Topology

Figure 33 on page 388 shows the sample network.

Figure 33: Topology for Extended BGP Communities



"CLI Quick Configuration" on page 388 shows the configuration for all of the devices in Figure 33 on page 388.

The section "Step-by-Step Procedure" on page 390 describes the steps on Device R3.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/3 unit 0 family inet address 10.0.0.14/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32 primary
set protocols bgp group ext type external
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.13
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 172.16.1.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.1.0/24 community 64510:1
set routing-options static route 172.16.2.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.2.0/24 community 64510:2
set routing-options static route 172.16.3.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.3.0/24 community 64510:3
set routing-options static route 172.16.4.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.4.0/24 community 64510:4
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces lo0 unit 0 family inet address 172.16.1.1/32
set interfaces lo0 unit 0 family inet address 172.16.2.2/32
set interfaces lo0 unit 0 family inet address 172.16.3.3/32
set interfaces lo0 unit 0 family inet address 172.16.4.4/32
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
```

Device R3

```
set interfaces fe-1/2/3 unit 0 family inet address 10.0.0.13/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group to-R1 type external
set protocols bgp group to-R1 import set-ext-comms
set protocols bgp group to-R1 peer-as 64510
set protocols bgp group to-R1 neighbor 10.0.0.14
set policy-options policy-statement set-ext-comms term route-1 from route-filter
  172.16.1.0/24 exact
set policy-options policy-statement set-ext-comms term route-1 then community add
  target-as
set policy-options policy-statement set-ext-comms term route-1 then accept
set policy-options policy-statement set-ext-comms term route-2 from route-filter
  172.16.2.0/24 exact
set policy-options policy-statement set-ext-comms term route-2 then community add
  target-ip
set policy-options policy-statement set-ext-comms term route-2 then accept
set policy-options policy-statement set-ext-comms term route-3 from route-filter
  172.16.3.0/24 exact
set policy-options policy-statement set-ext-comms term route-3 then community add
  origin-as
set policy-options policy-statement set-ext-comms term route-3 then accept
```

```
set policy-options policy-statement set-ext-comms term route-4 from route-filter
  172.16.4.0/24 exact
set policy-options policy-statement set-ext-comms term route-4 then community add
  origin-ip
set policy-options policy-statement set-ext-comms term route-4 then accept
set policy-options community origin-as members origin:64511:3
set policy-options community origin-ip members origin:172.16.7.7:4
set policy-options community target-as members target:64511:1
set policy-options community target-ip members target:172.16.7.7:2
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64511
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R3:

1. Configure the interfaces.

```
[edit interfaces]
user@R3# set fe-1/2/3 unit 0 family inet address 10.0.0.13/30

user@R3# set lo0 unit 0 family inet address 192.168.0.3/32
```

2. Configure the EBGP connection to Device R1.

```
[edit protocols bgp group to-R1]
user@R3# set type external
user@R3# set import set-ext-comms
user@R3# set peer-as 64510
user@R3# set neighbor 10.0.0.14
```

3. Configure the policy that adds extended community values to the routes received from Device R1.

   An extended community uses a notation of ***type*:*administrator*:*assigned-number***.

   The specific community values can be anything that accomplishes your administrative goals, within certain parameters, as explained in community.

```
[edit policy-options policy-statement set-ext-comms]
user@R3# set term route-1 from route-filter 172.16.1.0/24 exact
user@R3# set term route-1 then community add target-as
user@R3# set term route-1 then accept

user@R3# set term route-2 from route-filter 172.16.2.0/24 exact
user@R3# set term route-2 then community add target-ip
user@R3# set term route-2 then accept

user@R3# set term route-3 from route-filter 172.16.3.0/24 exact
user@R3# set term route-3 then community add origin-as
user@R3# set term route-3 then accept
```



```
user@R3# set term route-4 from route-filter 172.16.4.0/24 exact
user@R3# set term route-4 then community add origin-ip
user@R3# set term route-4 then accept

[edit policy-options]
user@R3# set community origin-as members origin:64511:3
user@R3# set community origin-ip members origin:172.16.7.7:4
user@R3# set community target-as members target:64511:1
user@R3# set community target-ip members target:172.16.7.7:2
```

4. Configure the autonomous system (AS) number and router ID.

```
[edit routing-options]
user@R3# set router-id 192.168.0.3
user@R3# set autonomous-system 64511
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R3# show interfaces
fe-1/2/3 {
  unit 0 {
    family inet {
      address 10.0.0.13/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.3/32;
    }
  }
}

user@R3# show protocols
bgp {
  group to-R1 {
    type external;
    import set-ext-comms;
    peer-as 64510;
    neighbor 10.0.0.14;
  }
}

user@R3# show policy-options
policy-statement set-ext-comms {
  term route-1 {
    from {
      route-filter 172.16.1.0/24 exact;
    }
    then {
```

```
            community add target-as;
            accept;
        }
    }
    term route-2 {
        from {
            route-filter 172.16.2.0/24 exact;
        }
        then {
            community add target-ip;
            accept;
        }
    }
    term route-3 {
        from {
            route-filter 172.16.3.0/24 exact;
        }
        then {
            community add origin-as;
            accept;
        }
    }
    term route-4 {
        from {
            route-filter 172.16.4.0/24 exact;
        }
        then {
            community add origin-ip;
            accept;
        }
    }
}
community origin-as members origin:64511:3;
community origin-ip members origin:172.16.7.7:4;
community target-as members target:64511:1;
community target-ip members target:172.16.7.7:2;
```

```
user@R3# show routing-options
router-id 192.168.0.3;
autonomous-system 64511;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Verifying the Routes on Device R1 on page 392
- Verifying the Routes on Device R3 on page 394

Verifying the Routes on Device R1

Purpose On Device R1, check the 172.16. routes in the routing table.

Action `user@R1>` **show route protocol static match-prefix 172.16.* detail**

```
inet.0: 15 destinations, 15 routes (15 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 835
                Address: 0x9260250
                Next-hop reference count: 19
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                State: <Active Int Ext>
                Local AS: 64510
                Age: 2:06:08
                Task: RT
                Announcement bits (2): 2-KRT 3-BGP_RT_Background
                AS path: I
                Communities: 64510:1

172.16.2.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 835
                Address: 0x9260250
                Next-hop reference count: 19
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                State: <Active Int Ext>
                Local AS: 64510
                Age: 2:06:08
                Task: RT
                Announcement bits (2): 2-KRT 3-BGP_RT_Background
                AS path: I
                Communities: 64510:2

172.16.3.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 835
                Address: 0x9260250
                Next-hop reference count: 19
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                State: <Active Int Ext>
                Local AS: 64510
                Age: 2:06:08
                Task: RT
                Announcement bits (2): 2-KRT 3-BGP_RT_Background
                AS path: I
                Communities: 64510:3

172.16.4.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 835
                Address: 0x9260250
                Next-hop reference count: 19
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                State: <Active Int Ext>
                Local AS: 64510
                Age: 2:06:08
                Task: RT
                Announcement bits (2): 2-KRT 3-BGP_RT_Background
                AS path: I
                Communities: 64510:4
```


Meaning The output shows that the regular community values are attached to the routes.

NOTE: The communities are attached to static routes, thus demonstrating that communities can be attached to non-BGP routes.

Verifying the Routes on Device R3

Purpose On Device R3, check the 172.16. routes in the routing table.

Action user@R3> **show route protocol bgp match-prefix 172.16.* detail**

```
betsy@tp5# run show route protocol bgp match-prefix 172.16.* detail logical-system
 R3

inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 611
                Address: 0x9260130
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via fe-1/2/3.0, selected
                State: <Active Ext>
                Local AS: 64511 Peer AS: 64510
                Age: 1:57:27
                Task: BGP_64510.10.0.0.14+54618
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:1 target:64511:1
                Accepted
                Localpref: 100
                Router ID: 192.168.0.1

172.16.2.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 611
                Address: 0x9260130
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via fe-1/2/3.0, selected
                State: <Active Ext>
                Local AS: 64511 Peer AS: 64510
                Age: 1:57:27
                Task: BGP_64510.10.0.0.14+54618
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:2 target:172.16.7.7:2
                Accepted
                Localpref: 100
                Router ID: 192.168.0.1

172.16.3.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 611
                Address: 0x9260130
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via fe-1/2/3.0, selected
                State: <Active Ext>
                Local AS: 64511 Peer AS: 64510
                Age: 1:57:27
                Task: BGP_64510.10.0.0.14+54618
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:3 origin:64511:3
                Accepted
                Localpref: 100
                Router ID: 192.168.0.1

172.16.4.0/24 (1 entry, 1 announced)
```

```
*BGP    Preference: 170/-101
        Next hop type: Router, Next hop index: 611
        Address: 0x9260130
        Next-hop reference count: 8
        Source: 10.0.0.14
        Next hop: 10.0.0.14 via fe-1/2/3.0, selected
        State: <Active Ext>
        Local AS: 64511 Peer AS: 64510
        Age: 1:57:27
        Task: BGP_64510.10.0.0.14+54618
        Announcement bits (1): 0-KRT
        AS path: 64510 I
        Communities: 64510:4 origin:172.16.7.7:4
        Accepted
        Localpref: 100
        Router ID: 192.168.0.1
```

Meaning The output shows that the regular community values remain attached to the routes, and the extended community values are added.

Related Documentation

- Example: Configuring Communities in a Routing Policy on page 372
- Example: Configuring a Routing Policy That Removes BGP Communities on page 413
- Example: Configuring a Routing Policy Based on the Number of BGP Communities on page 405
- *Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS*

Example: Configuring BGP Large Communities

This example shows you to configure optional transitive path attribute - a 12-byte BGP large community that provides the most significant 4-byte value to encode autonomous system number as the global administrator and the remaining two 4-byte assigned numbers to encode the local values as defined in RFC 8092. You can configure BGP large community at **[edit policy-options community *community-name* members]** and **[edit routing-options static route *ip-address* community]** hierarchy levels. The BGP large community attributes format has four fields: **large**:***global administrator***:***assigned number***:***assigned number***.

Requirements

This example uses the following hardware and software components:

- Three MX Series routers

- Junos OS Release 17.3 or later running on all devices

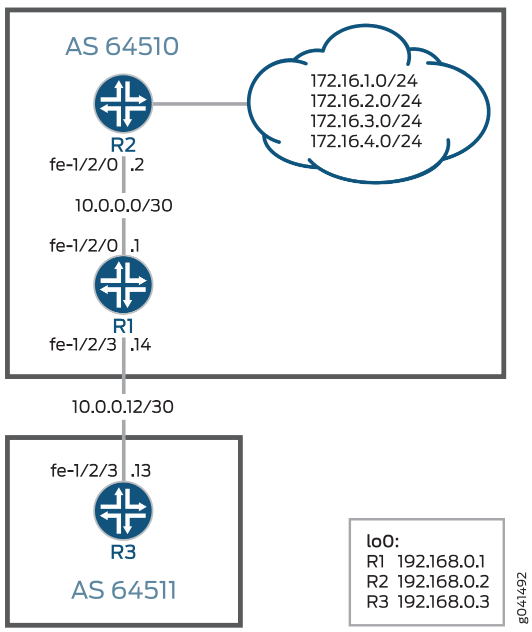
No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, Device R1 and Device R2 are OSPF neighbors in autonomous system (AS) 64510. Device R3 has an external BGP (EBGP) connection to Device R1. Device R2 has customer networks in the 172.16/16 address space, simulated with addresses on its loopback interface (lo0). Device R1 has static routes to several 172.16.x/24 networks, and attaches regular community values to these routes. Device R1 then uses an export policy to advertise the routes to Device R3. Device R3 receives these routes and uses an import policy to add large community values to the routes.

Topology

Figure 1 shows the sample network.



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

Device R1

```
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.1/30
set interfaces ge-0/0/1 unit 0 family inet address 10.0.0.14/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32 primary
set routing-options static route 172.16.1.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.1.0/24 community 64510:1
set routing-options static route 172.16.1.0/24 community large:64510:100:1
```

```
set routing-options static route 172.16.2.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.2.0/24 community 64510:2
set routing-options static route 172.16.2.0/24 community large:64510:200:2
set routing-options static route 172.16.3.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.3.0/24 community 64510:3
set routing-options static route 172.16.4.0/24 next-hop 10.0.0.2
set routing-options static route 172.16.4.0/24 community 64510:4
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
set protocols bgp group ext type external
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.13
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
```

Device R2

```
set interfaces ge-0/0/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set interfaces lo0 unit 0 family inet address 172.16.1.1/32
set interfaces lo0 unit 0 family inet address 172.16.2.2/32
set interfaces lo0 unit 0 family inet address 172.16.3.3/32
set interfaces lo0 unit 0 family inet address 172.16.4.4/32
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64510
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
```

Device R3

```
set interfaces ge-0/0/1 unit 0 family inet address 10.0.0.13/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 64511
set protocols bgp group to-R1 type external
set protocols bgp group to-R1 import set-large-comms
set protocols bgp group to-R1 peer-as 64510
set protocols bgp group to-R1 neighbor 10.0.0.14
set policy-options policy-statement set-large-comms term route-1 from route-filter
  172.16.1.0/24 exact
set policy-options policy-statement set-large-comms term route-1 then community add
  large2-as
set policy-options policy-statement set-large-comms term route-1 then accept
set policy-options policy-statement set-large-comms term route-2 from route-filter
  172.16.2.0/24 exact
set policy-options policy-statement set-large-comms term route-2 then community add
  large2-ip
set policy-options policy-statement set-large-comms term route-2 then accept
set policy-options policy-statement set-large-comms term route-3 from route-filter
  172.16.3.0/24 exact
set policy-options policy-statement set-large-comms term route-3 then community add
  large1-as
set policy-options policy-statement set-large-comms term route-3 then accept
set policy-options policy-statement set-large-comms term route-4 from route-filter
  172.16.4.0/24 exact
```



```
set policy-options policy-statement set-large-comms term route-4 then community add
  large1-ip
set policy-options policy-statement set-large-comms term route-4 then accept
set policy-options community large1-as members large:64511:3:1
set policy-options community large1-ip members large:7777:4:1
set policy-options community large2-as members large:64511:1:1
set policy-options community large2-ip members large:7777:2:1
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R3:

1. Configure the interfaces.

```
[edit interfaces]
set ge-0/0/1 unit 0 family inet address 10.0.0.13/30
set lo0 unit 0 family inet address 192.168.0.3/32
```

2. Configure the autonomous system (AS) number and router ID.

```
[edit routing-options]
set router-id 192.168.0.3
set autonomous-system 64511
```

3. Configure the EBGP connection to Device R1.

```
[edit protocols bgp group to-R1]
set type external
set import set-large-comms
set peer-as 64510
set neighbor 10.0.0.14
```

4. Configure the policy that adds large community values to the routes received from Device R1.

   A large community uses a notation of **large**:***global administrator***:***assigned number***:***assigned number***. The specific community values can be anything that accomplishes your administrative goals, within certain parameters.

```
[edit policy-options policy-statement set-large-comms]
set term route-1 from route-filter 172.16.1.0/24 exact
set term route-1 then community add large2-as
set term route-1 then accept
set term route-2 from route-filter 172.16.2.0/24 exact
set term route-2 then community add large2-ip
set term route-2 then accept
set term route-3 from route-filter 172.16.3.0/24 exact
set term route-3 then community add large1-as
set term route-3 then accept
set term route-4 from route-filter 172.16.4.0/24 exact
set term route-4 then community add large1-ip
set term route-4 then accept
```

```
[edit policy-options ]
set community large1-as members large:64511:3:1
set community large1-ip members large:7777:4:1
set community large2-as members large:64511:1:1
set community large2-ip members large:7777:2:1
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R3# show interfaces
ge-0/0/1 {
  unit 0 {
    family inet {
      address 10.0.0.13/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.3/32;
    }
  }
}
```

```
user@R3# show protocols
bgp {
  group to-R1 {
    type external;
    import set-large-comms;
    peer-as 64510;
    neighbor 10.0.0.14;
  }
}
```

```
user@R3# show policy-options
policy-statement set-large-comms {
  term route-1 {
    from {
      route-filter 172.16.1.0/24 exact;
    }
    then {
      community add large2-as;
      accept;
    }
  }
  term route-2 {
    from {
      route-filter 172.16.2.0/24 exact;
    }
    then {
```



```
                community add large2-ip;
                accept;
            }
        }
        term route-3 {
            from {
                route-filter 172.16.3.0/24 exact;
            }
            then {
                community add large1-as;
                accept;
            }
        }
        term route-4 {
            from {
                route-filter 172.16.4.0/24 exact;
            }
            then {
                community add large1-ip;
                accept;
            }
        }
        }
        community large1-as members large:64511:3:1;
        community large1-ip members large:7777:4:1;
        community large2-as members large:64511:1:1;
    community large2-ip members large:7777:2:1;
```

```
user@R3# show routing-options
router-id 192.168.0.3;
autonomous-system 64511;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Verifying R1 on page 401
- Verifying R3 on page 403

Verifying R1

Purpose On Device R1, check the 172.16. routes in the routing table.

Action

```
user@R1> show route protocol static match-prefix 172.16.* detail
inet.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
172.16.0.0/12 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 341
                Address: 0xb7a0d90
                Next-hop reference count: 9
                Next hop: 10.49.127.254 via fxp0.0, selected
                Session Id: 0x0
                State: < Active NoReadvrt Int Ext >
                Local AS: 64510
                Age: 4d 22:38:07
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

172.16.1.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 580
                Address: 0xb7a1270
                Next-hop reference count: 9
                Next hop: 10.0.0.2 via ge-0/0/0.0, selected
                Session Id: 0x140
                State: < Active Int Ext >
                Local AS: 64510
                Age: 4d 19:02:23
                Validation State: unverified
                Task: RT
                Announcement bits (2): 0-KRT 4-BGP_RT_Background
                AS path: I
                Communities: 64510:1 large:64510:100:1

172.16.2.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 580
                Address: 0xb7a1270
                Next-hop reference count: 9
                Next hop: 10.0.0.2 via ge-0/0/0.0, selected
                Session Id: 0x140
                State: < Active Int Ext >
                Local AS: 64510
                Age: 4d 19:02:23
                Validation State: unverified
                Task: RT
                Announcement bits (2): 0-KRT 4-BGP_RT_Background
                AS path: I
                Communities: 64510:2 large:64510:200:2

172.16.3.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 580
                Address: 0xb7a1270
                Next-hop reference count: 9
                Next hop: 10.0.0.2 via ge-0/0/0.0, selected
                Session Id: 0x140
                State: < Active Int Ext >
                Local AS: 64510
                Age: 4d 22:17:12
                Validation State: unverified
```



```
                Task: RT
                Announcement bits (2): 0-KRT 4-BGP_RT_Background
                AS path: I
                Communities: 64510:3

172.16.4.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 580
                Address: 0xb7a1270
                Next-hop reference count: 9
                Next hop: 10.0.0.2 via ge-0/0/0.0, selected
                Session Id: 0x140
                State: < Active Int Ext >
                Local AS: 64510
                Age: 4d 22:17:12
                Validation State: unverified
                Task: RT
                Announcement bits (2): 0-KRT 4-BGP_RT_Background
                AS path: I
                Communities: 64510:4

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

inet6.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
```

Meaning The output shows that the regular community and large community values are attached to the routes.

NOTE: The communities are attached to static routes, thus demonstrating that communities can be attached to non-BGP routes.

Verifying R3

Purpose On Device R3, check the 172.16. routes in the routing table.

Action

```
user@R3> show route protocol bgp match-prefix 172.16.* detail

inet.0: 14 destinations, 14 routes (14 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 581
                Address: 0xb7a10f0
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via ge-0/0/1.0, selected
                Session Id: 0x140
                State: < Active Ext >
                Local AS: 64511 Peer AS: 64510
                Age: 3d 16:36:18
                Validation State: unverified
                Task: BGP_64510.10.0.0.14
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:1 large:64510:100:1 large:64511:1:1
                Accepted
                Localpref: 100
                Router ID: 192.168.0.1

172.16.2.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 581
                Address: 0xb7a10f0
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via ge-0/0/1.0, selected
                Session Id: 0x140
                State: < Active Ext >
                Local AS: 64511 Peer AS: 64510
                Age: 3d 16:36:18
                Validation State: unverified
                Task: BGP_64510.10.0.0.14
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:2 large:7777:2:1 large:64510:200:2
                Accepted
                Localpref: 100
                Router ID: 192.168.0.1

172.16.3.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 581
                Address: 0xb7a10f0
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via ge-0/0/1.0, selected
                Session Id: 0x140
                State: < Active Ext >
                Local AS: 64511 Peer AS: 64510
                Age: 3d 16:36:18
                Validation State: unverified
                Task: BGP_64510.10.0.0.14
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:3 large:64511:3:1
                Accepted
```



```
                Localpref: 100
                Router ID: 192.168.0.1

172.16.4.0/24 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 581
                Address: 0xb7a10f0
                Next-hop reference count: 8
                Source: 10.0.0.14
                Next hop: 10.0.0.14 via ge-0/0/1.0, selected
                Session Id: 0x140
                State: < Active Ext >
                Local AS: 64511 Peer AS: 64510
                Age: 3d 16:36:18
                Validation State: unverified
                Task: BGP_64510.10.0.0.14
                Announcement bits (1): 0-KRT
                AS path: 64510 I
                Communities: 64510:4 large:7777:4:1
                Accepted
                Localpref: 100
                Router ID: 192.168.0.1

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

inet6.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
```

Meaning The output shows that the regular community values remain attached to the routes, and the large community values are added.

Related Documentation

- Understanding How to Define BGP Communities and Extended Communities on page 361
- **community on page 1175**

Example: Configuring a Routing Policy Based on the Number of BGP Communities

This example shows how to create a policy that accepts BGP routes based on the number of BGP communities.

- Requirements on page 405
- Overview on page 406
- Configuration on page 406
- Verification on page 411

Requirements

No special configuration beyond device initialization is required before you configure this example.

Overview

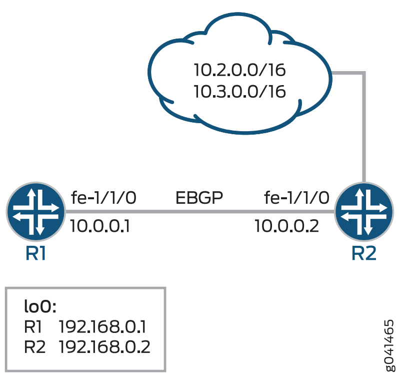
This example shows two routing devices with an external BGP (EBGP) connection between them. Device R2 uses the BGP session to send two static routes to Device R1. On Device R1, an import policy specifies that the BGP-received routes can contain up to five communities to be considered a match. For example, if a route contains three communities, it is considered a match and is accepted. If a route contains six or more communities, it is considered a nonmatch and is rejected.

It is important to remember that the default policy for EBGP is to accept all routes. To ensure that the nonmatching routes are rejected, you must include a **then reject** action at the end of the policy definition.

Topology

Figure 34 on page 406 shows the sample network.

Figure 34: BGP Policy with a Limit on the Number of Communities Accepted



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/1/0 unit 0 description to-R2
set interfaces fe-1/1/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers peer-as 2
set protocols bgp group external-peers neighbor 10.0.0.2 import import-communities
set policy-options policy-statement import-communities term 1 from protocol bgp
set policy-options policy-statement import-communities term 1 from community-count
  5 orlower
set policy-options policy-statement import-communities term 1 then accept
set policy-options policy-statement import-communities term 2 then reject
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 1
```

Device R2

```
set interfaces fe-1/1/0 unit 0 description to-R1
set interfaces fe-1/1/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers export statics
set protocols bgp group external-peers peer-as 1
set protocols bgp group external-peers neighbor 10.0.0.1
set policy-options policy-statement statics from protocol static
set policy-options policy-statement statics then community add 1
set policy-options policy-statement statics then accept
set policy-options community 1 members 2:1
set policy-options community 1 members 2:2
set policy-options community 1 members 2:3
set policy-options community 1 members 2:4
set policy-options community 1 members 2:5
set policy-options community 1 members 2:6
set policy-options community 1 members 2:7
set policy-options community 1 members 2:8
set policy-options community 1 members 2:9
set policy-options community 1 members 2:10
set routing-options static route 10.2.0.0/16 reject
set routing-options static route 10.2.0.0/16 install
set routing-options static route 10.3.0.0/16 reject
set routing-options static route 10.3.0.0/16 install
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 2
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set fe-1/1/0 unit 0 description to-R2
user@R1# set fe-1/1/0 unit 0 family inet address 10.0.0.1/30

user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure BGP.

   Apply the import policy to the BGP peering session with Device R2.

```
[edit protocols bgp group external-peers]
user@R1# set type external
user@R1# set peer-as 2
user@R1# set neighbor 10.0.0.2 import import-communities
```

3. Configure the routing policy that sends direct routes.

```
[edit policy-options policy-statement import-communities]
user@R1# set term 1 from protocol bgp
user@R1# set term 1 from community-count 5 orlower
```

```
user@R1# set term 1 then accept
user@R1# set term 2 then reject
```

4. Configure the autonomous system (AS) number and the router ID.

```
[edit routing-options ]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 1
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set fe-1/1/0 unit 0 description to-R1
user@R2# set fe-1/1/0 unit 0 family inet address 10.0.0.2/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the router ID and the autonomous system (AS) number.

```
[edit routing-options]
user@R2# set router-id 192.168.0.3
user@R2# set autonomous-system 2
```

3. Configure BGP.

```
[edit protocols bgp group external-peers]
user@R2# set type external
user@R2# set peer-as 1
user@R2# set neighbor 10.0.0.1
```

4. Configure multiple communities, or configure a single community with multiple members.

```
[edit policy-options community 1]
user@R2# set members 2:1
user@R2# set members 2:2
user@R2# set members 2:3
user@R2# set members 2:4
user@R2# set members 2:5
user@R2# set members 2:6
user@R2# set members 2:7
user@R2# set members 2:8
user@R2# set members 2:9
user@R2# set members 2:10
```

5. Configure the static routes.

```
[edit routing-options static]
user@R2# set route 10.2.0.0/16 reject
user@R2# set route 10.2.0.0/16 install

user@R2# set route 10.3.0.0/16 reject
user@R2# set route 10.3.0.0/16 install
```

6. Configure a routing policy that advertises static routes into BGP and adds the BGP community to the routes.

```
[edit policy-options policy-statement statics]
user@R2# set from protocol static
user@R2# set then community add 1
user@R2# set then accept
```

7. Apply the export policy.

```
[edit protocols bgp group external-peers]
user@R2# set export statics
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device R1

```
user@R1# show interfaces
fe-1/1/0 {
  unit 0{
    description to-R2;
    family inet {
      address 10.0.0.1/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.1/32;
    }
  }
}
}

user@R1# show protocols
bgp {
  group external-peers {
    type external;
    peer-as 2;
    neighbor 10.0.0.2 {
      import import-communities;
    }
  }
```

```
}
```

```
user@R1# show policy-options
policy-statement import-communities {
   term 1 {
      from {
         protocol bgp;
         community-count 5 orlower;
      }
      then accept;
   }
   term 2 {
      then reject;
   }
}
```

```
user@R1# show routing-options
router-id 192.168.0.1;
autonomous-system 1;
```

Device R2

```
user@R2# show interfaces
fe-1/1/0 {
   unit 0 {
      description to-R1;
      family inet {
         address 10.0.0.2/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.2/32;
      }
   }
}
```

```
user@R2# show protocols
bgp {
   group external-peers {
      type external;
      export statics;
      peer-as 1;
      neighbor 10.0.0.1;
   }
}
```

```
user@R2# show policy-options
policy-statement statics {
   from protocol static;
   then {
      community add 1;
      accept;
   }
}
community 1 members [ 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10 ];
```



```
user@R2# show routing-options
static {
  route 10.2.0.0/16 {
    reject;
    install;
  }
  route 10.3.0.0/16 {
    reject;
    install;
  }
}
router-id 192.168.0.3;
autonomous-system 2;
```

If you are done configuring the devices, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the BGP Routes

Purpose Make sure that the routing table on Device R1 contains the expected BGP routes.

Action

1. On Device R1, run the **show route protocols bgp** command.

   ```
   user@R1> show route protocols bgp

   inet.0: 5 destinations, 5 routes (3 active, 0 holddown, 2 hidden)
   ```

2. On Device R1, change the **community-count** configuration in the import policy.

   ```
   [edit policy-options policy-statement import-communities term 1]
   user@R1# set from community-count 5 orhigher
   user@R1# commit
   ```

3. On Device R1, run the **show route protocols bgp** command.

   ```
   user@R1> show route protocols bgp

   inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
   + = Active Route, - = Last Active, * = Both

   10.2.0.0/16        *[BGP/170] 18:29:53, localpref 100
                         AS path: 2 I, validation-state: unverified
                       > to 10.0.0.2 via fe-1/1/0.0
   10.3.0.0/16        *[BGP/170] 18:29:53, localpref 100
                         AS path: 2 I, validation-state: unverified
                       > to 10.0.0.2 via fe-1/1/0.0
   ```

4. On Device R1, run the **show route protocols bgp extensive** command to view the advertised communities.

   ```
   user@R1> show route protocols bgp extensive
   inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
   10.2.0.0/16 (1 entry, 1 announced)
   ```

```
TSI:
KRT in-kernel 10.2.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via fe-1/1/0.0, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 18:56:10
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Communities: 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3

10.3.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.3.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via fe-1/1/0.0, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 18:56:10
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Communities: 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3
```

Meaning The output shows that in Device R1's routing table, the BGP routes sent from Device R2 are hidden. When the **community-count** setting in Device R1's import policy is modified, the BGP routes are no longer hidden.

Related Documentation

- *Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS*
- *Understanding External BGP Peering Sessions*

Example: Configuring a Routing Policy That Removes BGP Communities

This example shows how to create a policy that accepts BGP routes, but removes BGP communities from the routes.

Requirements

No special configuration beyond device initialization is required before you configure this example.

Overview

This example shows two routing devices with an external BGP (EBGP) connection between them. Device R2 uses the BGP session to send two static routes to Device R1. On Device R1, an import policy specifies that all BGP communities must be removed from the routes.

By default, when communities are configured on EBGP peers, they are sent and accepted. To suppress the acceptance of communities received from a neighbor, you can remove all communities or a specified set of communities. When the result of a policy is an empty set of communities, the community attribute is not included. To remove all communities, first define a wildcard set of communities (here, the community is named **wild**):

```
[edit policy-options]
community wild members "* : *";
```

Then, in the routing policy statement, specify the **community delete** action:

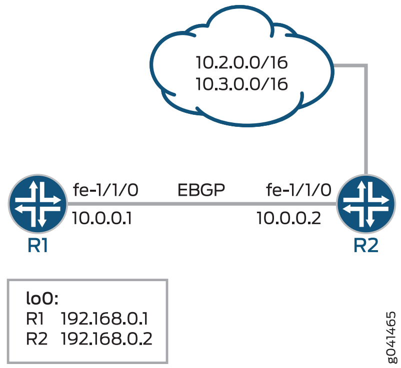
```
[edit policy-options]
policy-statement policy-name {
  term term-name {
    then community delete wild;
  }
}
```

To suppress a particular community from any autonomous system (AS), define the community as **community wild members "*:*****community-value*****"**.

Topology

Figure 35 on page 414 shows the sample network.

Figure 35: BGP Policy That Removes Communities



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/1/0 unit 0 description to-R2
set interfaces fe-1/1/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers peer-as 2
set protocols bgp group external-peers neighbor 10.0.0.2 import remove-communities
set policy-options policy-statement remove-communities term 1 from protocol bgp
set policy-options policy-statement remove-communities term 1 then community delete
  wild
set policy-options policy-statement remove-communities term 1 then accept
set policy-options policy-statement remove-communities term 2 then reject
set policy-options community wild members *:*
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 1
```

Device R2

```
set interfaces fe-1/1/0 unit 0 description to-R1
set interfaces fe-1/1/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers export statics
set protocols bgp group external-peers peer-as 1
set protocols bgp group external-peers neighbor 10.0.0.1
set policy-options policy-statement statics from protocol static
set policy-options policy-statement statics then community add 1
set policy-options policy-statement statics then accept
set policy-options community 1 members 2:1
set policy-options community 1 members 2:2
set policy-options community 1 members 2:3
set policy-options community 1 members 2:4
set policy-options community 1 members 2:5
set policy-options community 1 members 2:6
set policy-options community 1 members 2:7
```

```
set policy-options community 1 members 2:8
set policy-options community 1 members 2:9
set policy-options community 1 members 2:10
set routing-options static route 10.2.0.0/16 reject
set routing-options static route 10.2.0.0/16 install
set routing-options static route 10.3.0.0/16 reject
set routing-options static route 10.3.0.0/16 install
set routing-options router-id 192.168.0.3
set routing-options autonomous-system 2
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the interfaces.

```
[edit interfaces]
user@R1# set fe-1/1/0 unit 0 description to-R2
user@R1# set fe-1/1/0 unit 0 family inet address 10.0.0.1/30

user@R1# set lo0 unit 0 family inet address 192.168.0.1/32
```

2. Configure BGP.

   Apply the import policy to the BGP peering session with Device R2.

```
[edit protocols bgp group external-peers]
user@R1# set type external
user@R1# set peer-as 2
user@R1# set neighbor 10.0.0.2 import remove-communities
```

3. Configure the routing policy that deletes communities.

```
[edit policy-options policy-statement remove-communities]
user@R1# set term 1 from protocol bgp
user@R1# set term 1 then community delete wild
user@R1# set term 1 then accept
user@R1# set term 2 then reject
```

4. Configure the autonomous system (AS) number and the router ID.

```
[edit routing-options ]
user@R1# set router-id 192.168.0.1
user@R1# set autonomous-system 1
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set fe-1/1/0 unit 0 description to-R1
user@R2# set fe-1/1/0 unit 0 family inet address 10.0.0.2/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the router ID and the autonomous system (AS) number.

```
[edit routing-options]
user@R2# set router-id 192.168.0.3
user@R2# set autonomous-system 2
```

3. Configure BGP.

```
[edit protocols bgp group external-peers]
user@R2# set type external
user@R2# set peer-as 1
user@R2# set neighbor 10.0.0.1
```

4. Configure multiple communities, or configure a single community with multiple members.

```
[edit policy-options community 1]
user@R2# set members 2:1
user@R2# set members 2:2
user@R2# set members 2:3
user@R2# set members 2:4
user@R2# set members 2:5
user@R2# set members 2:6
user@R2# set members 2:7
user@R2# set members 2:8
user@R2# set members 2:9
user@R2# set members 2:10
```

5. Configure the static routes.

```
[edit routing-options static]
user@R2# set route 10.2.0.0/16 reject
user@R2# set route 10.2.0.0/16 install
user@R2# set route 10.3.0.0/16 reject
user@R2# set route 10.3.0.0/16 install
```

6. Configure a routing policy that advertises static routes into BGP and adds the BGP community to the routes.

```
[edit policy-options policy-statement statics]
```



```
user@R2# set from protocol static
user@R2# set then community add 1
user@R2# set then accept
```

7. Apply the export policy.

```
[edit protocols bgp group external-peers]
user@R2# set export statics
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device R1

```
user@R1# show interfaces
fe-1/1/0 {
   unit 0{
      description to-R2;
      family inet {
         address 10.0.0.1/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.1/32;
      }
   }
}

user@R1# show protocols
bgp {
   group external-peers {
      type external;
      peer-as 2;
      neighbor 10.0.0.2 {
         import remove-communities;
      }
   }
}

user@R1# show policy-options
policy-statement remove-communities {
   term 1 {
      from protocol bgp;
      then {
         community delete wild;
         accept;
      }
   }
   term 2 {
      then reject;
   }
```

```
}
community wild members *:*;
```

```
user@R1# show routing-options
router-id 192.168.0.1;
autonomous-system 1;
```

Device R2

```
user@R2# show interfaces
fe-1/1/0 {
   unit 0 {
      description to-R1;
      family inet {
         address 10.0.0.2/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.2/32;
      }
   }
}
```

```
user@R2# show protocols
bgp {
   group external-peers {
      type external;
      export statics;
      peer-as 1;
      neighbor 10.0.0.1;
   }
}
```

```
user@R2# show policy-options
policy-statement statics {
   from protocol static;
   then {
      community add 1;
      accept;
   }
}
community 1 members [ 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10 ];
```

```
user@R2# show routing-options
static {
   route 10.2.0.0/16 {
      reject;
      install;
   }
   route 10.3.0.0/16 {
      reject;
      install;
   }
}
router-id 192.168.0.3;
autonomous-system 2;
```


If you are done configuring the devices, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying the BGP Routes

Purpose Make sure that the routing table on Device R1 does not contain BGP communities.

Action 1. On Device R1, run the **show route protocols bgp extensive** command.

```
user@R1> show route protocols bgp extensive

inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
10.2.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.2.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via lt-1/1/0.5, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 20:39:01
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3

10.3.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.3.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via lt-1/1/0.5, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 20:39:01
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3
```


2. On Device R1, deactivate the **community remove** configuration in the import policy.

   [edit policy-options policy-statement remove-communities term 1]
   user@R1# **deactivate then community delete wild**
   user@R1# **commit**

3. On Device R1, run the **show route protocols bgp extensive** command to view the advertised communities.

```
user@R1> show route protocols bgp extensive
inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
10.2.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.2.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via lt-1/1/0.5, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 20:40:53
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Communities: 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3

10.3.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.3.0.0/16 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 671
                Address: 0x9458270
                Next-hop reference count: 4
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via lt-1/1/0.5, selected
                Session Id: 0x100001
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 20:40:53
                Validation State: unverified
                Task: BGP_2.10.0.0.2+179
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Communities: 2:1 2:2 2:3 2:4 2:5 2:6 2:7 2:8 2:9 2:10
                Accepted
                Localpref: 100
                Router ID: 192.168.0.3
```


Meaning The output shows that in Device R1's routing table, the communities are suppressed in the BGP routes sent from Device R2. When the **community remove** setting in Device R1's import policy is deactivated, the communities are no longer suppressed.

Related Documentation

- *Example: Configuring a Routing Policy to Redistribute BGP Routes with a Specific Community Tag into IS-IS*
- *Understanding External BGP Peering Sessions*

CHAPTER 8

Increasing Network Stability with BGP Route Flapping Actions

Understanding Damping Parameters

BGP *route flapping* describes the situation in which BGP systems send an excessive number of update messages to advertise network reachability information. BGP *flap damping* is a method of reducing the number of update messages sent between BGP peers, thereby reducing the load on these peers, without adversely affecting the route convergence time for stable routes.

Flap damping reduces the number of update messages by marking routes as ineligible for selection as the active or preferable route. Marking routes in this way leads to some delay, or *suppression*, in the propagation of route information, but the result is increased network stability. You typically apply flap damping to external BGP (EBGP) routes (routes in different ASs). You can also apply flap damping within a confederation, between confederation member ASs. Because routing consistency within an AS is important, do not apply flap damping to internal BGP (IBGP) routes. (If you do, it is ignored.)

There is an exception that rule. Starting in Junos OS Release 12.2, you can apply flap damping at the address family level. In a Junos OS Release 12.2 or later installation, when you apply flap damping at the address family level, it works for both IBGP and EBGP.

By default, route flap damping is not enabled. Damping is applied to external peers and to peers at confederation boundaries.

When you enable damping, default parameters are applied, as summarized in Table 25 on page 424.

Table 25: Damping Parameters

Damping Parameter	Description	Default Value	Possible Values
half-life ***minutes***	Decay half-life—Number of minutes after which an arbitrary value is halved if a route stays stable.	**15** (minutes)	**1** through **45**
max-suppress ***minutes***	Maximum hold-down time for a route, in minutes.	**60** (minutes)	**1** through **720**
reuse	Reuse threshold—Arbitrary value below which a suppressed route can be used again.	**750**	**1** through **20,000**
suppress	Cutoff (suppression) threshold—Arbitrary value above which a route can no longer be used or included in advertisements.	**3000**	**1** through **20,000**

To change the default BGP flap damping values, you define actions by creating a named set of damping parameters and including it in a routing policy with the damping action. For the damping routing policy to work, you also must enable BGP route flap damping.

Release History Table

Release	Description
12.2	Starting in Junos OS Release 12.2, you can apply flap damping at the address family level.

Related Documentation

- Understanding Routing Policies on page 17
- Example: Configuring BGP Route Flap Damping Parameters on page 430

Using Routing Policies to Damp BGP Route Flapping

BGP *route flapping* describes the situation in which BGP systems send an excessive number of update messages to advertise network reachability information. BGP *flap damping* is a way to reduce the number of update messages sent between BGP peers, thereby reducing the load on these peers without adversely affecting the route convergence time.

Flap damping reduces the number of update messages by marking routes as ineligible for selection as the active or preferable route. Doing this leads to some delay, or *suppression*, in the propagation of route information, but the result is increased network stability. You typically apply flap damping to external BGP (EBGP) routes (that is, to routes in different ASs). You can also apply it within a confederation, between confederation member ASs. Because routing consistency within an AS is important, do not apply flap damping to IBGP routes. (If you do, it is ignored.)

BGP flap damping is defined in RFC 2439, *BGP Route Flap Damping.*

To effect changes to the default BGP flap damping values, you define actions by creating a named set of damping parameters and including it in a routing policy with the **damping** action (described in "Configuring Actions That Manipulate Route Characteristics" on page 57). For the damping routing policy to work, you also must enable BGP route flap damping.

The following sections discuss the following topics:

Configuring BGP Flap Damping Parameters

To define damping parameters, include the **damping** statement:

```
[edit policy-options]
damping name {
   disable;
   half-life minutes;
   max-suppress minutes;
   reuse number;
   suppress number;
}
```

The name identifies the group of damping parameters. It can contain letters, numbers, and hyphens (-) and can be up to 255 characters. To include spaces in the name, enclose the entire name in quotation marks (" ").

You can specify one or more of the damping parameters described in Table 26 on page 425.

Table 26: Damping Parameters

Damping Parameter	Description	Default	Possible Values
half-life ***minutes***	Decay half-life, in minutes	15 minutes	1 through 45 minutes
max-suppress ***minutes***	Maximum hold-down time, in minutes	60 minutes	1 through 720 minutes
reuse	Reuse threshold	750 (unitless)	1 through 20,000 (unitless)
suppress	Cutoff (suppression) threshold	3000 (unitless)	1 through 20,000 (unitless)

If you do not specify one or more of the damping parameters, the default value of the parameter is used.

To understand how to configure these parameters, you need to understand how damping suppresses routes. How long a route can be suppressed is based on a *figure of merit*, which is a value that correlates to the probability of future instability of a route. Routes with higher figure-of-merit values are suppressed for longer periods of time. The figure-of-merit value decays exponentially over time.

A figure-of-merit value of zero is assigned to each new route. The value is increased each time the route is withdrawn or readvertised, or when one of its path attributes changes. With each incident of instability, the value increases as follows:

- Route is withdrawn—1000
- Route is readvertised—1000
- Route's path attributes change—500

NOTE: Other vendors' implementations for figure-of-merit increase the value only when a route is withdrawn. The Junos OS implementation for figure-of-merit increases the value for both route withdrawal and route readvertisement. To accommodate other implementations for figure-of-merit, multiply the **reuse** and **suppress** threshold values by 2.

When a route's figure-of-merit value reaches a particular level, called the *cutoff* or *suppression threshold*, the route is suppressed. If a route is suppressed, the routing table no longer installs the route into the forwarding table and no longer exports this route to any of the routing protocols. By default, a route is suppressed when its figure-of-merit value reaches 3000. To modify this default, include the **suppress** option at the **[edit policy-options damping *name*]** hierarchy level.

If a route has flapped, but then becomes stable so that none of the incidents listed previously occur within a configurable amount of time, the figure-of-merit value for the route decays exponentially. The default half-life is 15 minutes. For example, for a route with a figure-of-merit value of 1500, if no incidents occur, its figure-of-merit value is reduced to 750 after 15 minutes and to 375 after another 15 minutes. To modify the default half-life, include the **half-life** option at the **[edit policy-options damping *name*]** hierarchy level.

NOTE: For the half-life, configure a value that is less than the max-suppress. If you do not, the configuration is rejected.

A suppressed route becomes reusable when its figure-of-merit value decays to a value below a *reuse threshold*, thus allowing routes that experience transient instability to once again be considered valid. The default reuse threshold is 750. When the figure-of-merit value passes below the reuse threshold, the route once again is considered usable and can be installed in the forwarding table and exported from the routing table. To modify the default reuse threshold, include the **reuse** option at the **[edit policy-options damping *name*]** hierarchy level.

The maximum suppression time provides an upper bound on the time that a route can remain suppressed. The default maximum suppression time is 60 minutes. To modify the default, include the **max-suppress** option at the **[edit policy-options damping *name*]** hierarchy level.

NOTE: For the max-suppress, configure a value that is greater than the half-life. If you do not, the configuration is rejected.

A route's figure-of-merit value stops increasing when it reaches a maximum suppression threshold, which is determined based on the route's suppression threshold level, half-life, reuse threshold, and maximum hold-down time.

The merit ceiling, ε_c, which is the maximum merit that a flapping route can collect, is calculated using the following formula:

$$\varepsilon_c \leq \varepsilon_r e^{(t/\lambda)\,(\ln 2)}$$

ε_r is the figure-of-merit reuse threshold, t is the maximum hold-down time in minutes, and λ is the half-life in minutes. For example, if you use the default figure-of-merit values in this formula, but use a half-life of 30 minutes, the calculation is as follows:

$$\varepsilon_c \leq 750\, e^{(60/30)\,(\ln 2)}$$

$$\varepsilon_c \leq 3000$$

NOTE: The cutoff threshold, which you configure using the **suppress** option, must be less than or equal to the merit ceiling, ε_c. If the configured cutoff threshold or the default cutoff threshold is greater than the merit ceiling, the route is never suppressed and damping never occurs.

To display figure-of-merit information, use the **show policy damping** command.

A route that has been assigned a figure of merit is considered to have a damping state. To display the current damping information on the routing device, use the **show route detail** command.

Specifying BGP Flap Damping as the Action in Routing Policy Terms

To BGP flap damping as the action in a routing policy term, include the **damping** statement and the name of the configured damping parameters either as an option of the **route-filter** statement at the **[edit policy-options policy-statement *policy-name* term *term-name* from]** hierarchy level:

```
[edit policy-options policy-statement policy-name term term-name from]
route-filter destination-prefix match-type {
   damping damping-parameters;
}
```

or at the **[edit policy-options policy-statement *policy-name* term *term-name* then]** hierarchy level:

```
[edit policy-options policy-statement policy-name term term-name then]
damping damping-parameters;
```


Disabling Damping for Specific Address Prefixes

Normally, you enable or disable damping on a per-peer basis. However, you can disable damping for a specific prefix received from a peer by including the **disable** option:

```
[edit policy-options damping name]
disable;
```

Disabling Damping for a Specific Address Prefix

In this routing policy example, although damping is enabled for the peer, the **damping none** statement specifies that damping be disabled for prefix 10.0.0.0/8 in **Policy-A**. This route is not damped because the routing policy statement named **Policy-A** filters on the prefix 10.0.0.0/8 and the action points to the **damping** statement named **none**. The remaining prefixes are damped using the default parameters.

```
[edit]
policy-options {
  policy-statement Policy-A {
    from {
      route-filter 10.0.0.0/8 exact;
    }
    then damping none;
  }
  damping none {
    disable;
  }
}
```

Configuring BGP Flap Damping

Enable BGP flap damping and configure damping parameters:

```
[edit]
routing-options {
  autonomous-system 666;
}
protocols {
  bgp {
    damping;
    group group1 {
      traceoptions {
        file bgp-log size 1m files 10;
        flag damping;
      }
      import damp;
      type external;
      peer-as 10458;
      neighbor 192.168.2.30;
    }
  }
}
policy-options {
  policy-statement damp {
    from {
```

```
            route-filter 192.168.0.0/32 exact {
                damping high;
                accept;
            }
            route-filter 172.16.0.0/32 exact {
                damping medium;
                accept;
            }
            route-filter 10.0.0.0/8 exact {
                damping none;
                accept;
            }
        }
    }
    damping high {
        half-life 30;
        suppress 3000;
        reuse 750;
        max-suppress 60;
    }
    damping medium {
        half-life 15;
        suppress 3000;
        reuse 750;
        max-suppress 45;
    }
    damping none {
        disable;
    }
}
```

To display damping parameters for this configuration, use the **show policy damping** command:

```
user@host> show policy damping
Damping information for "high":
  Halflife: 30 minutes
  Reuse merit: 750 Suppress/cutoff merit: 3000
  Maximum suppress time: 60 minutes
  Computed values:
    Merit ceiling: 3008
    Maximum decay: 24933
Damping information for "medium":
  Halflife: 15 minutes
  Reuse merit: 750 Suppress/cutoff merit: 3000
  Maximum suppress time: 45 minutes
  Computed values:
    Merit ceiling: 6024
    Maximum decay: 12449
Damping information for "none":
Damping disabled
```

Related Documentation

- Example: Configuring BGP Route Flap Damping Parameters on page 430
- Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family on page 439

Example: Configuring BGP Route Flap Damping Parameters

This example shows how to configure damping parameters.

Requirements

Before you begin, configure router interfaces and configure routing protocols.

Overview

This example has three routing devices. Device R2 has external BGP (EBGP) connections with Device R1 and Device R3.

Device R1 and Device R3 have some static routes configured for testing purposes, and these static routes are advertised through BGP to Device R2.

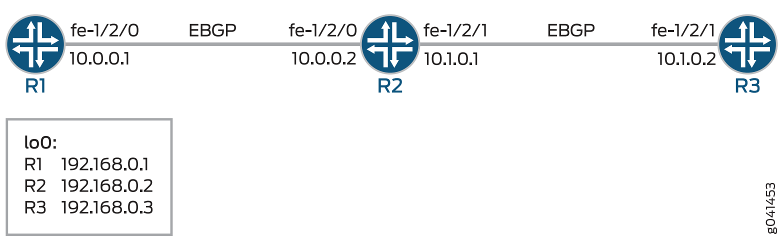
Device R2 damps routes received from Device R1 and Device R3 according to these criteria:

- Damp all prefixes with a mask length equal to or greater than 17 more aggressively than routes with a mask length between 9 and 16.
- Damp routes with a mask length between 0 and 8, inclusive, less than routes with a mask length greater than 8.
- Do not damp the 10.128.0.0/9 prefix at all.

The routing policy is evaluated when routes are being exported from the routing table into the forwarding table. Only the active routes are exported from the routing table.

Figure 36 on page 430 shows the sample network.

Figure 36: BGP Flap Damping Topology



“CLI Quick Configuration” on page 431 shows the configuration for all of the devices in Figure 36 on page 430.

The section “Step-by-Step Procedure” on page 432 describes the steps on Device R2.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct-and-static
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-direct-and-static term 1 from protocol direct
set policy-options policy-statement send-direct-and-static term 1 from protocol static
set policy-options policy-statement send-direct-and-static term 1 then accept
set routing-options static route 172.16.0.0/16 reject
set routing-options static route 172.16.128.0/17 reject
set routing-options static route 172.16.192.0/20 reject
set routing-options static route 10.0.0.0/9 reject
set routing-options static route 172.16.233.0/7 reject
set routing-options static route 10.224.0.0/11 reject
set routing-options static route 0.0.0.0/0 reject
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp damping
set protocols bgp group ext type external
set protocols bgp group ext import damp
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set protocols bgp group ext neighbor 10.1.0.2 peer-as 300
set policy-options policy-statement damp term 1 from route-filter 10.128.0.0/9 exact
  damping dry
set policy-options policy-statement damp term 1 from route-filter 0.0.0.0/0
  prefix-length-range /0-/8 damping timid
set policy-options policy-statement damp term 1 from route-filter 0.0.0.0/0
  prefix-length-range /17-/32 damping aggressive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options damping aggressive half-life 30
set policy-options damping aggressive suppress 2500
set policy-options damping timid half-life 5
set policy-options damping dry disable
set routing-options autonomous-system 200
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct-and-static
set protocols bgp group ext peer-as 200
```



```
set protocols bgp group ext neighbor 10.1.0.1
set policy-options policy-statement send-direct-and-static term 1 from protocol direct
set policy-options policy-statement send-direct-and-static term 1 from protocol static
set policy-options policy-statement send-direct-and-static term 1 then accept
set routing-options static route 10.128.0.0/9 reject
set routing-options autonomous-system 300
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure damping parameters:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

user@R2# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure the BGP neighbors.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set neighbor 10.0.0.1 peer-as 100
user@R2# set neighbor 10.1.0.2 peer-as 300
```

3. Create and configure the damping parameter groups.

```
[edit policy-options]
user@R2# set damping aggressive half-life 30
user@R2# set damping aggressive suppress 2500
user@R2# set damping timid half-life 5
user@R2# set damping dry disable
```

4. Configure the damping policy.

```
[edit policy-options policy-statement damp term 1]
user@R2# set from route-filter 10.128.0.0/9 exact damping dry
user@R2# set from route-filter 0.0.0.0/0 prefix-length-range /0-/8 damping timid
user@R2# set from route-filter 0.0.0.0/0 prefix-length-range /17-/32 damping
  aggressive
```

5. Enable damping for BGP.

```
[edit protocols bgp]
user@R2# set damping
```


6. Apply the policy as an import policy for the BGP neighbor.

```
[edit protocols bgp group ext]
user@R2# set import damp
```

NOTE: You can refer to the same routing policy one or more times in the same or different **import** statements.

7. Configure an export policy.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

8. Apply the export policy.

```
[edit protocols bgp group ext]
user@R2# set export send-direct
```

9. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R2# set autonomous-system 200
```

Results From configuration mode, confirm your configuration by issuing the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      address 10.0.0.2/30;
    }
  }
}
fe-1/2/1 {
  unit 0 {
    family inet {
      address 10.1.0.1/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 192.168.0.2/32;
    }
  }
```



```
}
user@R2# show protocols
bgp {
  damping;
  group ext {
    type external;
    import damp;
    export send-direct;
    neighbor 10.0.0.1 {
      peer-as 100;
    }
    neighbor 10.1.0.2 {
      peer-as 300;
    }
  }
}
user@R2# show policy-options
policy-statement damp {
  term 1 {
    from {
      route-filter 10.128.0.0/9 exact damping dry;
      route-filter 0.0.0.0/0 prefix-length-range /0-/8 damping timid;
      route-filter 0.0.0.0/0 prefix-length-range /17-/32 damping aggressive;
    }
  }
}
policy-statement send-direct {
  term 1 {
    from protocol direct;
    then accept;
  }
}
damping aggressive {
  half-life 30;
  suppress 2500;
}
damping timid {
  half-life 5;
}
damping dry {
  disable;
}
user@R2# show routing-options
autonomous-system 200;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Causing Some Routes to Flap on page 435
- Checking the Route Flaps on page 435

Causing Some Routes to Flap

Purpose To verify your route flap damping policy, some routes must flap. Having a live Internet feed almost guarantees that a certain number of route flaps will be present. If you have control over a remote system that is advertising the routes, you can modify the advertising router's policy to effect the advertisement and withdrawal of all routes or of a given prefix. In a test environment, you can cause routes to flap by clearing the BGP neighbors or by restarting the routing process on the BGP neighbors, as shown here.

Action From operational mode on Device R1 and Device R3, enter the **restart routing** command.

CAUTION: Use this command cautiously in a production network.

```
user@R1> restart routing

R1 started, pid 10474

user@R3> restart routing

R3 started, pid 10478
```

Meaning On Device R2, all of the routes from the neighbors are withdrawn and re-advertised.

Checking the Route Flaps

Purpose View the number of neighbor flaps.

Action From operational mode, enter the **show bgp summary** command.

```
user@R2> show bgp summary

Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
inet.0
                      12          1         11          0         11          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn 
State|#Active/Received/Accepted/Damped...
10.0.0.1                100         10         10       0       4        2:50 
0/9/0/9              0/0/0/0
10.1.0.2                300         10         10       0       4        2:53 
1/3/1/2              0/0/0/0
```

Meaning This output was captured after the routing process was restarted on Device R2's neighbors four times.

Verifying Route Flap Damping

Purpose Verify that routes are being hidden due to damping.

Action From operational mode, enter the **show route damping suppressed** command.

```
user@R2> show route damping suppressed

inet.0: 15 destinations, 17 routes (6 active, 0 holddown, 11 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          [BGP ] 00:00:12, localpref 100
                     AS path: 100 I, validation-state: unverified
                   > to 10.0.0.1 via fe-1/2/0.0
10.0.0.0/9         [BGP ] 00:00:12, localpref 100
                     AS path: 100 I, validation-state: unverified
                   > to 10.0.0.1 via fe-1/2/0.0
10.0.0.0/30        [BGP ] 00:00:12, localpref 100
                     AS path: 100 I, validation-state: unverified
                   > to 10.0.0.1 via fe-1/2/0.0
10.1.0.0/30        [BGP ] 00:00:15, localpref 100
                     AS path: 300 I, validation-state: unverified
                   > to 10.1.0.2 via fe-1/2/1.0
10.224.0.0/11      [BGP ] 00:00:12, localpref 100
                     AS path: 100 I, validation-state: unverified
                   > to 10.0.0.1 via fe-1/2/0.0
172.16.0.0/16      [BGP ] 00:00:12, localpref 100
                     AS path: 100 I, validation-state: unverified
                   > to 10.0.0.1 via fe-1/2/0.0
172.16.128.0/17    [BGP ] 00:00:12, localpref 100
                     AS path: 100 I, validation-state: unverified
                   > to 10.0.0.1 via fe-1/2/0.0
172.16.192.0/20    [BGP ] 00:00:12, localpref 100
                     AS path: 100 I, validation-state: unverified
                   > to 10.0.0.1 via fe-1/2/0.0
192.168.0.1/32     [BGP ] 00:00:12, localpref 100
                     AS path: 100 I, validation-state: unverified
                   > to 10.0.0.1 via fe-1/2/0.0
192.168.0.3/32     [BGP ] 00:00:15, localpref 100
                     AS path: 300 I, validation-state: unverified
                   > to 10.1.0.2 via fe-1/2/1.0
172.16.233.0/7        [BGP ] 00:00:12, localpref 100
                     AS path: 100 I, validation-state: unverified
                   > to 10.0.0.1 via fe-1/2/0.0
```

Meaning The output shows some routing instability. Eleven routes are hidden due to damping.

Displaying the Details of a Damped Route

Purpose Display the details of damped routes.

Action From operational mode, enter the **show route damping suppressed 172.16.192.0/20 detail** command.

```
user@R2> show route damping suppressed 172.16.192.0/20 detail

inet.0: 15 destinations, 17 routes (6 active, 0 holddown, 11 hidden)
172.16.192.0/20 (1 entry, 0 announced)
         BGP                 /-101
                Next hop type: Router, Next hop index: 758
                Address: 0x9414484
                Next-hop reference count: 9
                Source: 10.0.0.1
                Next hop: 10.0.0.1 via fe-1/2/0.0, selected
                Session Id: 0x100201
                State: <Hidden Ext>
                Local AS:   200 Peer AS:   100
                Age: 52
                Validation State: unverified
                Task: BGP_100.10.0.0.1+55922
                AS path: 100 I
                Localpref: 100
                Router ID: 192.168.0.1
                Merit (last update/now): 4278/4196
                damping-parameters: aggressive
                Last update:       00:00:52 First update:       01:01:55
                Flaps: 8
                Suppressed. Reusable in:       01:14:40
                Preference will be: 170
```

Meaning This output indicates that the displayed route has a mask length that is equal to or greater than /17, and confirms that it has been correctly mapped to the aggressive damping profile. You can also see the route's current (and last) figure of merit value, and when the route is expected to become active if it remains stable.

Verifying That Default Damping Parameters Are in Effect

Purpose Locating a damped route with a /16 mask confirms that the default parameters are in effect.

Action From operational mode, enter the **show route damping suppressed detail | match 0/16** command.

```
user@R2> show route damping suppressed detail | match 0/16

172.16.0.0/16 (1 entry, 0 announced)

user@R2> show route damping suppressed 172.16.0.0/16 detail
```



```
inet.0: 15 destinations, 17 routes (6 active, 0 holddown, 11 hidden)
172.16.0.0/16 (1 entry, 0 announced)
         BGP                 /-101
                Next hop type: Router, Next hop index: 758
                Address: 0x9414484
                Next-hop reference count: 9
                Source: 10.0.0.1
                Next hop: 10.0.0.1 via fe-1/2/0.0, selected
                Session Id: 0x100201
                State: <Hidden Ext>
                Local AS:   200 Peer AS:   100
                Age: 1:58
                Validation State: unverified
                Task: BGP_100.10.0.0.1+55922
                AS path: 100 I
                Localpref: 100
                Router ID: 192.168.0.1
                Merit (last update/now): 3486/3202
                Default damping parameters used
                Last update:       00:01:58 First update:       01:03:01
                Flaps: 8
                Suppressed. Reusable in:       00:31:40
                Preference will be: 170
```

Meaning Routes with a /16 mask are not impacted by the custom damping rules. Therefore, the default damping rules are in effect.

To repeat, the custom rules are as follows:

- Damp all prefixes with a mask length equal to or greater than 17 more aggressively than routes with a mask length between 9 and 16.
- Damp routes with a mask length between 0 and 8, inclusive, less than routes with a mask length greater than 8.
- Do not damp the 10.128.0.0/9 prefix at all.

Filtering the Damping Information

Purpose Use OR groupings or cascaded piping to simplify the determination of what damping profile is being used for routes with a given mask length.

Action From operational mode, enter the **show route damping suppressed** command.

user@R2> **show route damping suppressed detail | match "0 announced | damp"**

```
0.0.0.0/0 (1 entry, 0 announced)
                damping-parameters: timid
10.0.0.0/9 (1 entry, 0 announced)
                Default damping parameters used
                damping-parameters: aggressive
                damping-parameters: aggressive
10.224.0.0/11 (1 entry, 0 announced)
                Default damping parameters used
```



```
172.16.0.0/16 (1 entry, 0 announced)
                Default damping parameters used
172.16.128.0/17 (1 entry, 0 announced)
                damping-parameters: aggressive
172.16.192.0/20 (1 entry, 0 announced)
                damping-parameters: aggressive
192.168.0.1/32 (1 entry, 0 announced)
                damping-parameters: aggressive
192.168.0.3/32 (1 entry, 0 announced)
                damping-parameters: aggressive
172.16.233.0/7 (1 entry, 0 announced)
                damping-parameters: timid
```

Meaning When you are satisfied that your EBGP routes are correctly associated with a damping profile, you can issue the **clear bgp damping** operational mode command to restore an active status to your damped routes, which will return your connectivity to normal operation.

Related Documentation

- Understanding Damping Parameters on page 423
- Using Routing Policies to Damp BGP Route Flapping on page 424

Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family

This example shows how to configure an multiprotocol BGP multicast VPN (also called Next-Generation MVPN) with BGP route flap damping.

Requirements

This example uses Junos OS Release 12.2. BGP route flap damping support for MBGP MVPN, specifically, and on an address family basis, in general, is introduced in Junos OS Release 12.2.

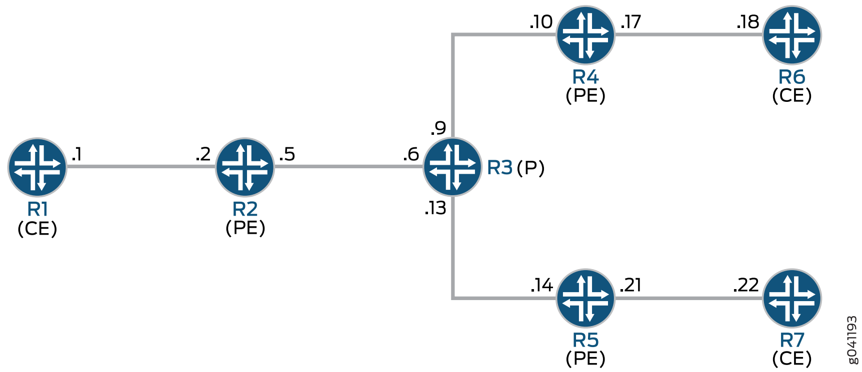
Overview

BGP route flap damping helps to diminish route instability caused by routes being repeatedly withdrawn and readvertised when a link is intermittently failing.

This example uses the default damping parameters and demonstrates an MBGP MVPN scenario with three provider edge (PE) routing devices, three customer edge (CE) routing devices, and one provider (P) routing device.

Figure 37 on page 440 shows the topology used in this example.

Figure 37: MBGP MVPN with BGP Route Flap Damping



On PE Device R4, BGP route flap damping is configured for address family **inet-mvpn**. A routing policy called **dampPolicy** uses the **nlri-route-type** match condition to damp only MVPN route types 3, 4, and 5. All other MVPN route types are not damped.

This example shows the full configuration on all devices in the "CLI Quick Configuration" on page 440 section. The "Configuring Device R4" on page 443 section shows the step-by-step configuration for PE Device R4.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces ge-1/2/0 unit 1 family inet address 10.1.1.1/30
set interfaces ge-1/2/0 unit 1 family mpls
set interfaces lo0 unit 1 family inet address 172.16.1.1/32
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.1
set protocols pim rp static address 172.16.100.1
set protocols pim interface all
set routing-options router-id 172.16.1.1
```

Device R2

```
set interfaces ge-1/2/0 unit 2 family inet address 10.1.1.2/30
set interfaces ge-1/2/0 unit 2 family mpls
set interfaces ge-1/2/1 unit 5 family inet address 10.1.1.5/30
set interfaces ge-1/2/1 unit 5 family mpls
set interfaces vt-1/2/0 unit 2 family inet
set interfaces lo0 unit 2 family inet address 172.16.1.2/32
set interfaces lo0 unit 102 family inet address 172.16.100.1/32
set protocols mpls interface ge-1/2/1.5
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.1.2
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 172.16.1.4
set protocols bgp group ibgp neighbor 172.16.1.5
set protocols ospf area 0.0.0.0 interface lo0.2 passive
```

```
set protocols ospf area 0.0.0.0 interface ge-1/2/1.5
set protocols ldp interface ge-1/2/1.5
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface ge-1/2/0.2
set routing-instances vpn-1 interface vt-1/2/0.2
set routing-instances vpn-1 interface lo0.102
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 provider-tunnel ldp-p2mp
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.102 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/0.2
set routing-instances vpn-1 protocols pim rp static address 172.16.1.2 with 172.16.4.1100.1
set routing-instances vpn-1 protocols pim interface ge-1/2/0.2 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 172.16.1.2
set routing-options autonomous-system 1001
```

Device R3

```
set interfaces ge-1/2/0 unit 6 family inet address 10.1.1.6/30
set interfaces ge-1/2/0 unit 6 family mpls
set interfaces ge-1/2/1 unit 9 family inet address 10.1.1.9/30
set interfaces ge-1/2/1 unit 9 family mpls
set interfaces ge-1/2/2 unit 13 family inet address 10.1.1.13/30
set interfaces ge-1/2/2 unit 13 family mpls
set interfaces lo0 unit 3 family inet address 172.16.1.3/32
set protocols mpls interface ge-1/2/0.6
set protocols mpls interface ge-1/2/1.9
set protocols mpls interface ge-1/2/2.13
set protocols ospf area 0.0.0.0 interface lo0.3 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.6
set protocols ospf area 0.0.0.0 interface ge-1/2/1.9
set protocols ospf area 0.0.0.0 interface ge-1/2/2.13
set protocols ldp interface ge-1/2/0.6
set protocols ldp interface ge-1/2/1.9
set protocols ldp interface ge-1/2/2.13
set protocols ldp p2mp
set routing-options router-id 172.16.1.3
```

Device R4

```
set interfaces ge-1/2/0 unit 10 family inet address 10.1.1.10/30
set interfaces ge-1/2/0 unit 10 family mpls
set interfaces ge-1/2/1 unit 17 family inet address 10.1.1.17/30
set interfaces ge-1/2/1 unit 17 family mpls
set interfaces vt-1/2/0 unit 4 family inet
set interfaces lo0 unit 4 family inet address 172.16.1.4/32
set interfaces lo0 unit 104 family inet address 172.16.100.1/32
set protocols rsvp interface all aggregate
set protocols mpls interface all
set protocols mpls interface ge-1/2/0.10
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.1.4
set protocols bgp group ibgp family inet-vpn unicast
```

```
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling damping
set protocols bgp group ibgp neighbor 172.16.1.2 import dampPolicy
set protocols bgp group ibgp neighbor 172.16.1.5
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface lo0.4 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.10
set protocols ldp interface ge-1/2/0.10
set protocols ldp p2mp
set policy-options policy-statement dampPolicy term term1 from family inet-mvpn
set policy-options policy-statement dampPolicy term term1 from nlri-route-type 3
set policy-options policy-statement dampPolicy term term1 from nlri-route-type 4
set policy-options policy-statement dampPolicy term term1 from nlri-route-type 5
set policy-options policy-statement dampPolicy term term1 then accept
set policy-options policy-statement dampPolicy then damping no-damp
set policy-options policy-statement dampPolicy then accept
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
set policy-options damping no-damp disable
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface vt-1/2/0.4
set routing-instances vpn-1 interface ge-1/2/1.17
set routing-instances vpn-1 interface lo0.104
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.104 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/1.17
set routing-instances vpn-1 protocols pim rp static address 172.16.100.1
set routing-instances vpn-1 protocols pim interface ge-1/2/1.17 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 172.16.1.4
set routing-options autonomous-system 64501
```

Device R5

```
set interfaces ge-1/2/0 unit 14 family inet address 10.1.1.14/30
set interfaces ge-1/2/0 unit 14 family mpls
set interfaces ge-1/2/1 unit 21 family inet address 10.1.1.21/30
set interfaces ge-1/2/1 unit 21 family mpls
set interfaces vt-1/2/0 unit 5 family inet
set interfaces lo0 unit 5 family inet address 172.16.1.5/32
set interfaces lo0 unit 105 family inet address 172.16.100.5/32
set protocols mpls interface ge-1/2/0.14
set protocols bgp group ibgp type internal
set protocols bgp group ibgp local-address 172.16.1.5
set protocols bgp group ibgp family inet-vpn any
set protocols bgp group ibgp family inet-mvpn signaling
set protocols bgp group ibgp neighbor 172.16.1.2
set protocols bgp group ibgp neighbor 172.16.1.4
set protocols ospf area 0.0.0.0 interface lo0.5 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.14
set protocols ldp interface ge-1/2/0.14
set protocols ldp p2mp
set policy-options policy-statement parent_vpn_routes from protocol bgp
set policy-options policy-statement parent_vpn_routes then accept
```



```
set routing-instances vpn-1 instance-type vrf
set routing-instances vpn-1 interface vt-1/2/0.5
set routing-instances vpn-1 interface ge-1/2/1.21
set routing-instances vpn-1 interface lo0.105
set routing-instances vpn-1 route-distinguisher 100:100
set routing-instances vpn-1 vrf-target target:1:1
set routing-instances vpn-1 protocols ospf export parent_vpn_routes
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface lo0.105 passive
set routing-instances vpn-1 protocols ospf area 0.0.0.0 interface ge-1/2/1.21
set routing-instances vpn-1 protocols pim rp static address 172.16.100.2
set routing-instances vpn-1 protocols pim interface ge-1/2/1.21 mode sparse
set routing-instances vpn-1 protocols mvpn
set routing-options router-id 172.16.1.5
set routing-options autonomous-system 1001
```

Device R6

```
set interfaces ge-1/2/0 unit 18 family inet address 10.1.1.18/30
set interfaces ge-1/2/0 unit 18 family mpls
set interfaces lo0 unit 6 family inet address 172.16.1.6/32
set protocols sap listen 233.1.1.1
set protocols ospf area 0.0.0.0 interface lo0.6 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.18
set protocols pim rp static address 172.16.100.2
set protocols pim interface all
set routing-options router-id 172.16.1.6
```

Device R7

```
set interfaces ge-1/2/0 unit 22 family inet address 10.1.1.22/30
set interfaces ge-1/2/0 unit 22 family mpls
set interfaces lo0 unit 7 family inet address 172.16.1.7/32
set protocols ospf area 0.0.0.0 interface lo0.7 passive
set protocols ospf area 0.0.0.0 interface ge-1/2/0.22
set protocols pim rp static address 172.16.100.2
set protocols pim interface all
set routing-options router-id 172.16.1.7
```

Configuring Device R4

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R4:

1. Configure the interfaces.

```
[edit interfaces]
user@R4# set ge-1/2/0 unit 10 family inet address 10.1.1.10/30
user@R4# set ge-1/2/0 unit 10 family mpls

user@R4# set ge-1/2/1 unit 17 family inet address 10.1.1.17/30
user@R4# set ge-1/2/1 unit 17 family mpls

user@R4# set vt-1/2/0 unit 4 family inet
```

```
user@R4# set lo0 unit 4 family inet address 172.16.1.4/32
user@R4# set lo0 unit 104 family inet address 172.16.100.4/32
```

2. Configure MPLS and the signaling protocols on the interfaces.

```
[edit protocols]
user@R4# set mpls interface all
user@R4# set mpls interface ge-1/2/0.10
user@R4# set rsvp interface all aggregate
user@R4# set ldp interface ge-1/2/0.10
user@R4# set ldp p2mp
```

3. Configure BGP.

   The BGP configuration enables BGP route flap damping for the **inet-mvpn** address family. The BGP configuration also imports into the routing table the routing policy called **dampPolicy**. This policy is applied to neighbor PE Device R2.

```
[edit protocols bgp group ibgp]
user@R4# set type internal
user@R4# set local-address 172.16.1.4
user@R4# set family inet-vpn unicast
user@R4# set family inet-vpn any
user@R4# set family inet-mvpn signaling damping
user@R4# set neighbor 172.16.1.2 import dampPolicy
user@R4# set neighbor 172.16.1.5
```

4. Configure an interior gateway protocol.

```
[edit protocols ospf]
user@R4# set traffic-engineering

[edit protocols ospf area 0.0.0.0]
user@R4# set interface all
user@R4# set interface lo0.4 passive
user@R4# set interface ge-1/2/0.10
```

5. Configure a damping policy that uses the **nlri-route-type** match condition to damp only MVPN route types 3, 4, and 5.

```
[edit policy-options policy-statement dampPolicy term term1]
user@R4# set from family inet-mvpn
user@R4# set from nlri-route-type 3
user@R4# set from nlri-route-type 4
user@R4# set from nlri-route-type 5
user@R4# set then accept
```


6. Configure the **damping** policy to disable BGP route flap damping.

   The **no-damp** policy (**damping no-damp disable**) causes any damping state that is present in the routing table to be deleted. The **then damping no-damp** statement applies the **no-damp** policy as an action and has no **from** match conditions. Therefore, all routes that are not matched by **term1** are matched by this term, with the result that all other MVPN route types are not damped.

   ```
   [edit policy-options policy-statement dampPolicy]
   user@R4# set then damping no-damp
   user@R4# set then accept
   ```

   ```
   [edit policy-options]
   user@R4# set damping no-damp disable
   ```

7. Configure the **parent_vpn_routes** to accept all other BGP routes that are not from the **inet-mvpn** address family.

   This policy is applied as an OSPF export policy in the routing instance.

   ```
   [edit policy-options policy-statement parent_vpn_routes]
   user@R4# set from protocol bgp
   user@R4# set then accept
   ```

8. Configure the VPN routing and forwarding (VRF) instance.

   ```
   [edit routing-instances vpn-1]
   user@R4# set instance-type vrf
   user@R4# set interface vt-1/2/0.4
   user@R4# set interface ge-1/2/1.17
   user@R4# set interface lo0.104
   user@R4# set route-distinguisher 100:100
   user@R4# set vrf-target target:1:1
   user@R4# set protocols ospf export parent_vpn_routes
   user@R4# set protocols ospf area 0.0.0.0 interface lo0.104 passive
   user@R4# set protocols ospf area 0.0.0.0 interface ge-1/2/1.17
   user@R4# set protocols pim rp static address 172.16.100.2
   user@R4# set protocols pim interface ge-1/2/1.17 mode sparse
   user@R4# set protocols mvpn
   ```

9. Configure the router ID and the autonomous system (AS) number.

   ```
   [edit routing-options]
   user@R4# set router-id 172.16.1.4
   user@R4# set autonomous-system 1001
   ```

10. If you are done configuring the device, commit the configuration.

   ```
   user@R4# commit
   ```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, **show routing-instances**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R4# show interfaces
ge-1/2/0 {
   unit 10 {
      family inet {
         address 10.1.1.10/30;
      }
      family mpls;
   }
}
ge-1/2/1 {
   unit 17 {
      family inet {
         address 10.1.1.17/30;
      }
      family mpls;
   }
}
vt-1/2/0 {
   unit 4 {
      family inet;
   }
}
lo0 {
   unit 4 {
      family inet {
         address 172.16.1.4/32;
      }
   }
   unit 104 {
      family inet {
         address 172.16.100.4/32;
      }
   }
}
```

```
user@R4# show protocols
rsvp {
   interface all {
      aggregate;
   }
}
mpls {
   interface all;
   interface ge-1/2/0.10;
}
bgp {
   group ibgp {
      type internal;
```



```
        local-address 172.16.1.4;
        family inet-vpn {
            unicast;
            any;
        }
        family inet-mvpn {
            signaling {
                damping;
            }
        }
        neighbor 172.16.1.2 {
            import dampPolicy;
        }
        neighbor 172.16.1.5;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface all;
        interface lo0.4 {
            passive;
        }
        interface ge-1/2/0.10;
    }
}
ldp {
    interface ge-1/2/0.10;
    p2mp;
}
```

```
user@R4# show policy-options
policy-statement dampPolicy {
    term term1 {
        from {
            family inet-mvpn;
            nlri-route-type [ 3 4 5 ];
        }
        then accept;
    }
    then {
        damping no-damp;
        accept;
    }
}
policy-statement parent_vpn_routes {
    from protocol bgp;
    then accept;
}
damping no-damp {
    disable;
}
```

```
user@R4# show routing-instances
vpn-1 {
    instance-type vrf;
    interface vt-1/2/0.4;
```

```
        interface ge-1/2/1.17;
        interface lo0.104;
        route-distinguisher 100:100;
        vrf-target target:1:1;
        protocols {
            ospf {
                export parent_vpn_routes;
                area 0.0.0.0 {
                    interface lo0.104 {
                        passive;
                    }
                    interface ge-1/2/1.17;
                }
            }
            pim {
                rp {
                    static {
                        address 172.16.100.2;
                    }
                }
                interface ge-1/2/1.17 {
                    mode sparse;
                }
            }
            mvpn;
        }
    }
```

```
user@R4# show routing-optons
router-id 172.16.1.4;
autonomous-system 1001;
```

Verification

Confirm that the configuration is working properly.

- Verifying That Route Flap Damping Is Disabled on page 448
- Verifying Route Flap Damping on page 449

Verifying That Route Flap Damping Is Disabled

Purpose Verify the presence of the **no-damp** policy, which disables damping for MVPN route types other than 3, 4, and 5.

Action From operational mode, enter the **show policy damping** command.

```
user@R4> show policy damping
Default damping information:
  Halflife: 15 minutes
  Reuse merit: 750 Suppress/cutoff merit: 3000
  Maximum suppress time: 60 minutes
  Computed values:
    Merit ceiling: 12110
    Maximum decay: 6193
```

```
Damping information for "no-damp":
 Damping disabled
```

Meaning The output shows that the default damping parameters are in effect and that the **no-damp** policy is also in effect for the specified route types.

Verifying Route Flap Damping

Purpose Check whether BGP routes have been damped.

Action From operational mode, enter the **show bgp summary** command.

```
user@R4> show bgp summary
Groups: 1 Peers: 2 Down peers: 0
Table          Tot Paths  Act Paths Suppressed    History Damp State    Pending
bgp.l3vpn.0
                       6          6          0          0          0          0
bgp.l3vpn.2
                       0          0          0          0          0          0
bgp.mvpn.0
                       2          2          0          0          0          0
Peer                     AS      InPkt     OutPkt    OutQ   Flaps Last Up/Dwn
State|#Active/Received/Accepted/Damped...
172.16.1.2             1001       3159       3155       0       0    23:43:47
 Establ
  bgp.l3vpn.0: 3/3/3/0
  bgp.l3vpn.2: 0/0/0/0
  bgp.mvpn.0: 1/1/1/0
  vpn-1.inet.0: 3/3/3/0
  vpn-1.mvpn.0: 1/1/1/0
172.16.1.5             1001       3157       3154       0       0    23:43:40
 Establ
  bgp.l3vpn.0: 3/3/3/0
  bgp.l3vpn.2: 0/0/0/0
  bgp.mvpn.0: 1/1/1/0
  vpn-1.inet.0: 3/3/3/0
  vpn-1.mvpn.0: 1/1/1/0
```

Meaning The Damp State field shows that zero routes in the bgp.mvpn.0 routing table have been damped. Further down, the last number in the State field shows that zero routes have been damped for BGP peer 172.16.1.2.

Related Documentation

- Understanding Damping Parameters on page 423
- Using Routing Policies to Damp BGP Route Flapping on page 424
- Example: Configuring BGP Route Flap Damping Parameters on page 430

CHAPTER 9

Tracking Traffic Usage with Source Class Usage and Destination Class Usage Actions

Understanding Source Class Usage and Destination Class Usage Options

You can maintain packet counts based on the entry and exit points for traffic passing through your network. Entry and exit points are identified by source and destination prefixes grouped into disjoint sets defined as source classes and *destination classes*. You can define classes based on a variety of parameters, such as routing neighbors, autonomous systems, and route filters.

Source class usage (SCU) counts packets sent to customers by performing lookups on the IP source address and the IP destination address. SCU makes it possible to track traffic originating from specific prefixes on the provider core and destined for specific prefixes on the customer edge. You must enable SCU accounting on both the inbound and outbound physical interfaces.

Destination class usage (DCU) counts packets from customers by performing lookups of the IP destination address. DCU makes it possible to track traffic originating from the customer edge and destined for specific prefixes on the provider core router.

On T Series Core Routers and M320 Multiservice Edge Routers, the source class and destination classes are not carried across the platform fabric. The implications of this are as follows:

- On T Series and M320 routers, SCU and DCU accounting is performed before the packet enters the fabric.
- On T Series and M320 routers, DCU is performed before output filters are evaluated.
- On M Series platforms, DCU is performed after output filters are evaluated.
- If an output filter drops traffic on M Series devices, the dropped packets are excluded from DCU statistics.
- If an output filter drops traffic on T Series and M320 routers, the dropped packets are included in DCU statistics.

NOTE: SCU and DCU is supported on PTX series routers when *enhanced-mode* is configured on the chassis.

On MX Series platforms with MPC/MIC interfaces, SCU and DCU are performed after output filters are evaluated. Packets dropped by output filters are not included in SCU or DCU statistics.

On MX Series platforms with non-MPC/MIC interfaces, SCU and DCU are performed before output filters are evaluated. Packets dropped by output filters are included in SCU and DCU statistics.

On PTX Series platforms, SCU and DCU accounting is performed before output filters are evaluated. Packets dropped by output filters are included in SCU and DCU statistics.

On Enhanced Scaling FPCs (T640-FPC1-ES, T640-FPC2-ES, T640-FPC3-ES, T640-FPC4-1P-ES , and T1600-FPC4-ES), the source class accounting is performed at ingress. Starting with Junos OS Release 14.2, the SCU accounting is performed at ingress on a T4000 Type 5 FPC. The implications of this are as follows:

- SCU accounting is performed when packets traverse from T4000 Type 5 FPC (ingress FPC) to Enhanced Scaling FPCs (egress FPC).
- SCU accounting is performed when packets traverse from Enhanced Scaling FPCs (ingress FPC) to T4000 Type 5 FPC (egress FPC).

NOTE: When the interface statistics are cleared and then the routing engine is replaced, the SCU and DCU statistics will not match the statistics of the previous routing engine.

For more information about source class usage, see the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide* and the *Junos OS Network Interfaces Library for Routing Devices*.

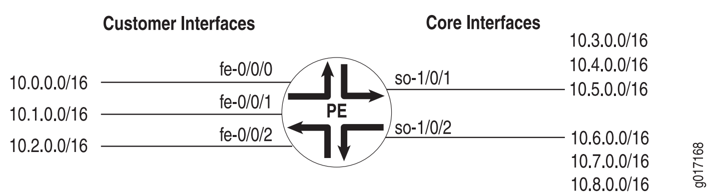
Related Documentation

- Example: Grouping Source and Destination Prefixes into a Forwarding Class on page 479
- *Configuring SCU or DCU*
- *Configuring SCU on a Virtual Loopback Tunnel Interface*
- *Configuring Class Usage Profiles*
- *Configuring the MIB Profile*
- *Configuring the Routing Engine Profile*

Source Class Usage Overview

Source class usage (SCU) is a logical extension of the destination class usage (DCU) concept. DCU was created so that Juniper Networks customers could count on a per-interface basis how much traffic was sent to specified prefixes. Figure 38 on page 453 shows a service provider edge (PE) router diagram.

Figure 38: DCU/SCU Concept



The Fast Ethernet interfaces contain inbound traffic from customers, and the SONET/SDH interfaces are connected to outbound public network prefixes. With DCU configured on the Fast Ethernet interfaces, you can track how much traffic is sent to a specific prefix in the core of the network originating from one of the specified interfaces (in this case, the Fast Ethernet interfaces).

However, DCU limits your ability to keep track of traffic moving in the reverse direction. It can account for all traffic that arrives on a core interface and heads toward a specific customer, but it cannot count traffic that arrives on a core interface from a specific prefix. For example, DCU can process cumulative traffic headed toward interface **fe-0/0/0**, but cannot differentiate between traffic coming only from **10.3.0.0/16** and traffic coming from all prefixes.

You can track source-based traffic by using SCU, which allows you to monitor the amount of traffic originating from a specific prefix. With this feature, usage can be tracked and customers can be billed for the traffic they receive.

Related Documentation

- System Requirements for SCU on page 455
- Roadmap for Configuring SCU on page 456
- Roadmap for Configuring SCU with Layer 3 VPNs on page 457
- SCU Configuration on page 463
- SCU with Layer 3 VPNs Configuration on page 471

Guidelines for Configuring SCU

When you enable SCU or DCU, keep the following information in mind:

- In Junos OS Release 5.6 and later for M Series routers only, you can use a source class or a destination class as a match condition in a firewall filter. To configure, include the **destination-class** or **source-class** statement at the **[edit firewall filter *firewall-name* term *term-name* from]** hierarchy level. For more information about firewall filters, see the *Junos Policy Framework Configuration Guide*.
- You can assign up to 126 source classes and 126 destination classes.
- When configuring policy action statements, you can configure only one source class for each matching route. In other words, more than one source class cannot be applied to the same route.
- A source or destination class is applied to a packet only once during the routing table lookup. When a network prefix matches a class-usage policy, SCU is assigned to packets first; DCU is assigned only if SCU has not been assigned. Be careful when using both class types, since misconfiguration can result in uncounted packets. The following example explores one potential mishap:

  A packet arrives on a router interface configured for both SCU and DCU. The packet's source address matches an SCU class, and its destination matches a DCU class. Consequently, the packet is subjected to a source lookup and is marked with the SCU class. The DCU class is ignored. As a result, the packet is forwarded to the outbound interface with only the SCU class still intact.

  However, the outbound interface lacks an SCU configuration. When the packet is ready to leave the router, the router detects that the output interface is not configured for SCU and the packet is not counted by SCU. Likewise, even though the prefix matched the DCU prefix, the DCU counters do not increment because DCU was superseded by SCU at the inbound interface.

To solve this problem, make sure you configure both the inbound and outbound interfaces completely or configure only one class type per interface per direction.

- Classes cannot be mapped to directly connected prefixes configured on local interfaces. This is true for DCU and SCU classes.
- If you use multiple terms within a single policy, you only need to configure the policy name and apply it to the forwarding table once. This makes it easier to change options within your terms without having to reconfigure the main policy.
- Execute command line interface (CLI) **show** commands and accounting profiles at the desired outbound interface to track SCU traffic. SCU counters increment at the SCU **output** interface.
- Apply your classes to the inbound and outbound interfaces by means of the **input** and **output** SCU interface parameters.
- On M320 and T Series routers, the source and destination classes are not carried across the platform fabric. For these routers, SCU and DCU accounting is performed before the packet enters the fabric and DCU is performed before output filters are evaluated.
- If an output filter drops traffic on M Series routers other than the M120 router and M320 router, the dropped packets are excluded from DCU statistics. If an output filter drops traffic on M320 and T Series routers, the dropped packets are included in DCU statistics.

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- Roadmap for Configuring SCU on page 456
- SCU Configuration on page 463

System Requirements for SCU

To implement SCU, your system must meet these requirements:

- Junos OS Release 8.2 or later for M120 and MX Series router support
- Junos OS Release 6.2 or later for IPv6 SCU
- Junos OS Release 5.6 or later to use a source class or a destination class as a match condition in a firewall filter
- Junos OS Release 5.4 or later for IPv4 SCU
- Three Juniper Networks M Series, MX Series, or T Series routers for basic SCU and five routers for SCU with Layer 3 VPNs. One router acts as a source class usage transit router, and the other routers are used to generate traffic or participate in the Layer 3 VPN.
- For M Series and T Series routers, a Tunnel Services PIC for SCU with Layer 3 VPNs

Related Documentation

- Source Class Usage Overview on page 453
- Roadmap for Configuring SCU on page 456
- Roadmap for Configuring SCU with Layer 3 VPNs on page 457
- SCU Configuration on page 463
- SCU with Layer 3 VPNs Configuration on page 471

Terms and Acronyms for SCU

D

destination address (DA)
: The IP address of a device intended as the receiver for a packet. This address is included in the IP header and is the main address analyzed by the router during routing table lookups and DCU.

destination class usage (DCU)
: A method of grouping certain types of traffic and monitoring these groups through CLI **show** commands, accounting profiles, or SNMP. DCU uses a destination address lookup when determining group membership. For more information about DCU, see the *Junos Policy Framework Configuration Guide*.

S

source address (SA)
: The IP address of a device sending a packet. This address is included in the IP header and is analyzed by the router for a variety of services, including source-based filtering, policing, class of service (CoS), and SCU.

source class usage (SCU)
: A method of grouping certain types of traffic and monitoring these groups through CLI **show** commands, accounting profiles, or SNMP. SCU uses a source address lookup when determining group membership. For more information about SCU, see the *Junos Policy Framework Configuration Guide*.

Related Documentation

- Source Class Usage Overview on page 453
- Roadmap for Configuring SCU on page 456
- Roadmap for Configuring SCU with Layer 3 VPNs on page 457

Roadmap for Configuring SCU

To configure source class usage (SCU), you must:

1. Create a routing policy that includes prefix route filters that indicate the IPv4 or IPv6 source addresses to monitor. See “Configuring Route Filters and Source Classes in a Routing Policy” on page 457.
2. Apply the filters to the forwarding table. See “Applying the Policy to the Forwarding Table” on page 458.

3. Enable accounting on the inbound and outbound interfaces. See “Enabling Accounting on Inbound and Outbound Interfaces” on page 459.

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- SCU Configuration on page 463

Roadmap for Configuring SCU with Layer 3 VPNs

SCU can be implemented over regular interfaces; it is also used in combination with Layer 3 VPNs. When you view SCU traffic on an ingress provider edge (PE) router, use the standard procedure outlined in “Roadmap for Configuring SCU” on page 456. However, when you enable packet counting for Layer 3 VPNs at the egress point of the MPLS tunnel, you need to take some additional steps, as follows:

1. Configure SCU on the virtual loopback tunnel (vt) interface of the egress PE router. See “Configuring Input SCU on the vt Interface of the Egress PE Router” on page 459.
2. Map the SCU-enabled input interface of that router to the virtual routing and forwarding (VRF) instance. See “Mapping the SCU-Enabled vt Interface to the VRF Instance” on page 460.
3. Configure SCU on the output interface of the egress router. See “Configuring SCU on the Output Interface” on page 461.
4. Configure an accounting profile and associate the source class with that accounting profile. You can also specify the filename for the data capture, a class usage profile name, and an interval indicating how often you want the SCU information to be saved. See “Associating an Accounting Profile with SCU Classes” on page 461.

NOTE: SCU is not supported over Layer 2 VPNs.

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- SCU with Layer 3 VPNs Configuration on page 471

Configuring Route Filters and Source Classes in a Routing Policy

Begin configuring SCU by creating prefix route filters in a policy statement. These prefixes indicate the IPv4 or IPv6 source addresses to monitor. Within the policy statement, you must define and name the source classes attached to the filters.

```
[edit policy-options]
policy-statement policy-name {
  term term-name {
```

```
      from {
        route-filter address/prefix;
      }
      then source-class class-name;
    }
  }
```

NOTE: When configuring policy action statements, you can configure only one source class for each matching route. In other words, more than one source class cannot be applied to the same route.

An alternate configuration method, using the **forwarding-class** policy action, is even more flexible. It allows your IPv4 or IPv6 route filters to apply to an SCU profile, a DCU profile, or both simultaneously. Additionally, if you have only one term, you can implement the **from** and **then** statements at the **[edit policy-options policy-statement *policy-name*]** hierarchy level.

```
[edit policy-options]
policy-statement policy-name {
  from {
    route-filter 105.15.0.0/16 orlonger;
  }
  then forwarding-class class-name;
}
```

A third option is the existing DCU parameter of **destination-class**. For more information on DCU, see the *Junos Policy Framework Configuration Guide*.

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- Roadmap for Configuring SCU on page 456
- SCU Configuration on page 463

Applying the Policy to the Forwarding Table

Next, apply the policy you created to the forwarding table. When you apply the policy, the network prefixes you defined are marked with the appropriate source class.

```
[edit routing-options]
forwarding-table {
  export policy-name;
}
```

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- Roadmap for Configuring SCU on page 456
- SCU Configuration on page 463

Enabling Accounting on Inbound and Outbound Interfaces

Unlike DCU, which only requires implementation on a single interface, accounting for SCU must be enabled on two interfaces: the inbound and outbound physical or logical interfaces traversed by the source class. You must define explicitly the two interfaces on which SCU monitored traffic is expected to arrive and depart. This is because SCU performs two lookups in the routing table: a source address (SA) and a destination address (DA) lookup. In contrast, DCU only has a single destination address lookup. By specifying the addresses involved in the additional SCU SA lookup, you minimize the performance impact on your router.

An individual SCU interface can be configured as an input interface, an output interface, or both. SCU can be enabled in an IPv4 (**family inet**) or IPv6 (**family inet6**) network. To configure SCU accounting, include the **source-class-usage** statement at the **[edit interfaces *interface-name* unit *logical-unit-number* family (inet | inet 6) accounting]** hierarchy level:

```
[edit]
interfaces {
  interface-name {
    unit unit-number {
      family (inet | inet6) {
        accounting {
          source-class-usage {
            (input | output | input output);
          }
          destination-class-usage;
        }
      }
    }
  }
}
```

After the full SCU configuration is enabled, every packet arriving on an SCU input interface is subjected to an SA-based lookup and then a DA-based lookup. In addition, an individual set of counters for every configured SCU class is maintained by the router on a per-interface and per-protocol family basis.

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- Roadmap for Configuring SCU on page 456
- SCU Configuration on page 463

Configuring Input SCU on the vt Interface of the Egress PE Router

To enable SCU in a Layer 3 VPN, configure source class usage on the virtual loopback tunnel (**vt**) interface of the egress PE router that is either configured for or equipped with a Tunnel PIC. The interface is equivalent to the inbound SCU interface, so use the **input** statement at the **[edit interfaces *vt-interface-number* unit 0 family inet accounting source-class-usage]** hierarchy level:

```
[edit]
interfaces {
  vt-0/3/0 {
    unit 0 {
      family inet {
        accounting {
          source-class-usage {
            input;
          }
        }
      }
    }
  }
}
```

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- Roadmap for Configuring SCU with Layer 3 VPNs on page 457
- SCU with Layer 3 VPNs Configuration on page 471

Mapping the SCU-Enabled vt Interface to the VRF Instance

Next, include the VPN loopback tunnel interface in the desired VRF instance at the **[edit routing-instances *routing-instance-name*]** hierarchy level:

```
[edit]
routing-instances {
  routing-instance-name {
    instance-type vrf;
    interface at-2/1/1.0;
    interface vt-0/3/0.0;
    route-distinguisher 10.250.14.225:100;
    vrf-import import-policy-name;
    vrf-export export-policy-name;
    protocols {
      bgp {
        group to-r4 {
          local-address 10.20.253.1;
          peer-as 400;
          neighbor 10.20.253.2;
        }
      }
    }
  }
}
```

Related Documentation

- Source Class Usage Overview on page 453

- System Requirements for SCU on page 455
- Roadmap for Configuring SCU with Layer 3 VPNs on page 457
- SCU with Layer 3 VPNs Configuration on page 471

Configuring SCU on the Output Interface

Since VPN traffic enters the egress router through the VPN loopback tunnel interface, you still need to determine the exit interface for this traffic. To complete your SCU configuration, configure the output version of source class usage on the exit interface of your egress router:

```
[edit interfaces]
at-1/1/0 {
   unit 0 {
      family inet {
         accounting {
            source-class-usage {
               output;
            }
         }
      }
   }
}
```

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- Roadmap for Configuring SCU with Layer 3 VPNs on page 457
- SCU with Layer 3 VPNs Configuration on page 471

Associating an Accounting Profile with SCU Classes

Once your source classes are defined, implemented on the inbound and outbound interfaces, and applied to the forwarding table, you are ready to associate the source class with an accounting profile. Configure the accounting profile at the **[edit accounting-options class-usage-profile]** hierarchy level. You can associate either an SCU source class or a DCU destination class with the accounting profile. You can also specify the filename for the data capture, a class usage profile name, and an interval (in minutes) indicating how often you want the SCU information to be saved to the file.

```
[edit]
accounting-options {
   file filename;
   class-usage-profile profile-name {
      file filename;
      interval minutes;
      source-classes {
         source-class-name;
      }
```



```
        destination-classes {
          destination-class-name;
        }
      }
    }
```

NOTE: SCU accounting occurs on the outbound interface before output filter processing. If an SCU-marked packet is discarded in the router, the SCU counters can indicate more traffic than actually exists. You must use filter counters or traceoptions logs to ensure that all packets dropped by the SCU filter are recorded. If logged, you can subtract the discarded packets from the SCU counter tallies and calculate the true traffic profile.

Because DCU accounting occurs after the filtering process, DCU is unaffected by this disclaimer.

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- Roadmap for Configuring SCU with Layer 3 VPNs on page 457
- SCU with Layer 3 VPNs Configuration on page 471

Verifying Your SCU Accounting Profile

Purpose To view the results of the SCU accounting profile you created.

Action Navigate to the **/var/log** directory of your router. It should contain the designated class usage profile log. The layout of an SCU profile looks like this:

```
profile_name,epoch-timestamp,interface-name,source-class-name,packet-count,
byte-count
```

An example of the actual output from a profile looks like this:

```
scu_profile,980313078,ge-1/0/0.0,gold,82,6888
scu_profile,980313078,ge-1/0/0.0,silver,164,13776
scu_profile,980313078,ge-1/0/0.0,bronze,0,0
scu_profile,980313678,ge-1/0/0.0,gold,82,6888
scu_profile,980313678,ge-1/0/0.0,silver,246,20664
scu_profile,980313678,ge-1/0/0.0,bronze,0,0
```

To view the parameters of your SCU accounting profile, you can use the **show accounting-options class-usage-profile *scu-profile-name*** command.

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455

SCU Configuration

Configuring SCU

Figure 39: SCU Topology Diagram

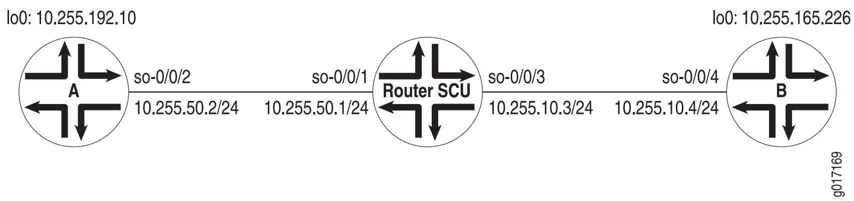


Figure 39 on page 463 shows a basic SCU configuration with three routers. Source routers A and B use loopback addresses as the prefixes to be monitored. Most of the configuration tasks and actual monitoring occurs on transit Router SCU.

Begin your configuration on Router A. The loopback address on Router A contains the origin of the prefix that is to be assigned to source class A on Router SCU. However, no SCU processing happens on this router. Therefore, configure Router A for basic OSPF routing and include your loopback interface and interface **so-0/0/2** in the OSPF process.

Router A:

```
[edit]
interfaces {
  so-0/0/2 {
    unit 0 {
      family inet {
        address 10.255.50.2/24;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.192.10/32;
      }
    }
  }
}
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/2.0;
      interface lo0.0;
    }
  }
}
```


Router SCU handles the bulk of the activity in this example. On Router SCU, enable source class usage on the inbound and outbound interfaces at the **[edit interfaces *interface-name* unit *unit-number* family inet accounting]** hierarchy level. Make sure you specify the expected traffic: input, output, or, in this case, both.

Next, configure a route filter policy statement that matches the prefixes of the loopback addresses from routers A and B. Include statements in the policy that classify packets from Router A in one group named **scu-class-a** and packets from Router B in a second class named **scu-class-b**. Notice the efficient use of a single policy containing multiple terms.

Last, apply the policy to the forwarding table.

Router SCU

```
[edit]
interfaces {
  so-0/0/1 {
    unit 0 {
      family inet {
        accounting {
          source-class-usage {
            input;
            output;
          }
        }
        address 10.255.50.1/24;
      }
    }
  }
  so-0/0/3 {
    unit 0 {
      family inet {
        accounting {
          source-class-usage {
            input;
            output;
          }
        }
        address 10.255.10.3/24;
      }
    }
  }
  lo0 {
    unit 0 {
      family inet {
        address 10.255.6.111/32;
      }
    }
  }
}
protocols {
  ospf {
    area 0.0.0.0 {
      interface so-0/0/1.0;
      interface so-0/0/3.0;
    }
```



```
    }
}
routing-options {
    forwarding-table {
        export scu-policy;
    }
}
policy-options {
    policy-statement scu-policy {
        term 0 {
            from {
                route-filter 10.255.192.0/24 orlonger;
            }
            then source-class scu-class-a;
        }
        term 1 {
            from {
                route-filter 10.255.165.0/24 orlonger;
            }
            then source-class scu-class-b;
        }
    }
}
```

Complete the configuration tasks on Router B. Just as Router A provides a source prefix, Router B's loopback address matches the prefix assigned to **scu-class-b** on Router SCU. Again, no SCU processing happens on this router, so configure Router B for basic OSPF routing and include your loopback interface and interface **so-0/0/4** in the OSPF process.

Router B:

```
[edit]
interfaces {
    so-0/0/4 {
        unit 0 {
            family inet {
                address 10.255.10.4/24;
            }
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.255.165.226/32;
            }
        }
    }
}
protocols {
    ospf {
        area 0.0.0.0 {
            interface so-0/0/4.0;
            interface lo0.0;
        }
    }
}
```

Verifying Your Work

To verify that SCU is functioning properly, use the following commands:

- **show interfaces *interface-name* statistics**
- **show interfaces *interface-name*** (**extensive** | **detail**)
- **show route** (**extensive** | **detail**)
- **show interfaces source-class *source-class-name interface-name***
- **clear interface *interface-name* statistics**

You should always verify SCU statistics at the outbound SCU interface on which you configured the **output** statement. You can perform the following three steps to check the functionality of SCU:

1. Clear all counters on your SCU-enabled router and verify that they are empty.
2. Send a ping from one edge router to another edge router to generate SCU traffic across the SCU-enabled router.
3. Verify that the counters are incrementing correctly on the outbound interface.

The following section shows the output of these commands as used with the configuration example.

```
user@scu> clear interfaces statistics all

user@scu>  show interfaces so-0/0/1.0 statistics
  Logical interface so-0/0/1.0 (Index 4) (SNMP ifIndex 119)
    Flags: Point-To-Point SNMP-Traps Encapsulation: PPP
    Protocol inet, MTU: 4470
      Source class                             Packets                Bytes
                       scu-class-a          0         0
        scu-class-b          0          0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.255.50/24, Local: 10.255.50.1

user@scu> show interfaces so-0/0/3.0 statistics
  Logical interface so-0/0/3.0 (Index 6) (SNMP ifIndex 113)
    Flags: Point-To-Point SNMP-Traps Encapsulation: PPP
    Protocol inet, MTU: 4470
      Source class                             Packets                Bytes
                      scu-class-a                    0                    0
                      scu-class-b                    0                    0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.255.10/24, Local: 10.255.10.3

user@scu> show interfaces source-class scu-class-a so-0/0/3.0
    Protocol inet
      Source class                             Packets                Bytes
                      scu-class-a                    0                    0

user@scu> show interfaces source-class scu-class-b so-0/0/1.0
    Protocol inet
      Source class                             Packets                Bytes
                      scu-class-b                    0                    0
```



```
user@routerB> ping 10.255.192.10 source 10.255.165.226 rapid 10000

user@routerA> ping 10.255.165.226 source 10.255.192.10 rapid 10000

user@scu> show interfaces source-class scu-class-a so-0/0/3.0
    Protocol inet
      Source class                             Packets               Bytes
                      scu-class-a                20000             1680000

user@scu> show interfaces source-class scu-class-a so-0/0/1.0
    Protocol inet
      Source class                             Packets               Bytes
                      scu-class-b                20000             1680000

user@scu> show interfaces so-0/0/3.0 statistics
  Logical interface so-0/0/3.0 (Index 6) (SNMP ifIndex 113)
    Flags: Point-To-Point SNMP-Traps Encapsulation: PPP
    Protocol inet, MTU: 4470
      Source class                             Packets               Bytes
                      scu-class-a        20000        1680000
                      scu-class-b                    0                   0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.255.10/24, Local: 10.255.10.3

user@scu> show interfaces so-0/0/1.0 statistics
  Logical interface so-0/0/1.0 (Index 4) (SNMP ifIndex 119)
    Flags: Point-To-Point SNMP-Traps Encapsulation: PPP
    Protocol inet, MTU: 4470
      Source class                             Packets               Bytes
                      scu-class-a                    0                   0
                      scu-class-b        20000        1680000
      Addresses, Flags: Is-Preferred Is-Primary
       Destination: 10.255.50/24, Local: 10.255.50.1
```

```
user@scu> show route extensive 10.255.192.0

inet.0: 26 destinations, 28 routes (25 active, 0 holddown, 1 hidden)
10.255.192.0/18 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.255.192.0/18 -> {so-0/0/1.0}
Source class: scu-class-a
        *OSPF   Preference: 150
                Next hop: via so-0/0/1.0, selected
                State: <Active Int Ext>
                Age: 2:49:31  Metric: 0  Tag: 0
                Task: OSPF
                Announcement bits (1): 0-KRT
                AS path: I
```

```
user@scu> show route extensive 10.255.165.0
inet.0: 26 destinations, 28 routes (25 active, 0 holddown, 1 hidden)
10.255.165.0/20 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.255.165.0/20 -> {so-0/0/3.0}
```

```
Source class: scu-class-b
        *OSPF   Preference: 150
                Next hop: via so-0/0/3.0, selected
                State: <Active Int Ext>
                Age: 2:49:31  Metric: 0  Tag: 0
                Task: OSPF
                Announcement bits (1): 0-KRT
                AS path: I
```

```
user@scu> show interfaces so-0/0/1 detail
Physical interface: so-0/0/1, Enabled, Physical link is Up
  Interface index: 12, SNMP ifIndex: 17, Generation: 11
  Link-level type: PPP, MTU: 4474, Clocking: Internal, SONET mode, Speed: OC3,
  Loopback: None, FCS: 16, Payload scrambler: Enabled
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Link flags     : Keepalives
  Hold-times     : Up 0 ms, Down 0 ms
  Keepalive settings: Interval 10 seconds, Up-count 1, Down-count 3
  Keepalive statistics:
    Input : 46 (last seen 00:00:01 ago)
    Output: 45 (last sent 00:00:00 ago)
  LCP state: Opened
  NCP state: inet: Opened, inet6: Not-configured, iso: Not-configured, mpls:
  Not-configured
  CHAP state: Not-configured
  Last flapped   : 2002-04-19 11:49:22 PDT (03:10:09 ago)
  Statistics last cleared: 2002-04-19 14:52:04 PDT (00:07:27 ago)
  Traffic statistics:
   Input  bytes  :            1689276                  40 bps
   Output bytes  :            1689747                  48 bps
   Input  packets:              20197                   0 pps
   Output packets:              20200                   0 pps
  Queue counters:       Queued packets  Transmitted packets      Dropped packets

    0 best-effort               20053                20053                    0

    1 expedited-fo                  0                    0                    0

    2 assured-forw                  0                    0                    0

    3 network-cont                146                  146                    0

  SONET alarms   : None
  SONET defects  : None
  Logical interface so-0/0/1.0 (Index 4) (SNMP ifIndex 119) (Generation 3)
    Flags: Point-To-Point SNMP-Traps Encapsulation: PPP
    Protocol inet, MTU: 4470
      Flags: SCU-in, SCU-out
    Generation: 6 Route table: 0
       Source class             Packets         Bytes
         scu-class-a          0           0
         scu-class-b        20000       1680000
      Filters: Input: icmp-so-0/0/1.0-i, Output: icmp-so-0/0/1.0-o
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.255.50/24, Local: 10.255.50.1, Broadcast: Unspecified,
        Generation: 8
```



```
user@scu> show interfaces so-0/0/1 extensive
Physical interface: so-0/0/1, Enabled, Physical link is Up
  Interface index: 12, SNMP ifIndex: 17, Generation: 11
  Link-level type: PPP, MTU: 4474, Clocking: Internal, SONET mode, Speed: OC3,
  Loopback: None, FCS: 16, Payload scrambler: Enabled
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps
  Link flags     : Keepalives
  Hold-times     : Up 0 ms, Down 0 ms
  Keepalive settings: Interval 10 seconds, Up-count 1, Down-count 3
  Keepalive statistics:
    Input : 51 (last seen 00:00:04 ago)
    Output: 50 (last sent 00:00:05 ago)
  LCP state: Opened
  NCP state: inet: Opened, inet6: Not-configured, iso: Not-configured, mpls:
  Not-configured
  CHAP state: Not-configured
  Last flapped   : 2002-04-19 11:49:22 PDT (03:11:05 ago)
  Statistics last cleared: 2002-04-19 14:52:04 PDT (00:08:23 ago)
  Traffic statistics:
   Input  bytes  :             1689884                  264 bps
   Output bytes  :             1690388                  280 bps
   Input  packets:               20215                    0 pps
   Output packets:               20217                    0 pps
  Input errors:
    Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0,
    Bucket drops: 0, Policed discards: 0, L3 incompletes: 0,
    L2 channel errors: 0, L2 mismatch timeouts: 0, HS link CRC errors: 0,
    HS link FIFO overflows: 0
  Output errors:
    Carrier transitions: 0, Errors: 0, Drops: 0, Aged packets: 0,
    HS link FIFO underflows: 0
  Queue counters:       Queued packets  Transmitted packets      Dropped packets

    0 best-effort               20053                20053                    0

    1 expedited-fo                  0                    0                    0

    2 assured-forw                  0                    0                    0

    3 network-cont                164                  164                    0

  SONET alarms   : None
  SONET defects  : None
  SONET PHY:           Seconds        Count  State
    PLL Lock                 0            0  OK
    PHY Light                0            0  OK
  SONET section:
    BIP-B1                   0            0
    SEF                      0            0  OK
    LOS                      0            0  OK
    LOF                      0            0  OK
    ES-S                     0
    SES-S                    0
    SEFS-S                   0
  SONET line:
    BIP-B2                   0            0
    REI-L                    0            0
    RDI-L                    0            0  OK
    AIS-L                    0            0  OK
    BERR-SF                  0            0  OK
```



```
    BERR-SD                  0            0  OK
    ES-L                     0
    SES-L                    0
    UAS-L                    0
    ES-LFE                   0
    SES-LFE                  0
    UAS-LFE                  0
  SONET path:
    BIP-B3                   0            0
    REI-P                    0            0
    LOP-P                    0            0  OK
    AIS-P                    0            0  OK
    RDI-P                    0            0  OK
    UNEQ-P                   0            0  OK
    PLM-P                    0            0  OK
    ES-P                     0
    SES-P                    0
    UAS-P                    0
    ES-PFE                   0
    SES-PFE                  0
    UAS-PFE                  0
  Received SONET overhead:
    F1      : 0x00, J0      : 0x00, K1      : 0x00, K2      : 0x00
    S1      : 0x00, C2      : 0xcf, C2(cmp) : 0xcf, F2      : 0x00
    Z3      : 0x00, Z4      : 0x00, S1(cmp) : 0x00, V5      : 0x00
    V5(cmp) : 0x00
  Transmitted SONET overhead:
    F1      : 0x00, J0      : 0x01, K1      : 0x00, K2      : 0x00
    S1      : 0x00, C2      : 0xcf, F2      : 0x00, Z3      : 0x00
    Z4      : 0x00, V5      : 0x00
  Received path trace: e so-0/0/1
    65 20 73 6f 2d 30 2f 30 2f 31 00 00 00 00 00 00   e so-0/0/1......
    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
    00 00 00 00 00 00 00 00 00 00 00 00 00 00 0d 0a   ................
  Transmitted path trace: scu so-0/0/1
    67 68 62 20 73 6f 2d 30 2f 30 2f 31 00 00 00 00   scu so-0/0/1....
    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
  HDLC configuration:
    Policing bucket: Disabled
    Shaping bucket : Disabled
    Giant threshold: 4484, Runt threshold: 3
  Packet Forwarding Engine configuration:
    Destination slot: 0, PLP byte: 1 (0x00)
    CoS transmit queue          Bandwidth          Buffer     Priority   Limit
                             %           bps   %        bytes
    0 best-effort            0             0   0            0       low    none
    1 expedited-forwarding   0             0   0            0       low    none
    2 assured-forwarding     0             0   0            0       low    none
    3 network-control        0             0   0            0       low    none
  Logical interface so-0/0/1.0 (Index 4) (SNMP ifIndex 119) (Generation 3)
    Flags: Point-To-Point SNMP-Traps Encapsulation: PPP
    Protocol inet, MTU: 4470
      Flags: SCU-in, SCU-out
    Generation: 6 Route table: 0
       Source class             Packets         Bytes
         scu-class-a           0            0
         scu-class-b         20000        1680000
      Filters: Input: icmp-so-0/0/1.0-i, Output: icmp-so-0/0/1.0-o
```



```
Addresses, Flags: Is-Preferred Is-Primary
  Destination: 10.255.50/24, Local: 10.255.50.1, Broadcast: Unspecified,
  Generation: 8
```

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455
- Roadmap for Configuring SCU on page 456

SCU with Layer 3 VPNs Configuration

- Configuring SCU in a Layer 3 VPN on page 471
- Verifying Your Work on page 477

Configuring SCU in a Layer 3 VPN

Figure 40: SCU in a Layer 3 VPN Topology Diagram

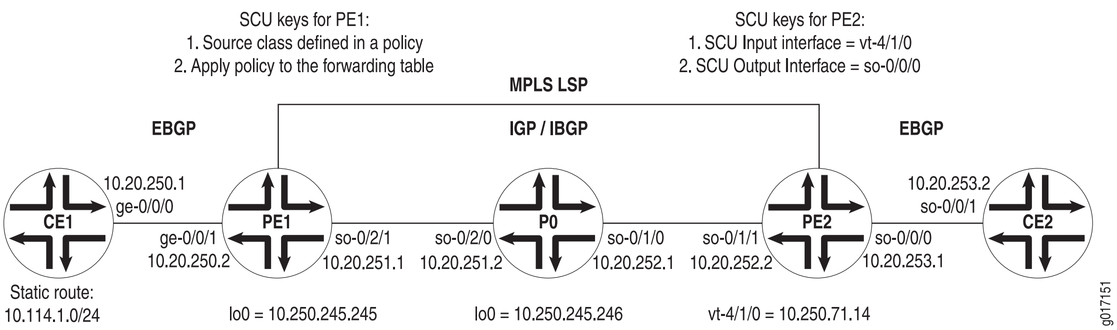


Figure 40 on page 471 displays a Layer 3 VPN topology. CE1 and CE2 are customer edge (CE) routers connected by a VPN through provider routers PE1, P0, and PE2. EBGP is established between routers CE1 and PE1, IBGP connects routers PE1 and PE2 over an IS-IS/MPLS/LDP core, and a second EBGP connection flows between routers PE2 and CE2.

On Router CE1, begin your VPN by setting up an EBGP connection to PE1. Install a static route of **10.114.1.0/24** and advertise this route to your EBGP neighbor.

Router CE1

```
[edit]
interfaces {
  ge-0/0/0 {
    unit 0 {
      family inet {
        address 10.20.250.1/30;
      }
    }
  }
}
routing-options {
  static {
    route 10.114.1.0/24 reject;
  }
```

```
    autonomous-system 100;
}
protocols {
    bgp {
        group to-pe1 {
            local-address 10.20.250.1;
            export inject-direct;
            peer-as 300;
            neighbor 10.20.250.2;
        }
    }
}
policy-options {
    policy-statement inject-direct {
        term 1 {
            from {
                protocol static;
                route-filter 10.114.1.0/24 exact;
            }
            then accept;
        }
        term 2 {
            from protocol direct;
            then accept;
        }
    }
}
```

On PE1, complete the EBGP connection to CE1 through a VRF routing instance. Set an export policy for your VRF instance that puts BGP traffic into a community, and an import policy that accepts like community traffic from your VPN neighbor. Lastly, configure an IBGP relationship to Router PE2 that runs over an IS-IS, MPLS, and LDP core.

Router PE1

```
[edit]
interfaces {
    ge-0/0/1 {
        unit 0 {
            family inet {
                address 10.20.250.2/30;
            }
        }
    }
    so-0/2/1 {
        unit 0 {
            family inet {
                address 10.20.251.1/30;
            }
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.250.245.245/32;
```



```
            }
            family iso;
            family mpls;
        }
    }
}
routing-options {
    autonomous-system 300;
}
protocols {
    mpls {
        interface so-0/2/1;
    }
    bgp {
        group ibgp {
            type internal;
            local-address 10.250.245.245;
            family inet-vpn {
                unicast;
            }
            neighbor 10.250.71.14;
        }
    }
    isis {
        interface so-0/2/1;
    }
    ldp {
        interface so-0/2/1;
    }
}
policy-options {
    policy-statement red-import {
        from {
            protocol bgp;
            community red-com;
        }
        then accept;
    }
    policy-statement red-export {
        from protocol bgp;
        then {
            community add red-com;
            accept;
        }
    }
    community red-com members target:20:20;
}
routing-instances {
    red {
        instance-type vrf;
        interface ge-0/0/1.0;
        route-distinguisher 10.250.245.245:100;
        vrf-import red-import;
        vrf-export red-export;
        protocols {
            bgp {
```

```
            group to-ce1 {
                local-address 10.20.250.2;
                peer-as 100;
                neighbor 10.20.250.1;
            }
        }
    }
  }
}
```

On P0, connect the IBGP neighbors located at PE1 and PE2. Remember to include VPN-related protocols (MPLS, LDP, and IGP) on all interfaces.

Router P0

```
[edit]
interfaces {
    so-0/1/0 {
        unit 0 {
            family inet {
                address 10.20.252.1/30;
            }
            family iso;
            family mpls;
        }
    }
    so-0/2/0 {
        unit 0 {
            family inet {
                address 10.20.251.2/30;
            }
            family iso;
            family mpls;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.250.245.246/32;
            }
            family iso;
            family mpls;
        }
    }
}
routing-options {
    autonomous-system 300;
}
protocols {
    mpls {
        interface so-0/1/0;
        interface so-0/2/0;
    }
    isis {
        interface all;
    }
    ldp {
        interface all;
```



```
    }
  }
```

On PE2, complete the IBGP relationship to Router PE1. Establish an EBGP connection to CE2 through a VRF routing instance. Set an export policy for the VRF instance that places BGP traffic into a community, and an import policy that accepts like community traffic from the VPN neighbor. Next, establish a policy that adds the static route from CE1 to a source class called **GOLD1**. Also, export this SCU policy into the forwarding table. Finally, set your **vt** interface as the SCU input interface and establish the CE-facing interface **so-0/0/0** as the SCU output interface.

Router PE2

```
[edit]
interfaces {
  so-0/1/1 {
    unit 0 {
      family inet {
        address 10.20.252.2/30;
      }
      family iso;
      family mpls;
    }
  }
  so-0/0/0 {
    unit 0 {
      family inet {
        accounting {
          source-class-usage {
            output;
          }
        }
        address 10.20.253.1/30;
      }
    }
  }
  vt-4/1/0 {
    unit 0 {
      family inet {
        accounting {
          source-class-usage {
            input;
          }
        }
        address 10.250.71.14/32;
      }
      family iso;
      family mpls;
    }
  }
}
routing-options {
  autonomous-system 300;
  forwarding-table {
    export inject-customer2-dest-class;
  }
}
```

```
protocols {
   mpls {
      interface so-0/1/1;
      interface vt-4/1/0;
   }
   bgp {
      group ibgp {
         type internal;
         local-address 10.250.71.14;
         family inet-vpn {
            unicast;
         }
         neighbor 10.250.245.245;
      }
   }
   isis {
      interface so-0/1/1;
   }
   ldp {
      interface so-0/1/1;
   }
}
routing-instances {
   red {
      instance-type vrf;
      interface so-0/0/0.0;
      interface vt-4/1/0.0;
      route-distinguisher 10.250.71.14:100;
      vrf-import red-import;
      vrf-export red-export;
      protocols {
         bgp {
            group to-ce2 {
               local-address 10.20.253.1;
               peer-as 400;
               neighbor 10.20.253.2;
            }
         }
      }
   }
}
policy-options {
   policy-statement red-import {
      from {
         protocol bgp;
         community red-com;
      }
      then accept;
   }
   policy-statement red-export {
      from protocol bgp;
      then {
         community add red-com;
         accept;
      }
   }
```



```
    policy-statement inject-customer2-dest-class {
        term term-gold1-traffic {
            from {
                route-filter 10.114.1.0/24 exact;
            }
            then source-class GOLD1;
        }
    }
    community red-com members target:20:20;
}
```

On Router CE2, complete the VPN path by finishing the EBGP connection to PE2.

Router CE2

```
[edit]
interfaces {
    so-0/0/1 {
        unit 0 {
            family inet {
                address 10.20.253.2/30;
            }
        }
    }
}
routing-options {
    autonomous-system 400;
}
protocols {
    bgp {
        group to-pe2 {
            local-address 10.20.253.2;
            export inject-direct;
            peer-as 300;
            neighbor 10.20.253.1;
        }
    }
}
policy-options {
    policy-statement inject-direct {
        from {
            protocol direct;
        }
        then accept;
    }
}
```

Verifying Your Work

To verify that SCU is functioning properly in the Layer 3 VPN, use the following commands:

- **show interfaces** ***interface-name*** **statistics**
- **show interfaces source-class** ***source-class-name interface-name***
- **show interfaces** ***interface-name*** (**extensive** | **detail**)

- **show route (extensive | detail)**
- **clear interface *interface-name* statistics**

You should always verify SCU statistics at the outbound SCU interface on which you configured the **output** statement. To check SCU functionality, follow these steps:

1. Clear all counters on your SCU-enabled router and verify they are empty.
2. Send a ping from the ingress CE router to the second CE router to generate SCU traffic across the SCU-enabled VPN route.
3. Verify that the counters are incrementing correctly on the outbound interface.

The following section shows the output of these commands used with the configuration example.

```
user@pe2> clear interfaces statistics all

 user@pe2> show interfaces so-0/0/0.0 statistics
   Logical interface so-0/0/0.0 (Index 6) (SNMP ifIndex 113)
    Flags: Point-To-Point SNMP-Traps Encapsulation: PPP
    Protocol inet, MTU: 4470
      Source class                           Packets               Bytes
 GOLD1          0        0
      Addresses, Flags: Is-Preferred Is-Primary

 user@pe2> show interfaces source-class GOLD1 so-0/0/0.0
    Protocol inet
      Source class                           Packets               Bytes
GOLD1           0        0

user@ce1> ping 10.20.253.2 source 10.114.1.1 rapid count 10000

user@scu>  show interfaces source-class GOLD1 so-0/0/0.0
    Protocol inet
      Source class                           Packets               Bytes
GOLD1          20000      1680000

user@scu>  show interfaces so-0/0/0.0 statistics
  Logical interface so-0/0/0.0 (Index 6) (SNMP ifIndex 113)
    Flags: Point-To-Point SNMP-Traps Encapsulation: PPP
    Protocol inet, MTU: 4470
      Source class                           Packets               Bytes
GOLD1          20000      1680000
      Addresses, Flags: Is-Preferred Is-Primary
         Destination: 10.20.253/24, Local: 10.20.253.1
```

Related Documentation

- Source Class Usage Overview on page 453
- System Requirements for SCU on page 455

Example: Grouping Source and Destination Prefixes into a Forwarding Class

This example shows how to group source and destination prefixes into a forwarding class.

Requirements

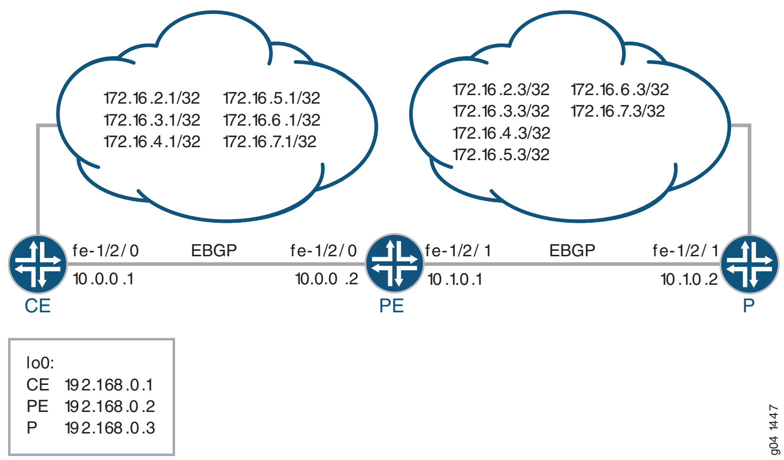
No special configuration beyond device initialization is required before configuring this example.

Overview

This example uses three routing devices: a customer edge (CE) device, a provider edge (PE) device, and a provider core (P) device.

Figure 41 on page 479 shows the sample network.

Figure 41: SCU and DCU Sample Network



Source class usage (SCU) counts packets sent to the customer edge by performing lookup on the IP source address and the IP destination address. SCU makes it possible to track traffic originating from specific prefixes on the provider core and destined for specific prefixes on the customer edge.

DCU counts packets from customers by performing a lookup of the IP destination address. DCU makes it possible to track traffic originating from the customer edge and destined for specific prefixes on the provider core router.

On Device PE's fe-1/2/1 interface, facing the provider core (represented by Device P), SCU input is configured with the **source-class-usage input** statement to track traffic originating at Device P and destined to Device CE. On this same interface, the **destination-class-usage input** statement is configured to track traffic originating at Device CE destined to the provider core.

```
user@PE# show interfaces fe-1/2/1 unit 0 family inet
accounting {
  source-class-usage {
    input; # tracks traffic destined to customer edge
  }
  destination-class-usage; # tracks traffic destined to provider core
}
address 10.1.0.1/30;
```

Unlike destination class usage (DCU), which only requires implementation on a single interface, accounting for SCU must be enabled on two interfaces: the inbound and outbound interfaces traversed by the source class. You must define explicitly the two interfaces on which SCU monitored traffic is expected to arrive and depart. This is because SCU performs two lookups in the routing table: a source address (SA) and a destination address (DA) lookup. In contrast, DCU only has a single destination address lookup.

On Device PE's fe-1/2/0 interface, facing Device CE, SCU output is configured with the **source-class-usage output** statement.

```
user@PE# show interfaces fe-1/2/0 unit 0 family inet
accounting {
  source-class-usage {
    output;
  }
}
address 10.0.0.2/30;
```

To account for traffic destined to the customer, the policy called scu_class uses route filters to place traffic into the gold1, gold2, and gold3 classes.

```
user@PE# show policy-options
policy-statement scu_class {
  term gold1 {
    from {
      route-filter 172.16.2.0/24 orlonger;
    }
    then source-class gold1;
  }
  term gold2 {
    from {
      route-filter 172.16.3.0/24 orlonger;
    }
    then source-class gold2;
  }
  term gold3 {
    from {
      route-filter 172.16.4.0/24 orlonger;
    }
    then source-class gold3;
  }
}
```


To account for traffic destined to the provider, the policy called dcu_class uses route filters to place traffic into the silver1, silver2, and silver3 classes.

```
user@PE# show policy-options
policy-statement dcu_class {
  term silver1 {
    from {
      route-filter 172.16.5.0/24 orlonger;
    }
    then destination-class silver1;
  }
  term silver2 {
    from {
      route-filter 172.16.6.0/24 orlonger;
    }
    then destination-class silver2;
  }
  term silver3 {
    from {
      route-filter 172.16.7.0/24 orlonger;
    }
    then destination-class silver3;
  }
}
```

The policies are then applied to the forwarding table.

```
forwarding-table {
  export [ dcu_class scu_class ];
}
```

The example uses static routes to provide connectivity and loopback interface addresses for testing the operation.

"CLI Quick Configuration" on page 481 shows the configuration for all of the devices in Figure 41 on page 479.

The section "Step-by-Step Procedure" on page 483 describes the steps on Device PE.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device CE

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
```



```
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.1.0.0/30 next-hop 10.0.0.2
set routing-options autonomous-system 100
```

Device PE

```
set interfaces fe-1/2/0 unit 0 family inet accounting source-class-usage output
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet accounting source-class-usage input
set interfaces fe-1/2/1 unit 0 family inet accounting destination-class-usage
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set protocols bgp group ext neighbor 10.1.0.2 peer-as 300
set policy-options policy-statement dcu_class term silver1 from route-filter 172.16.5.0/24
  orlonger
set policy-options policy-statement dcu_class term silver1 then destination-class silver1
set policy-options policy-statement dcu_class term silver2 from route-filter 172.16.6.0/24
  orlonger
set policy-options policy-statement dcu_class term silver2 then destination-class silver2
set policy-options policy-statement dcu_class term silver3 from route-filter 172.16.7.0/24
  orlonger
set policy-options policy-statement dcu_class term silver3 then destination-class silver3
set policy-options policy-statement scu_class term gold1 from route-filter 172.16.2.0/24
  orlonger
set policy-options policy-statement scu_class term gold1 then source-class gold1
set policy-options policy-statement scu_class term gold2 from route-filter 172.16.3.0/24
  orlonger
set policy-options policy-statement scu_class term gold2 then source-class gold2
set policy-options policy-statement scu_class term gold3 from route-filter 172.16.4.0/24
  orlonger
set policy-options policy-statement scu_class term gold3 then source-class gold3
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options autonomous-system 200
set routing-options forwarding-table export dcu_class
set routing-options forwarding-table export scu_class
```

Device P

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set interfaces lo0 unit 0 family inet address 172.16.0.3/32
set interfaces lo0 unit 0 family inet address 172.16.0.3/32
set interfaces lo0 unit 0 family inet address 172.16.0.3/32
set interfaces lo0 unit 0 family inet address 172.16.0.3/32
set interfaces lo0 unit 0 family inet address 172.16.0.3/32
set interfaces lo0 unit 0 family inet address 172.16.0.3/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
```



```
set protocols bgp group ext export send-static
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.1.0.1
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set routing-options static route 10.0.0.0/30 next-hop 10.1.0.1
set routing-options static route 172.16.2.0/24 discard
set routing-options static route 172.16.3.0/24 discard
set routing-options static route 172.16.4.0/24 discard
set routing-options static route 172.16.5.0/24 discard
set routing-options static route 172.16.6.0/24 discard
set routing-options static route 172.16.7.0/24 discard
set routing-options autonomous-system 300
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To group source and destination prefixes in a forwarding class:

1. Create the router interfaces.

```
[edit interfaces]
user@PE# set fe-1/2/0 unit 0 family inet accounting source-class-usage output
user@PE# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

user@PE# set fe-1/2/1 unit 0 family inet accounting source-class-usage input
user@PE# set fe-1/2/1 unit 0 family inet accounting destination-class-usage
user@PE# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30

user@PE# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure BGP.

```
[edit protocols bgp group ext]
user@PE# set type external
user@PE# set export send-direct
user@PE# set neighbor 10.0.0.1 peer-as 100
user@PE# set neighbor 10.1.0.2 peer-as 300
```

3. Configure the DCU policy.

```
[edit policy-options policy-statement dcu_class]
user@PE# set term silver1 from route-filter 172.16.5.0/24 orlonger
user@PE# set term silver1 then destination-class silver1

user@PE# set term silver2 from route-filter 172.16.6.0/24 orlonger
user@PE# set term silver2 then destination-class silver2

user@PE# set term silver3 from route-filter 172.16.7.0/24 orlonger
user@PE# set term silver3 then destination-class silver3
```

4. Configure the SCU policy.

   ```
   [edit policy-options policy-statement scu_class]
   user@PE# set term gold1 from route-filter 172.16.2.0/24 orlonger
   user@PE# set term gold1 then source-class gold1

   user@PE# set term gold2 from route-filter 172.16.3.0/24 orlonger
   user@PE# set term gold2 then source-class gold2

   user@PE# set term gold3 from route-filter 172.16.4.0/24 orlonger
   user@PE# set term gold3 then source-class gold3
   ```

5. Apply the policies to the forwarding table.

   ```
   [edit routing-options forwarding-table]
   user@PE# set export dcu_class
   user@PE# set export scu_class
   ```

   NOTE: You can refer to the same routing policy one or more times in the same or different export statement.

6. (Optional) Configure a routing policy that advertises direct routes.

   ```
   [edit policy-options policy-statement send-direct term 1]
   user@PE# set from protocol direct
   user@PE# set then accept
   ```

7. Configure the autonomous system (AS) number.

   ```
   [edit routing-options]
   user@PE# set autonomous-system 200
   ```

Results From configuration mode, confirm your configuration by issuing the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      accounting {
        source-class-usage {
          output;
        }
      }
      address 10.0.0.2/30;
    }
  }
```



```
}
fe-1/2/1 {
   unit 0 {
      family inet {
         accounting {
            source-class-usage {
               input;
            }
            destination-class-usage;
         }
         address 10.1.0.1/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.2/32;
      }
   }
}
```

```
user@PE# show protocols
bgp {
   group ext {
      type external;
      export send-direct;
      neighbor 10.0.0.1 {
         peer-as 100;
      }
      neighbor 10.1.0.2 {
         peer-as 300;
      }
   }
}
```

```
user@PE# show policy-options
policy-statement dcu_class {
   term silver1 {
      from {
         route-filter 172.16.5.0/24 orlonger;
      }
      then destination-class silver1;
   }
   term silver2 {
      from {
         route-filter 172.16.6.0/24 orlonger;
      }
      then destination-class silver2;
   }
   term silver3 {
      from {
         route-filter 172.16.7.0/24 orlonger;
      }
      then destination-class silver3;
   }
}
```

```
policy-statement scu_class {
  term gold1 {
    from {
      route-filter 172.16.2.0/24 orlonger;
    }
    then source-class gold1;
  }
  term gold2 {
    from {
      route-filter 172.16.3.0/24 orlonger;
    }
    then source-class gold2;
  }
  term gold3 {
    from {
      route-filter 172.16.4.0/24 orlonger;
    }
    then source-class gold3;
  }
}
policy-statement send-direct {
  term 1 {
    from protocol direct;
    then accept;
  }
}
```

```
user@PE# show routing-options
autonomous-system 200;
forwarding-table {
  export [ dcu_class scu_class ];
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Making Sure That the DCU Policy Is Working on page 486
- Making Sure That the SCU Policy Is Working on page 487

Making Sure That the DCU Policy Is Working

Purpose Verify that traffic sent from the provider core into the customer network is causing the DCU policy counters to increment.

Action 1. From Device P, ping an address in the customer network.

```
user@P> ping rapid count 10000000 172.16.0.1

PING 172.16.0.1 (6.0.0.1): 56 data bytes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```


2. On Device PE, check the interface statistics on the interface facing the provider core.

```
user@PE> show interfaces statistics fe-1/2/1.0
```

```
  Logical interface fe-1/2/1.0 (Index 108) (SNMP ifIndex 546)
    Flags: SNMP-Traps 0x4000 Encapsulation: ENET2
    Input packets : 251956
    Output packets: 251961
    Protocol inet, MTU: 1500
      Flags: Sendbcast-pkt-to-re, DCU, SCU-in
                                            Packets                Bytes
      Destination class            (packet-per-second)    (bits-per-second)

                        silver1                7460               626640
                                (                 0) (                 0)
                        silver2               22440              2401416
                                (               256) (            171963)
                        silver3                9004               756336
                                (                 0) (                 0)
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.1.0.0/30, Local: 10.1.0.1, Broadcast: 10.1.0.3
```

Meaning Packet and bit rates are displayed with packet and byte counters.

Alternatively, you can use the **show interfaces destination-class all** command to display the same information.

Making Sure That the SCU Policy Is Working

Purpose Verify that traffic sent from the customer network into the provider core is causing the SCU policy counters to increment.

Action

1. From Device CE, ping an address in the customer network.

```
user@CE> ping rapid count 10000000 172.16.0.1
```

```
PING 172.16.0.1 (6.0.0.1): 56 data bytes
```

2. On Device PE, check the interface statistics on the interface facing the customer network.

```
user@PE> show interfaces statistics fe-1/2/0.0
```

```
  Logical interface fe-1/2/0.0 (Index 93) (SNMP ifIndex 554)
    Flags: SNMP-Traps 0x4000 Encapsulation: ENET2
    Input packets : 32246
    Output packets: 32245
    Protocol inet, MTU: 1500
      Flags: Sendbcast-pkt-to-re, Is-Primary, SCU-out
                                            Packets                Bytes
      Source class                 (packet-per-second)    (bits-per-second)

                          gold1                8871               745164
                                (               259) (            174497)
```

```
                              gold2                1812                152208
                                    (                 0) (                  0)
                              gold3                5711                479724
                                    (                 0) (                  0)
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.0.0.0/30, Local: 10.0.0.2, Broadcast: 10.0.0.3
```

Meaning Packet and bit rates are displayed with packet and byte counters.

Alternatively, you can use the **show interfaces source-class all** command to display the same information.

Related Documentation

- Understanding Source Class Usage and Destination Class Usage Options on page 451
- Route Filter Match Conditions on page 53

CHAPTER 10

Avoiding Traffic Routing Threats with Conditional Routing Policies

Conditional Advertisement and Import Policy (Routing Table) with certain match conditions

BGP accepts all non-looped routes learned from neighbors and imports them into the RIB-In table. If these routes are accepted by the BGP import policy, they are then imported into the inet.0 routing table. In cases where only certain routes are required to be imported, provisions can be made such that the peer routing device exports routes based on a condition or a set of conditions.

The condition for exporting a route can be based on:

- The peer the route was learned from
- The interface the route was learned on
- Some other required attribute

For example:

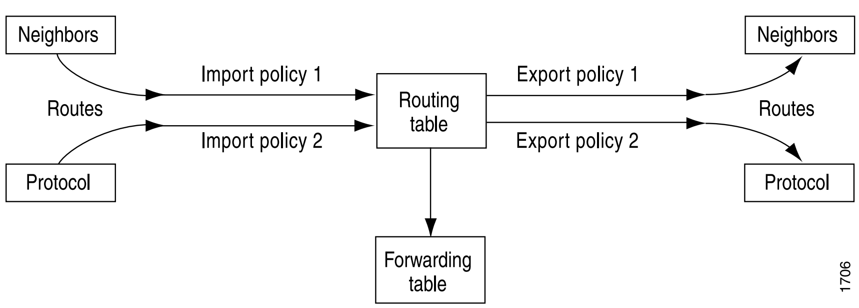
```
[edit]
policy-options {
  condition condition-name {
    if-route-exists address table table-name;
  }
}
```

This is known as conditional installation of prefixes and is described in "Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table" on page 493.

The Juniper Networks® Junos® Operating System (Junos OS) supports conditional export of routes based on the existence of another route in the routing table. Junos OS does not, however, support policy conditions for import policy.

Figure 42 on page 490 illustrates where BGP import and export policies are applied. An import policy is applied to inbound routes that are visible in the output of the **show route receive-protocol bgp *neighbor-address*** command. An export policy is applied to outbound routes that are visible in the output of the **show route advertising-protocol bgp *neighbor-address*** command.

Figure 42: BGP Import and Export Policies



To enable conditional installation of prefixes, an export policy must be configured on the device where the prefix export has to take place. The export policy evaluates each route to verify that it satisfies all the match conditions under the **from** statement. It also searches for the existence of the route defined under the **condition** statement (also configured under the **from** statement).

If the route does not match the entire set of required conditions defined in the policy, or if the route defined under the **condition** statement does not exist in the routing table, the route is not exported to its BGP peers. Thus, a conditional export policy matches the routes for the desired route or prefix you want installed in the peers' routing table.

To configure the conditional installation of prefixes with the help of an export policy:

1. Create a **condition** statement to check prefixes.

```
[edit]
policy-options {
  condition condition-name {
    if-route-exists address table table-name;
  }
}
```

2. Create an export policy with the newly created condition using the **condition** statement.

```
[edit]
policy-options {
  policy-statement policy-name {
    term 1 {
      from {
        protocols bgp;
        condition condition-name;
      }
      then {
        accept;
      }
    }
  }
}
```

3. Apply the export policy to the device that requires only selected prefixes to be exported from the routing table.

```
[edit]
protocols bgp {
  group group-name {
    export policy-name;
  }
}
```

Related Documentation

- Conditional Advertisement Enabling Conditional Installation of Prefixes Use Cases on page 492
- Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table on page 493

Conditional Advertisement Enabling Conditional Installation of Prefixes Use Cases

Networks are usually subdivided into smaller, more-manageable units called autonomous systems (ASs). When BGP is used by routers to form peer relationships in the same AS, it is referred to as internal BGP (IBGP). When BGP is used by routers to form peer relationships in different ASs, it is referred to as external BGP (EBGP).

After performing route sanity checks, a BGP router accepts the routes received from its peers and installs them into the routing table. By default, all routers in IBGP and EBGP sessions follow the standard BGP advertisement rules. While a router in an IBGP session advertises only the routes learned from its direct peers, a router in an EBGP session advertises all routes learned from its direct and indirect peers (peers of peers). Hence, in a typical network configured with EBGP, a router adds all routes received from an EBGP peer into its routing table and advertises nearly all routes to all EBGP peers.

A service provider exchanging BGP routes with both customers and peers on the Internet is at risk of malicious and unintended threats that can compromise the proper routing of traffic, as well as the operation of the routers.

This has several disadvantages:

- **Non-aggregated route advertisements**—A customer could erroneously advertise all its prefixes to the ISP rather than an aggregate of its address space. Given the size of the Internet routing table, this must be carefully controlled. An edge router might also need only a default route out toward the Internet and instead be receiving the entire BGP routing table from its upstream peer.
- **BGP route manipulation**—If a malicious administrator alters the contents of the BGP routing table, it could prevent traffic from reaching its intended destination.
- **BGP route hijacking**—A rogue administrator of a BGP peer could maliciously announce a network’s prefixes in an attempt to reroute the traffic intended for the victim network to the administrator’s network to either gain access to the contents of traffic or to block the victim’s online services.
- **BGP denial of service (DoS)**—If a malicious administrator sends unexpected or undesirable BGP traffic to a router in an attempt to use all of the router’s available BGP resources, it might result in impairing the router’s ability to process valid BGP route information.

Conditional installation of prefixes can be used to address all the problems previously mentioned. If a customer requires access to remote networks, it is possible to install a specific route in the routing table of the router that is connected with the remote network. This does not happen in a typical EBGP network and hence, conditional installation of prefixes becomes essential.

ASs are not only bound by physical relationships but by business or other organizational relationships. An AS can provide services to another organization, or act as a transit AS between two other ASs. These transit ASs are bound by contractual agreements between the parties that include parameters on how to connect to each other and most importantly, the type and quantity of traffic they carry for each other. Therefore, for both legal and financial reasons, service providers must implement policies that control how BGP routes are exchanged with neighbors, which routes are accepted from those neighbors, and how those routes affect the traffic between the ASs.

There are many different options available to filter routes received from a BGP peer to both enforce inter-AS policies and mitigate the risks of receiving potentially harmful routes. Conventional route filtering examines the attributes of a route and accepts or rejects the route based on such attributes. A policy or filter can examine the contents of the AS-Path, the next-hop value, a community value, a list of prefixes, the address family of the route, and so on.

In some cases, the standard “acceptance condition” of matching a particular attribute value is not enough. The service provider might need to use another condition outside of the route itself, for example, another route in the routing table. As an example, it might be desirable to install a default route received from an upstream peer, only if it can be verified that this peer has reachability to other networks further upstream. This conditional route installation avoids installing a default route that is used to send traffic toward this peer, when the peer might have lost its routes upstream, leading to black-holed traffic. To achieve this, the router can be configured to search for the presence of a particular route in the routing table, and based on this knowledge accept or reject another prefix.

“Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table” on page 493 explains how the conditional installation of prefixes can be configured and verified.

Related Documentation

- Conditional Advertisement and Import Policy (Routing Table) with certain match conditions on page 490
- Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table on page 493

Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table

This example shows how to configure conditional installation of prefixes in a routing table using BGP export policy.

Requirements

This example uses the following hardware and software components:

- M Series Multiservice Edge Routers, MX Series 3D Universal Edge Routers, or T Series Core Routers
- Junos OS Release 9.0 or later

Overview

In this example, three routers in three different autonomous systems (ASs) are connected and configured with the BGP protocol. Router Internet, which is the upstream router, has five addresses configured on its lo0.0 loopback interface (172.16.11.1/32, 172.16.12.1/32, 172.16.13.1, 172.16.14.1/32, and 172.16.15.1/32), and an extra loopback address (192.168.9.1/32) to be configured as the router ID. These six addresses are exported into BGP to emulate the contents of a BGP routing table of a router connected to the Internet, and advertised to Router North.

Router North exports a default route into BGP, and advertises the default route and the five BGP routes to Router South, which is the downstream router. Router South receives the default route and only one other route (172.16.11.1/32), and installs this route and the default route in its routing table.

To summarize, the example meets the following requirements:

- On Device North, send 0/0 to Device South only if a particular route is also sent (in the example 172.16.11.1/32).
- On Device South, accept the default route and the 172.16.11.1/32 route. Drop all other routes. Consider that Device South might be receiving the entire Internet table, while the operator only wants Device South to have the default and one other specific prefix.

The first requirement is met with an export policy on Device North:

```
user@North# show policy-options
policy-statement conditional-export-bgp {
  term prefix_11 {
    from {
      protocol bgp;
      route-filter 10.11.0.0/5 orlonger;
    }
    then accept;
  }
  term conditional-default {
    from {
      route-filter 0.0.0.0/0 exact;
      condition prefix_11;
    }
    then accept;
  }
  term others {
    then reject;
  }
}
condition prefix_11 {
  if-route-exists {
    172.16.11.1/32;
    table inet.0;
  }
}
```


The logic of the conditional export policy can be summarized as follows: If 0/0 is present, and if 172.16.11.1/32 is present, then send the 0/0 prefix. This implies that if 172.16.11.1/32 is not present, then do not send 0/0.

The second requirement is met with an import policy on Device South:

```
user@South# show policy-options
policy-statement import-selected-routes {
  term 1 {
    from {
      rib inet.0;
      neighbor 10.0.78.14;
      route-filter 0.0.0.0/0 exact;
      route-filter 10.11.0.0/8 orlonger;
    }
    then accept;
  }
  term 2 {
    then reject;
  }
}
```

In this example, four routes are dropped as a result of the import policy on Device South. This is because the export policy on Device North leaks all of the routes received from Device Internet, and the import policy on Device South excludes some of these routes.

It is important to understand that in Junos OS, although an import policy (inbound route filter) might reject a route, not use it for traffic forwarding, and not include it in an advertisement to other peers, the router retains these routes as hidden routes. These hidden routes are not available for policy or routing purposes. However, they do occupy memory space on the router. A service provider filtering routes to control the amount of information being kept in memory and processed by a router might want the router to entirely drop the routes being rejected by the import policy.

Hidden routes can be viewed by using the **show route receive-protocol bgp** ***neighbor-address*** **hidden** command. The hidden routes can then be retained or dropped from the routing table by configuring the **keep all | none** statement at the **[edit protocols bgp]** or **[edit protocols bgp group** ***group-name*****]** hierarchy level.

The rules of BGP route retention are as follows:

- By default, all routes learned from BGP are retained, except those where the AS path is looped. (The AS path includes the local AS.)
- By configuring the **keep all** statement, all routes learned from BGP are retained, even those with the local AS in the AS path.
- By configuring the **keep none** statement, BGP discards routes that were received from a peer and that were rejected by import policy or other sanity checking. When this statement is configured and the inbound policy changes, Junos OS re-advertises all the routes advertised by the peer.

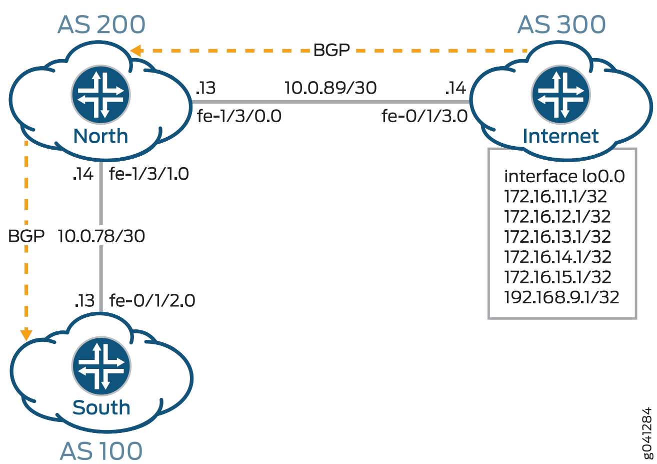
When you configure **keep all** or **keep none** and the peers support route refresh, the local speaker sends a refresh message and performs an import evaluation. For these peers, the sessions do not restart. To determine if a peer supports refresh, check for **Peer supports Refresh capability** in the output of the **show bgp neighbor** command.

CAUTION: If you configure **keep all** or **keep none** and the peer does not support session restart, the associated BGP sessions are restarted (flapped).

Topology

Figure 43 on page 496 shows the topology used in this example.

Figure 43: Conditional Installation of Prefixes



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Router Internet

```
set interfaces lo0 unit 0 family inet address 172.16.11.1/32
set interfaces lo0 unit 0 family inet address 172.16.12.1/32
set interfaces lo0 unit 0 family inet address 172.16.13.1/32
set interfaces lo0 unit 0 family inet address 172.16.14.1/32
set interfaces lo0 unit 0 family inet address 172.16.15.1/32
set interfaces lo0 unit 0 family inet address 192.168.9.1/32
set interfaces fe-0/1/3 unit 0 family inet address 10.0.89.14/30
set protocols bgp group toNorth local-address 10.0.89.14
set protocols bgp group toNorth peer-as 200
set protocols bgp group toNorth neighbor 10.0.89.13
set protocols bgp group toNorth export into-bgp
set policy-options policy-statement into-bgp term 1 from interface lo0.0
set policy-options policy-statement into-bgp term 1 then accept
set routing-options router-id 192.168.9.1
set routing-options autonomous-system 300
```

Router North

```
set interfaces fe-1/3/1 unit 0 family inet address 10.0.78.14/30
set interfaces fe-1/3/0 unit 0 family inet address 10.0.89.13/30
set interfaces lo0 unit 0 family inet address 192.168.8.1/32
set protocols bgp group toInternet local-address 10.0.89.13
set protocols bgp group toInternet peer-as 300
set protocols bgp group toInternet neighbor 10.0.89.14
set protocols bgp group toSouth local-address 10.0.78.14
set protocols bgp group toSouth export conditional-export-bgp
set protocols bgp group toSouth peer-as 100
set protocols bgp group toSouth neighbor 10.0.78.13
set policy-options policy-statement conditional-export-bgp term prefix_11 from protocol
  bgp
set policy-options policy-statement conditional-export-bgp term prefix_11 from route-filter
  10.11.0.0/5 orlonger
set policy-options policy-statement conditional-export-bgp term prefix_11 then accept
set policy-options policy-statement conditional-export-bgp term conditional-default
  from route-filter 0.0.0.0/0 exact
set policy-options policy-statement conditional-export-bgp term conditional-default
  from condition prefix_11
set policy-options policy-statement conditional-export-bgp term conditional-default
  then accept
set policy-options policy-statement conditional-export-bgp term others then reject
set policy-options condition prefix_11 if-route-exists 172.16.11.1/32
set policy-options condition prefix_11 if-route-exists table inet.0
set routing-options static route 0/0 reject
set routing-options router-id 192.168.8.1
set routing-options autonomous-system 200
```

Router South

```
set interfaces fe-0/1/2 unit 0 family inet address 10.0.78.13/30
set interfaces lo0 unit 0 family inet address 192.168.7.1/32
set protocols bgp group toNorth local-address 10.0.78.13
set protocols bgp group toNorth import import-selected-routes
set protocols bgp group toNorth peer-as 200
set protocols bgp group toNorth neighbor 10.0.78.14
set policy-options policy-statement import-selected-routes term 1 from neighbor 10.0.78.14
set policy-options policy-statement import-selected-routes term 1 from route-filter
  10.11.0.0/8 orlonger
set policy-options policy-statement import-selected-routes term 1 from route-filter
  0.0.0.0/0 exact
set policy-options policy-statement import-selected-routes term 1 then accept
set policy-options policy-statement import-selected-routes term 2 then reject
set routing-options router-id 192.168.7.1
set routing-options autonomous-system 100
```

Configuring Conditional Installation of Prefixes

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure conditional installation of prefixes:

1. Configure the router interfaces forming the links between the three routers.

Router Internet
[edit interfaces]
user@Internet# **set fe-0/1/3 unit 0 family inet address 10.0.89.14/30**

Router North
[edit interfaces]
user@North# **set fe-1/3/1 unit 0 family inet address 10.0.78.14/30**
user@North# **set fe-1/3/0 unit 0 family inet address 10.0.89.13/30**

Router South
[edit interfaces]
user@South# **set fe-0/1/2 unit 0 family inet address 10.0.78.13/30**

2. Configure five loopback interface addresses on Router Internet to emulate BGP routes learned from the Internet that are to be imported into the routing table of Router South, and configure an additional address (192.168.9.1/32) that will be configured as the router ID.

   Router Internet
   [edit interfaces lo0 unit 0 family inet]
   user@Internet# **set address 172.16.11.1/32**
   user@Internet# **set address 172.16.12.1/32**
   user@Internet# **set address 172.16.13.1/32**
   user@Internet# **set address 172.16.14.1/32**
   user@Internet# **set address 172.16.15.1/32**
   user@Internet# **set address 192.168.9.1/32**

   Also, configure the loopback interface addresses on Routers North and South.

   Router North
   [edit interfaces lo0 unit 0 family inet]
   user@North# **set address 192.168.8.1/32**

   Router South
   [edit interfaces lo0 unit 0 family inet]
   user@South# **set address 192.168.7.1/32**

3. Configure the static default route on Router North to be advertised to Router South.

   [edit routing-options]
   user@North# **set static route 0/0 reject**

4. Define the condition for exporting prefixes from the routing table on Router North.

   [edit policy-options condition prefix_11]
   user@North# **set if-route-exists 172.16.11.1/32**
   user@North# **set if-route-exists table inet.0**

5. Define export policies (**into-bgp** and **conditional-export-bgp**) on Routers Internet and North respectively, to advertise routes to BGP.

   NOTE: Ensure that you reference the condition, **prefix_11** (configured in Step 4), in the export policy.

Router Internet
[edit policy-options policy-statement into-bgp]
user@Internet# **set term 1 from interface lo0.0**
user@Internet# **set term 1 then accept**

Router North
[edit policy-options policy-statement conditional-export-bgp]
user@North# **set term prefix_11 from protocol bgp**
user@North# **set term prefix_11 from route-filter 10.11.0.0/5 orlonger**
user@North# **set term prefix_11 then accept**
user@North# **set term conditional-default from route-filter 0.0.0.0/0 exact**
user@North# **set term conditional-default from condition prefix_11**
user@North# **set term conditional-default then accept**
user@North# **set term others then reject**

6. Define an import policy (**import-selected-routes**) on Router South to import some of the routes advertised by Router North into its routing table.

   [edit policy-options policy-statement import-selected-routes]
   user@South# **set term 1 from neighbor 10.0.78.14**
   user@South# **set term 1 from route-filter 10.11.0.0/8 orlonger**
   user@South# **set term 1 from route-filter 0.0.0.0/0 exact**
   user@South# **set term 1 then accept**
   user@South# **set term 2 then reject**

7. Configure BGP on all three routers to enable the flow of prefixes between the autonomous systems.

   NOTE: Ensure that you apply the defined import and export policies to the respective BGP groups for prefix advertisement to take place.

Router Internet
[edit protocols bgp group toNorth]
user@Internet# **set local-address 10.0.89.14**
user@Internet# **set peer-as 200**
user@Internet# **set neighbor 10.0.89.13**
user@Internet# **set export into-bgp**

Router North
[edit protocols bgp group toInternet]
user@North# **set local-address 10.0.89.13**
user@North# **set peer-as 300**
user@North# **set neighbor 10.0.89.14**

[edit protocols bgp group toSouth]
user@North# **set local-address 10.0.78.14**
user@North# **set peer-as 100**
user@North# **set neighbor 10.0.78.13**
user@North# **set export conditional-export-bgp**

Router South
[edit protocols bgp group toNorth]
user@South# **set local-address 10.0.78.13**
user@South# **set peer-as 200**

user@South# **set neighbor 10.0.78.14**
user@South# **set import import-selected-routes**

8. Configure the router ID and autonomous system number for all three routers.

NOTE: In this example, the router ID is configured based on the IP address configured on the lo0.0 interface of the router.

Router Internet
[edit routing options]
user@Internet# **set router-id 192.168.9.1**
user@Internet# **set autonomous-system 300**

Router North
[edit routing options]
user@North# **set router-id 192.168.8.1**
user@North# **set autonomous-system 200**

Router South
[edit routing options]
user@South# **set router-id 192.168.7.1**
user@South# **set autonomous-system 100**

Results

From configuration mode, confirm your configuration by issuing the **show interfaces**, **show protocols bgp**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device Internet

```
user@Internet# show interfaces
fe-0/1/3 {
  unit 0 {
    family inet {
      address 10.0.89.14/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      address 172.16.11.1/32;
      address 172.16.12.1/32;
      address 172.16.13.1/32;
      address 172.16.14.1/32;
      address 172.16.15.1/32;
      address 192.168.9.1/32;
    }
  }
}
```



```
user@Internet# show protocols bgp
group toNorth {
   local-address 10.0.89.14;
   export into-bgp;
   peer-as 200;
   neighbor 10.0.89.13;
}
```

```
user@Internet# show policy-options
policy-statement into-bgp {
   term 1 {
      from interface lo0.3;
      then accept;
   }
}
```

```
user@Internet# show routing-options
router-id 192.168.9.1;
autonomous-system 300;
```

Device North

```
user@North# show interfaces
fe-1/3/1 {
   unit 0 {
      family inet {
         address 10.0.78.14/30;
      }
   }
}
fe-1/3/0 {
   unit 0 {
      family inet {
         address 10.0.89.13/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.8.1/32;
      }
   }
}
```

```
user@North# show protocols bgp
group toInternet {
   local-address 10.0.89.13;
   peer-as 300;
   neighbor 10.0.89.14;
}
group toSouth {
   local-address 10.0.78.14;
   export conditional-export-bgp;
   peer-as 100;
   neighbor 10.0.78.13;
}
```

```
user@North# show policy-options
```

```
policy-statement conditional-export-bgp {
   term prefix_11 {
      from {
         protocol bgp;
         route-filter 10.11.0.0/5 orlonger;
      }
      then accept;
   }
   term conditional-default {
      from {
         route-filter 0.0.0.0/0 exact;
         condition prefix_11;
      }
      then accept;
   }
   term others {
      then reject;
   }
}
condition prefix_11 {
   if-route-exists {
      172.16.11.1/32;
      table inet.0;
   }
}
```

```
user@North# show routing-options
static {
   route 0.0.0.0/0 reject;
}
router-id 192.168.8.1;
autonomous-system 200;
```

Device South

```
user@South# show interfaces
fe-0/1/2 {
   unit 0 {
      family inet {
         address 10.0.78.13/30;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.7.1/32;
      }
   }
}
```

```
user@South# show protocols bgp
bgp {
   group toNorth {
      local-address 10.0.78.13;
      import import-selected-routes;
      peer-as 200;
      neighbor 10.0.78.14;
```



```
    }
}
user@South# show policy-options
policy-statement import-selected-routes {
    term 1 {
        from {
            neighbor 10.0.78.14;
            route-filter 10.11.0.0/8 orlonger;
            route-filter 0.0.0.0/0 exact;
        }
        then accept;
    }
    term 2 {
        then reject;
    }
}
user@South# show routing-options
router-id 192.168.7.1;
autonomous-system 100;
```

If you are done configuring the routers, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying BGP

Purpose Verify that BGP sessions have been established between the three routers.

Action From operational mode, run the **show bgp neighbor *neighbor-address*** command.

1. Check the BGP session on Router Internet to verify that Router North is a neighbor.

```
user@Internet> show bgp neighbor 10.0.89.13
Peer: 10.0.89.13+179 AS 200   Local: 10.0.89.14+56187 AS 300
  Type: External    State: Established    Flags: [ImportEval Sync]
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Export: [ into-bgp ]
  Options: [Preference LocalAddress PeerAS Refresh]
  Local Address: 10.0.89.14 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.168.8.1     Local ID: 192.168.9.1       Active Holdtime: 90
  Keepalive Interval: 30         Group index: 0    Peer index: 0
```



```
  BFD: disabled, down
  Local Interface: fe-0/1/3.0
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 200)
  Peer does not support Addpath
  Table inet.0 Bit: 10000
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              0
    Received prefixes:            0
    Accepted prefixes:            0
    Suppressed due to damping:    0
    Advertised prefixes:          6
  Last traffic (seconds): Received 9    Sent 18   Checked 28
  Input messages:  Total 12     Updates 1       Refreshes 0     Octets 232
  Output messages: Total 14     Updates 1       Refreshes 0     Octets 383
  Output Queue[0]: 0
```

2. Check the BGP session on Router North to verify that Router Internet is a neighbor.

```
user@North> show bgp neighbor 10.0.89.14
Peer: 10.0.89.14+56187 AS 300  Local: 10.0.89.13+179 AS 200
  Type: External    State: Established    Flags: [ImportEval Sync]
  Last State: OpenConfirm   Last Event: RecvKeepAlive
  Last Error: None
  Options: [Preference LocalAddress PeerAS Refresh]
  Local Address: 10.0.89.13 Holdtime: 90 Preference: 170
  Number of flaps: 0
  Peer ID: 192.168.9.1     Local ID: 192.168.8.1       Active Holdtime: 90
  Keepalive Interval: 30         Group index: 0    Peer index: 0
  BFD: disabled, down
  Local Interface: fe-1/3/0.0
  NLRI for restart configured on peer: inet-unicast
  NLRI advertised by peer: inet-unicast
  NLRI for this session: inet-unicast
  Peer supports Refresh capability (2)
  Stale routes from peer are kept for: 300
  Peer does not support Restarter functionality
  NLRI that restart is negotiated for: inet-unicast
  NLRI of received end-of-rib markers: inet-unicast
  NLRI of all end-of-rib markers sent: inet-unicast
  Peer supports 4 byte AS extension (peer-as 300)
  Peer does not support Addpath
  Table inet.0 Bit: 10001
    RIB State: BGP restart is complete
    Send state: in sync
    Active prefixes:              6
    Received prefixes:            6
    Accepted prefixes:            6
    Suppressed due to damping:    0
    Advertised prefixes:          0
  Last traffic (seconds): Received 14   Sent 3    Checked 3
  Input messages:  Total 16     Updates 2       Refreshes 0     Octets 402
```



```
Output messages: Total 15     Updates 0       Refreshes 0     Octets 348
Output Queue[0]: 0
```

Check the following fields in these outputs to verify that BGP sessions have been established:

- **Peer**—Check if the peer AS number is listed.
- **Local**—Check if the local AS number is listed.
- **State**—Ensure that the value is **Established**. If not, check the configuration again and see **show bgp neighbor** for more details on the output fields.

Similarly, verify that Routers North and South form peer relationships with each other.

Meaning BGP sessions are established between the three routers.

Verifying Prefix Advertisement from Router Internet to Router North

Purpose Verify that the routes sent from Router Internet are received by Router North.

Action

1. From operational mode on Router Internet, run the **show route advertising-protocol bgp *neighbor-address*** command.

```
user@Internet> show route advertising-protocol bgp 10.0.89.13
inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.11.1/32             Self                                    I
* 172.16.12.1/32             Self                                    I
* 172.16.13.1/32             Self                                    I
* 172.16.14.1/32             Self                                    I
* 172.16.15.1/32             Self                                    I
* 192.168.9.1/32          Self                                    I
```

   The output verifies that Router Internet advertises the routes 172.16.11.1/32, 172.16.12.1/32, 172.16.13.1/32, 172.16.14.1/32, 172.16.15.1/32, and 192.168.9.1/32 (the loopback address used as router ID) to Router North.

2. From operational mode on Router North, run the **show route receive-protocol bgp *neighbor-address*** command.

```
user@North> show route receive-protocol bgp 10.0.89.14
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.11.1/32             10.0.89.14                              300 I
* 172.16.12.1/32             10.0.89.14                              300 I
* 172.16.13.1/32             10.0.89.14                              300 I
* 172.16.14.1/32             10.0.89.14                              300 I
* 172.16.15.1/32             10.0.89.14                              300 I
* 192.168.9.1/32          10.0.89.14                              300 I
```

   The output verifies that Router North has received all the routes advertised by Router Internet.

Meaning

Prefixes sent by Router Internet have been successfully installed into the routing table on Router North.

Verifying Prefix Advertisement from Router North to Router South

Purpose

Verify that the routes received from Router Internet and the static default route are advertised by Router North to Router South.

Action

1. From operational mode on Router North, run the **show route 0/0 exact** command.

```
user@North> show route 0/0 exact
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 00:10:22
                      Reject
```

   The output verifies the presence of the static default route (0.0.0.0/0) in the routing table on Router North.

2. From operational mode on Router North, run the **show route advertising-protocol bgp** ***neighbor-address*** command.

```
user@North> show route advertising-protocol bgp 10.0.78.13
inet.0: 12 destinations, 12 routes (12 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 0.0.0.0/0               Self                                    I
* 172.16.11.1/32             Self                                    300 I
* 172.16.12.1/32             Self                                    300 I
* 172.16.13.1/32             Self                                    300 I
* 172.16.14.1/32             Self                                    300 I
* 172.16.15.1/32             Self                                    300 I
```

The output verifies that Router North is advertising the static route and the 172.16.11.1/32 route received from Router Internet, as well as many other routes, to Router South.

Verifying BGP Import Policy for Installation of Prefixes

Purpose Verify that the BGP import policy successfully installs the required prefixes.

Action See if the import policy on Router South is operational by checking if only the static default route from Router North and the 172.16.11.1/32 route from Router South are installed in the routing table.

From operational mode, run the **show route receive-protocol bgp** ***neighbor-address*** command.

```
user@South> show route receive-protocol bgp 10.0.78.14
inet.0: 10 destinations, 11 routes (6 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 0.0.0.0/0               10.0.78.14                              200 I
* 172.16.11.1/32             10.0.78.14                              200 300 I
```

The output verifies that the BGP import policy is operational on Router South, and only the static default route of 0.0.0.0/0 from Router North and the 172.16.11.1/32 route from Router Internet have leaked into the routing table on Router South.

Meaning The installation of prefixes is successful because of the configured BGP import policy.

Verifying Conditional Export from Router North to Router South

Purpose Verify that when Device Internet stops sending the 172.16.11.1/32 route, Device North stops sending the default 0/0 route.

Action

1. Cause Device Internet to stop sending the 172.16.11.1/32 route by deactivating the 172.16.11.1/32 address on the loopback interface.

```
[edit interfaces lo0 unit 0 family inet]
user@Internet# deactivate address 172.16.11.1/32
user@Internet# commit
```

2. From operational mode on Router North, run the **show route advertising-protocol bgp** ***neighbor-address*** command.

```
user@North> show route advertising-protocol bgp 10.0.78.13
inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.12.1/32             Self                                    300 I
* 172.16.13.1/32             Self                                    300 I
* 172.16.14.1/32             Self                                    300 I
* 172.16.15.1/32             Self                                    300 I
```

   The output verifies that Router North is not advertising the default route to Router South. This is the expected behavior when the 172.16.11.1/32 route is not present.

3. Reactivate the 172.16.11.1/32 address on Device Internet's loopback interface.

```
[edit interfaces lo0 unit 0 family inet]
user@Internet# activate address 172.16.11.1/32
user@Internet# commit
```

Verifying the Presence of Routes Hidden by Policy (Optional)

Purpose Verify the presence of routes hidden by the import policy configured on Router South.

NOTE: This section demonstrates the effects of various changes you can make to the configuration depending on your needs.

Action View routes hidden from the routing table of Router South by:

- Using the **hidden** option for the **show route receive-protocol bgp** ***neighbor-address*** command.
- Deactivating the import policy.

1. From operational mode, run the **show route receive-protocol bgp *neighbor-address* hidden** command to view hidden routes.

```
user@South> show route receive-protocol bgp 10.0.78.14 hidden
inet.0: 10 destinations, 11 routes (6 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.12.1/32             10.0.78.14                               200 300
I
  172.16.13.1/32             10.0.78.14                               200 300
I
  172.16.14.1/32             10.0.78.14                               200 300
I
  172.16.15.1/32             10.0.78.14                               200 300
I
```

   The output verifies the presence of routes hidden by the import policy (172.16.12.1/32, 172.16.13.1/32, 172.16.14.1/32, and 172.16.15.1/32) on Router South.

2. Deactivate the BGP import policy by configuring the **deactivate import** statement at the **[edit protocols bgp group *group-name*]** hierarchy level.

```
[edit protocols bgp group toNorth]
user@South# deactivate import
user@South# commit
```

3. Run the **show route receive-protocol bgp *neighbor-address*** operational mode command to check the routes after deactivating the import policy.

```
user@South> show route receive-protocol bgp 10.0.78.14
inet.0: 10 destinations, 11 routes (10 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 0.0.0.0/0               10.0.78.14                              200 I
* 172.16.11.1/32             10.0.78.14                               200 300
I
* 172.16.12.1/32             10.0.78.14                               200 300
I
* 172.16.13.1/32             10.0.78.14                               200 300
I
* 172.16.14.1/32             10.0.78.14                               200 300
I
* 172.16.15.1/32             10.0.78.14                               200 300
I
```

   The output verifies the presence of previously hidden routes (172.16.12.1/32, 172.16.13.1/32, 172.16.14.1/32, and 172.16.15.1/32).

4. Activate the BGP import policy and remove the hidden routes from the routing table by configuring the **activate import** and **keep none** statements respectively at the **[edit protocols bgp group *group-name*]** hierarchy level.

```
[edit protocols bgp group toNorth]
user@South# activate import
user@South# set keep none
```

user@South# **commit**

5. From operational mode, run the **show route receive-protocol bgp** ***neighbor-address*** **hidden** command to check the routes after activating the import policy and configuring the **keep none** statement.

   ```
   user@South> show route receive-protocol bgp 10.0.78.14 hidden
   ```

   ```
   inet.0: 6 destinations, 7 routes (6 active, 0 holddown, 0 hidden)
   ```

   The output verifies that the hidden routes are not maintained in the routing table because of the configured **keep none** statement.

Related Documentation

- Conditional Advertisement Enabling Conditional Installation of Prefixes Use Cases on page 492
- Conditional Advertisement and Import Policy (Routing Table) with certain match conditions on page 490

CHAPTER 11

Protecting Against DoS Attacks by Forwarding Traffic to the Discard Interface

Understanding Forwarding Packets to the Discard Interface

The discard (dsc) interface is not a physical interface, but a virtual interface that discards packets. You can configure one discard interface. This interface allows you to identify the ingress point of a denial-of-service (DoS) attack. When your network is under attack, the target host IP address is identified, and the local policy forwards attacking packets to the discard interface. Traffic routed out of the discard interface is silently discarded.

The discard interface allows you to protect a network from DoS attacks by identifying the target IP address that is being attacked and configuring a policy to forward all packets to a discard interface. All packets forwarded to the discard interface are dropped.

To configure the discard interface, include the **dsc** statement:

```
[edit interfaces interface-name]
dsc {
  unit 0 {
    family inet {
      filter {
        input filter-name;
        output filter-name;
      }
    }
  }
}
```

The **dsc** interface name denotes the discard interface. The discard interface supports only unit 0.

The following two configurations are required to configure a policy to forward all packets to the discard interface.

Configure an input policy to associate a community with the discard interface:

```
[edit]
```

```
policy-options {
  community community-name members [ community-id ];
  policy-statement statement-name {
    term term-name {
      from community community-name;
      then {
        next-hop address; # Remote end of the point-to-point interface
        accept;
      }
    }
  }
}
```

Configure an output policy to set up the community on the routes injected into the network:

```
[edit]
policy-options {
  policy-statement statement-name {
    term term-name {
      from prefix-list name;
      then community (set | add | delete) community-name;
    }
  }
}
```

Related Documentation

- Example: Forwarding Packets to the Discard Interface on page 512

Example: Forwarding Packets to the Discard Interface

This example shows how to use discard routing to mitigate denial of service (DoS) attacks, protect vital network resources from outside attack, provide protection services for customers so that each customer can initiate its own protection, and log and track DoS attempts.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In discard routing, routers are configured with rules that disallow millions of requests in a short period of time from being sent to the same address. If too many requests are received in a short period of time, the router simply discards the requests without forwarding them. The requests are sent to a router that does not forward the packets. The problematic routes are sometimes referred to as discard routes or black-holed routes. The types of routes that should be discarded are identified as attacks to customers from peers or other customers, attacks from customers to peers or other customers, attack controllers, which are hosts providing attack instructions, and unallocated address spaces, known as bogons or invalid IP addresses.

After the attack attempt is identified, operators can put a configuration in place to mitigate the attack. One way to configure discard routing in Junos OS is to create a discard static route for each next hop used for discard routes. A discard static route uses the **discard** option.

For example:

```
user@host# show routing-options
static {
  route 192.0.2.101/32 discard;
  route 192.0.2.103/32 discard;
  route 192.0.2.105/32 discard;
}
```

```
user@host>  show route protocol static terse
inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 192.0.2.101/32     S   5                        Discard
* ? 192.0.2.103/32     S   5                        Discard
* ? 192.0.2.105/32     S   5                        Discard
```

Another strategy, which is the main focus of this example, is to use routing policy and the discard interface. In this approach, the discard interface contains the next hop you are assigning to the black-hole routes. A discard interface can have only one logical unit (unit 0), but you can configure multiple IP addresses on unit 0.

For example:

```
user@host# show interfaces dsc
unit 0 {
  family inet {
    address 192.0.2.102/32 {
      destination 192.0.2.101;
    }
    address 192.0.2.104/32 {
      destination 192.0.2.103;
    }
    address 192.0.2.106/32 {
      destination 192.0.2.105;
    }
  }
}
```

```
user@host>  show interfaces terse dsc
b
Interface               Admin Link Proto    Local                 Remote
dsc                     up    up
dsc.0                   up    up   inet     192.0.2.102         --> 192.0.2.101
                                            192.0.2.104         --> 192.0.2.103
                                            192.0.2.106         --> 192.0.2.105
```

The advantage of using a discard interface instead of using discard static routes is that the discard interface allows you to configure and assign filters to the interface for counting, logging, and sampling the traffic. This is demonstrated in this example.

To actually discard packets requires a routing policy attached to the BGP sessions. To locate discard-eligible routes, you can use a route filter, an access list, or a BGP community value.

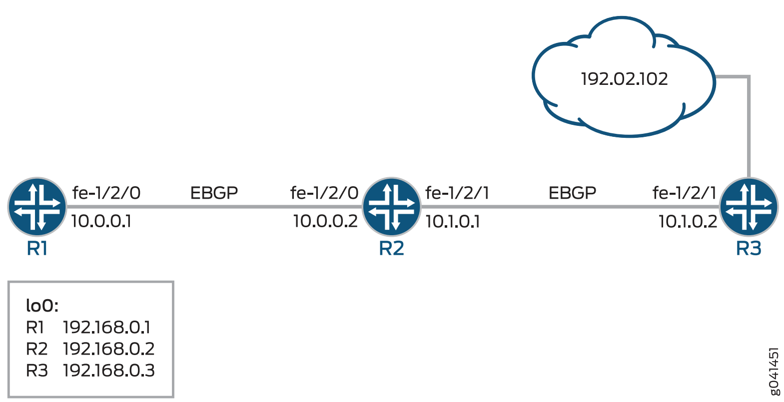
For example, here is how you would use a route filter:

Route Filter

```
protocols {
  bgp {
    import blackhole-by-route;
  }
}
policy-options {
  policy-statement blackhole-by-route {
    term specific-routes {
      from {
        route-filter 10.10.10.1/32 exact;
        route-filter 10.20.20.2/32 exact;
        route-filter 10.30.30.3/32 exact;
        route-filter 10.40.40.4/32 exact;
      }
      then {
        next-hop 192.0.2.101
      }
    }
  }
}
```

Figure 44 on page 514 shows the sample network.

Figure 44: Discard Interface Sample Network



The example includes three routers with external BGP (EBGP) sessions established.

Device R1 represents the attacking device. Device R3 represents the router closest to the device that is being attacked. Device R2 mitigates the attack by forwarding packets to the discard interface.

The example shows an outbound filter applied to the discard interface.

NOTE: An issue with using a single black-hole filter is visibility. All discard packets increment the same counter. To see which categories of packets are being discarded, use destination class usage (DCU), and associate a user-defined class with each black-hole community. Then reference the DCU classes in a firewall filter. For related examples, see "Example: Grouping Source and Destination Prefixes into a Forwarding Class" on page 479 and "Example: Configuring a Rate-Limiting Filter Based on Destination Class" on page 783.

Compared to using route filters and access lists, using a community value is the least administratively difficult and the most scalable approach. Therefore, this is the approach shown in this example.

By default, the next hop must be equal the external BGP (EBGP) peer address. Altering the next hop for black-hole services requires the multihop feature to be configured on the EBGP sessions.

"CLI Quick Configuration" on page 515 shows the configuration for all of the devices in Figure 44 on page 514.

The section "Step-by-Step Procedure" on page 516 describes the steps on Device R2.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.1/30
set interfaces dsc unit 0 family inet filter output log-discard
set interfaces dsc unit 0 family inet address 192.0.2.102/32 destination 192.0.2.101
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp import blackhole-policy
set protocols bgp group ext type external
set protocols bgp group ext multihop
set protocols bgp group ext export dsc-export
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set protocols bgp group ext neighbor 10.1.0.2 peer-as 300
set policy-options policy-statement blackhole-policy term blackhole-communities from
  community blackhole-all-routers
```

```
set policy-options policy-statement blackhole-policy term blackhole-communities then
  next-hop 192.0.2.101
set policy-options policy-statement dsc-export from route-filter 192.0.2.101/32 exact
set policy-options policy-statement dsc-export from route-filter 192.0.2.102/32 exact
set policy-options policy-statement dsc-export then community set blackhole-all-routers
set policy-options policy-statement dsc-export then accept
set policy-options community blackhole-all-routers members 100:5555
set routing-options static route 192.0.2.102/32 next-hop 192.0.2.101
set routing-options autonomous-system 200
set firewall filter log-discard term one then count counter
set firewall filter log-discard term one then log
```

Device R3

```
set interfaces fe-1/2/1 unit 0 family inet address 10.1.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set interfaces lo0 unit 0 family inet address 192.0.2.102/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.1.0.1
set routing-options autonomous-system 300
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Create the router interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

user@R2# set fe-1/2/1 unit 0 family inet address 10.1.0.1/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure a firewall filter that matches all packets and counts and logs the packets.

```
[edit firewall filter log-discard term one]
user@R2# set then count counter
user@R2# set then log
```

3. Create a discard interface and apply the output firewall filter.

   Input firewall filters have no impact in this context.

```
[edit interfaces dsc unit 0 family inet]
user@R2# set filter output log-discard
user@R2# set address 192.0.2.102/32 destination 192.0.2.101
```

4. Configure a static route that sends the next hop to the destination address that is specified in the discard interface.

```
[edit routing-options static]
```



```
user@R2# set route 192.0.2.102/32 next-hop 192.0.2.101
```

5. Configure BGP peering.

```
[edit protocols bgp ]
user@R2# set group ext type external
user@R2# set group ext multihop
user@R2# set group ext neighbor 10.0.0.1 peer-as 100
user@R2# set group ext neighbor 10.1.0.2 peer-as 300
```

6. Configure the routing policies.

```
[edit policy-options policy-statement blackhole-policy term blackhole-communities]
user@R2# set from community blackhole-all-routers
user@R2# set then next-hop 192.0.2.101

[edit policy-options policy-statement dsc-export]
user@R2# set from route-filter 192.0.2.101/32 exact
user@R2# set from route-filter 192.0.2.102/32 exact
user@R2# set then community set blackhole-all-routers
user@R2# set then accept

[edit policy-options community blackhole-all-routers]
user@R2# set members 100:5555
```

7. Apply the routing policies.

```
[edit protocols bgp ]
user@R2# set import blackhole-policy
user@R2# set group ext export dsc-export
```

8. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R2# set autonomous-system 200
```

Results From configuration mode, confirm your configuration by issuing the **show interfaces**, **show protocols** , **show policy-options**, **show routing-options**, and **show firewall** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@R2# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      address 10.0.0.2/30;
    }
  }
}
fe-1/2/1 {
```

```
    unit 0 {
        family inet {
            address 10.1.0.1/30;
        }
    }
}
dsc {
    unit 0 {
        family inet {
            filter {
                output log-discard;
            }
            address 192.0.2.102/32 {
                destination 192.0.2.101;
            }
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@R2# show protocols
bgp {
    import blackhole-policy;
    group ext {
        type external;
        multihop;
        export dsc-export;
        neighbor 10.0.0.1 {
            peer-as 100;
        }
        neighbor 10.1.0.2 {
            peer-as 300;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement blackhole-policy {
    term blackhole-communities {
        from community blackhole-all-routers;
        then {
            next-hop 192.0.2.101;
        }
    }
}
policy-statement dsc-export {
    from {
        route-filter 192.0.2.101/32 exact;
        route-filter 192.0.2.102/32 exact;
    }
    then {
```



```
        community set blackhole-all-routers;
        accept;
    }
}
community blackhole-all-routers members 100:5555;
```

```
user@R2# show routing-options
static {
    route 192.0.2.102/32 next-hop 192.0.2.101;
}
autonomous-system 200;
```

```
user@R2# show firewall
filter log-discard {
    term one {
        then {
            count counter;
            log;
        }
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Clearing the Firewall Counters on page 519
- Pinging the 192.0.2.101 Address on page 520
- Checking the Output Filter on page 520
- Checking the Community Attribute on page 520

Clearing the Firewall Counters

Purpose Clear the counters to make sure you are starting from a known zero (0) state.

Action

1. From Device R2, run the **clear firewall** command.

   ```
   user@R2> clear firewall filter log-discard
   ```

2. From Device R2, run the **show firewall** command.

   ```
   user@R2> show firewall filter log-discard
   Filter: /log-discard
   Counters:
   Name                                              Bytes              Packets
   counter                                               0
   0
   ```

Pinging the 192.0.2.101 Address

Purpose Send packets to the destination address.

Action From Device R1, run the **ping** command.

```
user@R1> ping 192.0.2.101
PING 192.0.2.101 (192.0.2.101): 56 data bytes
^C
--- 192.0.2.101 ping statistics ---
4 packets transmitted, 0 packets received, 100% packet loss
```

Meaning As expected, the ping request fails, and no response is sent. The packets are being discarded.

Checking the Output Filter

Purpose Verify that Device R2's firewall filter is functioning properly.

Action From Device R2, enter the **show firewall filter log-discard** command.

```
user@R2> show firewall filter log-discard
Filter: log-discard
Counters:
Name                                                Bytes              Packets
counter                                               336                    4
```

Meaning As expected, the counter is being incremented.

NOTE: The ping packet carries an additional 20 bytes of IP overhead as well as 8 bytes of ICMP header.

Checking the Community Attribute

Purpose Verify that the route is being tagged with the community attribute.

Action From Device R1, enter the **show route extensive** command, using the neighbor address for Device R2, 192.0.2.101.

```
user@R1> show route 192.0.2.101 extensive

inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
192.0.2.101/32 (1 entry, 1 announced)
TSI:
```

```
KRT in-kernel 192.0.2.101/32 -> {10.0.0.2}
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 684
                Address: 0x94141d8
                Next-hop reference count: 2
                Source: 10.0.0.2
                Next hop: 10.0.0.2 via fe-1/2/0.0, selected
                Session Id: 0x8000a
                State: <Active Ext>
                Local AS:   100 Peer AS:   200
                Age: 53:03
                Validation State: unverified
                Task: BGP_200.10.0.0.2+63097
                Announcement bits (1): 2-KRT
                AS path: 200 I
                Communities: 100:5555
                Accepted
                Localpref: 100
                Router ID: 192.168.0.2
```

Meaning As expected, when Device R2 advertises the 192.0.2.101 route to Device R1, Device R2 adds the 100:5555 community tag.

Related Documentation

- Understanding Forwarding Packets to the Discard Interface on page 511
- Example: Configuring Routing Policy Prefix Lists on page 282

CHAPTER 12

Improving Commit Times with Dynamic Routing Policies

Understanding Dynamic Routing Policies

The verification process required to commit configuration changes can entail a significant amount of overhead and time. For example, changing a prefix in one line of a routing policy that is 20,000 lines long can take up to 20 seconds to commit. It can be useful to be able to commit routing policy changes much more quickly.

In Junos OS Release 9.5 and later, you can configure routing policies and certain routing policy objects in a dynamic database that is not subject to the same verification required in the standard configuration database. As a result, the time it takes to commit changes to the dynamic database is much shorter than for the standard configuration database. You can then reference these policies and policy objects in routing policies you configure in the standard database. BGP is the only protocol to which you can apply routing policies that reference policies and policy objects configured in the dynamic database. After you configure and commit a routing policy based on the objects configured in the dynamic database, you can quickly update any existing routing policy by making changes to the dynamic database configuration.

CAUTION: **Because the Junos OS does not validate configuration changes to the dynamic database, when you use this feature, you should test and verify all configuration changes before committing them.**

Configuring Routing Policies and Policy Objects in the Dynamic Database

Junos OS Release 9.5 and later support a configuration database, the *dynamic database*, which can be edited in a similar way to the standard configuration database but which is not subject to the same verification process to commit configuration changes. As a result, the time it takes to commit a configuration change is much faster. The policies and policy objects defined in the dynamic database can then be referenced in routing policies configured in the standard configuration. The dynamic database is stored in the **/var/run/db/juniper.dyn** directory.

To configure the dynamic database, enter the **configure dynamic** command to enter the configuration mode for the dynamic database:

```
user@host> configure dynamic
Entering configuration mode

[edit dynamic]
user@host#
```

In this dynamic configuration database, you can configure the following statements at the **[edit policy-options]** hierarchy level:

- **as-path** ***name***
- **as-path-group** ***group-name***
- **community** ***community-name***
- **condition** ***condition-name***
- **prefix-list** ***prefix-list-name***
- **policy-statement** ***policy-statement-name***

NOTE: No other configuration is supported at the **[edit dynamic]** hierarchy level.

Use the **policy-statement** ***policy-statement-name*** statement to configure routing policies as you would in the standard configuration database.

To exit configuration mode for the dynamic database, issue the **exit configuration-mode** command from any level within the **[edit dynamic]** hierarchy, or use the **exit** command from the top level.

Configuring Routing Policies Based on Dynamic Database Configuration

In the standard configuration mode, you can configure routing policies that reference policies and policy objects configured at the **[edit dynamic]** hierarchy level in the dynamic database. To define a routing policy that references the dynamic database configuration, include the **dynamic-db** statement at the **[edit policy-options policy-statement** ***policy-statement-name*****]** hierarchy level:

```
[edit policy-options]
policy-statement policy-statement-name {
  dynamic-db;
}
```

You can also define specific policy objects based on the configuration of these objects in the dynamic database. To define a policy object based on the dynamic database, include the **dynamic-db** statement with the following statements at the **[edit policy-options]** hierarchy level:

- **as-path** ***name***
- **as-path-group** ***group-name***
- **community** ***community-name***
- **condition** ***condition-name***
- **prefix-list** ***prefix-list-name***

In the standard configuration, you can also define a routing policy that references any policy object you have configured in the standard configuration that references an object configured in the dynamic database.

For example, in standard configuration mode, you configure a prefix list **prefix-list pl2** that references a prefix list, also named **prefix-list pl2**, that has been configured in the dynamic database:

```
[edit policy-options]
prefix-list pl2 {
  dynamic-db; # Reference a prefix list configured in the dynamic database.
}
```

You then configure a routing policy in the standard configuration that includes **prefix-list pl2**:

```
[edit policy-options]
policy-statement one {
  term term1 {
    from {
      prefix-list pl2; # Include the prefix list configured in the standard configuration
        # database, but which references a prefix list configured in the dynamic database.
    }
    then accept;
  }
  then reject;
}
```

If you need to update the configuration of **prefix-list pl2**, you do so in the dynamic database configuration using the **[edit dynamic]** hierarchy level. This enables you to make commit configuration changes to the prefix list more quickly than you can in the standard configuration database.

NOTE: If you are downgrading the Junos OS to Junos OS Release 9.4 or earlier, you must first delete any routing policies that reference the dynamic database. That is, you must delete any routing policies or policy objects configured with the **dynamic-db** statement.

Applying Dynamic Routing Policies to BGP

BGP is the only routing protocol to which you can apply routing policies that reference the dynamic database configuration. You must apply these policies in the standard configuration. Dynamic policies can be applied to BGP export or import policy. They can also be applied at the global, group, or neighbor hierarchy level.

To apply a BGP export policy, include the **export [*policy-names*]** statement at the **[edit protocols bgp]**, **[edit protocols bgp group *group-name*]**, or **[edit protocols bgp group *group-name* neighbor *address*]** hierarchy level.

```
[edit]
protocols
  bgp {
    export [ policy-names ];
  }
}
```

To apply a BGP import policy, include the **import [*policy-names*]** statement at the **[edit protocols bgp]**, **[edit protocols bgp group *group-name*]**, or **[edit protocols bgp group *group-name* neighbor *address*]** hierarchy level.

```
[edit]
protocols
  bgp {
    import [ policy-names ];
  }
}
```

Include one or more policy names configured in that standard configuration at the **[edit policy-options *policy-statement*]** hierarchy level that reference policies configured in the dynamic database.

Preventing Reestablishment of BGP Peering Sessions After NSR Routing Engine Switchover

If you have active nonstop routing (NSR) enabled, the dynamic database is not synchronized with the backup Routing Engine. As a result, if a switchover to a backup Routing Engine occurs, import and export policies running on the master Routing Engine at the time of the switchover might no longer be available. Therefore, you might want to prevent a BGP peering session from automatically being reestablished as soon as a switchover occurs.

You can configure the router not to reestablish a BGP peering session after an active nonstop routing switchover either for a specified period or until you manually reestablish the session. Include the **idle-after-switch-over (*seconds* | forever)** statement at the **[edit**

protocols bgp], **[edit protocols bgp group *group-name*]**, or **[edit protocols bgp group *group-name* neighbor *address*]** hierarchy level:

```
[edit]
bgp {
  protocols {
    idle-after-switch-over (seconds | never);
  }
}
```

For ***seconds***, specify a value from 1 through 4,294,967,295 ($2^{32} - 1$). The BGP peering session is not reestablished until after the specified period. If you specify the **forever** option, the BGP peering session is not established until you issue the **clear bgp neighbor** command.

Related Documentation

- Example: Configuring Dynamic Routing Policies on page 527
- *Junos OS High Availability Library for Routing Devices*

Example: Configuring Dynamic Routing Policies

This example shows how to configure routing policy objects in a dynamic database that is not subject to the same verification required in the standard configuration database.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

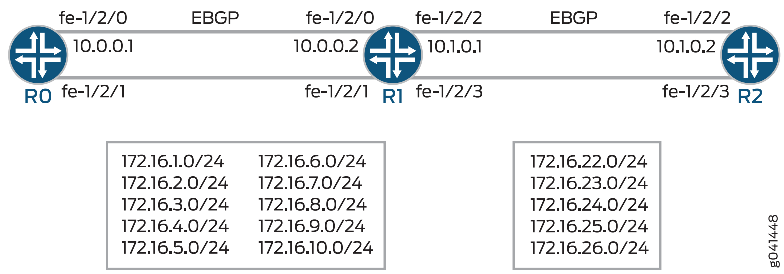
The verification process required to commit configuration changes can entail a significant amount of overhead and time.

The time it takes to commit changes to the dynamic database is much shorter than for the standard configuration database. You can reference these policies and policy objects in routing policies you configure in the standard database. BGP is the only protocol to which you can apply routing policies that reference policies and policy objects configured in the dynamic database. After you configure and commit a routing policy based on the objects configured in the dynamic database, you can quickly update any existing routing policy by making changes to the dynamic database configuration.

> CAUTION: **Because Junos OS does not validate configuration changes to the dynamic database, when you use this feature, you should test and verify all configuration changes before committing them.**

Figure 45 on page 528 shows the sample network.

Figure 45: Dynamic Routing Policy Sample Network



The example includes three routers with external BGP (EBGP) sessions established. Only Device R1 makes use of the dynamic database.

On Device R0's fe-1/2/1 interface, multiple IPv4 interfaces are configured, and a routing policy injects these prefixes into BGP, using the **from interface fe-1/2/1.0** policy condition as a shorthand method for specifying all of the IP addresses configured on Device R0's fe-1/2/1 interface.

Likewise, on Device R2's fe-1/2/3 interface, multiple IPv4 addresses are configured, and a routing policy injects these prefixes into BGP. Device R2's configuration is slightly different from Device R0's in that Device R2's configuration demonstrates the use of a prefix list.

On Device R1, in the dynamic database, two prefix lists are defined, one for the interface addresses learned from Device R0 and another for the interface addresses learned from Device R2. Device R1's standard database contains routing policies with prefix lists that are similar to those defined in the dynamic database.

In its peer session with Device R0, Device R1 has the static-database policies applied. In contrast, in its peer session with Device R2, Device R1's configuration references the dynamic database.

The results of these different configurations are analyzed in the "Verification" on page 537 section.

"CLI Quick Configuration" on page 528 shows the configuration for all of the devices in Figure 45 on page 528.

The section "Step-by-Step Procedure" on page 531 describes the steps on Device R1's dynamic database.

The section "Step-by-Step Procedure" on page 531 describes the steps on Device R1's standard database.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R0

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces fe-1/2/1 unit 0 family inet address 172.16.4.1/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.3.1/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.2.1/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.1.1/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.5.1/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.6.1/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.7.1/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.8.1/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.9.1/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.10.1/24
set interfaces lo0 unit 0 family inet address 10.255.14.151/32
set protocols bgp group ext type external
set protocols bgp group ext neighbor 10.0.0.2 export t2
set protocols bgp group ext neighbor 10.0.0.2 peer-as 200
set policy-options policy-statement t2 from interface fe-1/2/0.0
set policy-options policy-statement t2 from interface fe-1/2/1.0
set policy-options policy-statement t2 then accept
set routing-options router-id 10.255.14.151
set routing-options autonomous-system 100
```

Device R1 Dynamic Database

```
[edit dynamic]
set policy-options prefix-list dyn_prfx1 172.16.1.0/24
set policy-options prefix-list dyn_prfx1 172.16.2.0/24
set policy-options prefix-list dyn_prfx1 172.16.3.0/24
set policy-options prefix-list dyn_prfx1 172.16.4.0/24
set policy-options prefix-list dyn_prfx1 172.16.5.0/24
set policy-options prefix-list dyn_prfx1 172.16.6.0/24
set policy-options prefix-list dyn_prfx1 172.16.7.0/24
set policy-options prefix-list dyn_prfx1 172.16.8.0/24
set policy-options prefix-list dyn_prfx2 172.16.2.0/24
set policy-options prefix-list dyn_prfx2 172.16.3.0/24
set policy-options prefix-list dyn_prfx2 172.16.4.0/24
set policy-options prefix-list dyn_prfx2 172.16.5.0/24
set policy-options prefix-list dyn_prfx2 172.16.6.0/24
set policy-options policy-statement dyn_policy1 term t1 from prefix-list dyn_prfx1
set policy-options policy-statement dyn_policy1 term t1 then accept
set policy-options policy-statement dyn_policy1 term t2 then reject
set policy-options policy-statement dyn_policy2 term t1 from prefix-list dyn_prfx2
set policy-options policy-statement dyn_policy2 term t1 then accept
set policy-options policy-statement dyn_policy2 term t2 then reject
```

Device R1 Standard Database

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces fe-1/2/2 unit 0 family inet address 10.1.0.1/30
set interfaces fe-1/2/1 unit 0 family inet address 172.16.4.2/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.3.2/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.2.2/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.1.2/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.5.2/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.6.2/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.7.2/24
```

```
set interfaces fe-1/2/1 unit 0 family inet address 172.16.8.2/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.9.2/24
set interfaces fe-1/2/1 unit 0 family inet address 172.16.10.2/24
set interfaces fe-1/2/3 unit 0 family inet address 172.16.22.2/24
set interfaces fe-1/2/3 unit 0 family inet address 172.16.23.2/24
set interfaces fe-1/2/3 unit 0 family inet address 172.16.24.2/24
set interfaces fe-1/2/3 unit 0 family inet address 172.16.25.2/24
set interfaces fe-1/2/3 unit 0 family inet address 172.16.26.2/24
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group to_r0 idle-after-switch-over 300
set protocols bgp group to_r0 neighbor 10.0.0.1 import dyn_policy1
set protocols bgp group to_r0 neighbor 10.0.0.1 export dyn_policy2
set protocols bgp group to_r0 neighbor 10.0.0.1 peer-as 100
set protocols bgp group to_R2 import static_policy1
set protocols bgp group to_R2 export static_policy2
set protocols bgp group to_R2 idle-after-switch-over 300
set protocols bgp group to_R2 neighbor 10.1.0.2 peer-as 300
set policy-options prefix-list static_prfx1 172.16.22.0/24
set policy-options prefix-list static_prfx1 172.16.23.0/24
set policy-options prefix-list static_prfx1 172.16.24.0/24
set policy-options prefix-list static_prfx1 172.16.25.0/24
set policy-options prefix-list static_prfx2 172.16.1.0/24
set policy-options prefix-list static_prfx2 172.16.2.0/24
set policy-options prefix-list static_prfx2 172.16.3.0/24
set policy-options prefix-list static_prfx2 172.16.4.0/24
set policy-options policy-statement dyn_policy1 dynamic-db
set policy-options policy-statement dyn_policy2 dynamic-db
set policy-options policy-statement static_policy1 term t1 from prefix-list static_prfx1
set policy-options policy-statement static_policy1 term t1 then accept
set policy-options policy-statement static_policy1 term t2 then reject
set policy-options policy-statement static_policy2 term t1 from prefix-list static_prfx2
set policy-options policy-statement static_policy2 term t1 then accept
set policy-options policy-statement static_policy2 term t2 then reject
set routing-options autonomous-system 200
```

Device R2

```
set interfaces fe-1/2/2 unit 0 family inet address 10.1.0.2/30
set interfaces fe-1/2/3 unit 0 family inet address 172.16.22.1/24
set interfaces fe-1/2/3 unit 0 family inet address 172.16.23.1/24
set interfaces fe-1/2/3 unit 0 family inet address 172.16.24.1/24
set interfaces fe-1/2/3 unit 0 family inet address 172.16.25.1/24
set interfaces fe-1/2/3 unit 0 family inet address 172.16.26.1/24
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols bgp group to_vin neighbor 10.1.0.1 export p1
set protocols bgp group to_vin neighbor 10.1.0.1 peer-as 200
set policy-options prefix-list ppx1 172.16.22.0/24
set policy-options prefix-list ppx1 172.16.23.0/24
set policy-options prefix-list ppx1 172.16.24.0/24
set policy-options prefix-list ppx1 172.16.25.0/24
set policy-options prefix-list ppx1 172.16.26.0/24
set policy-options policy-statement p1 term t1 from family inet
set policy-options policy-statement p1 term t1 from prefix-list ppx1
set policy-options policy-statement p1 term t1 then accept
set routing-options autonomous-system 300
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1's dynamic database:

1. Enter configuration mode for the dynamic database.

```
user@R1> configure dynamic
Entering configuration mode
[edit dynamic]
```

2. Create a prefix list for the interface addresses learned from Device R0.

```
[edit dynamic policy-options prefix-list dyn_prfx1]
user@R1# set 172.16.1.0/24
user@R1# set 172.16.2.0/24
user@R1# set 172.16.3.0/24
user@R1# set 172.16.4.0/24
user@R1# set 172.16.5.0/24
user@R1# set 172.16.6.0/24
user@R1# set 172.16.7.0/24
user@R1# set 172.16.8.0/24
```

3. Create a prefix list for the interface addresses learned from Device R2.

```
[edit dynamic policy-options prefix-list dyn_prfx2]
user@R1# set 172.16.2.0/24
user@R1# set 172.16.3.0/24
user@R1# set 172.16.4.0/24
user@R1# set 172.16.5.0/24
user@R1# set 172.16.6.0/24
```

4. Configure the routing policies.

```
[edit dynamic policy-options policy-statement dyn_policy1]
user@R1# set term t1 from prefix-list dyn_prfx1
user@R1# set term t1 then accept
user@R1# set term t2 then reject

user@R1# set term t1 from prefix-list dyn_prfx2
user@R1# set term t1 then accept
user@R1# set term t2 then reject
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1's standard database:

1. Create the router interfaces.

```
[edit interfaces]
```

```
user@R1# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

user@R1# set fe-1/2/2 unit 0 family inet address 10.1.0.1/30

user@R1# set fe-1/2/1 unit 0 family inet address 172.16.4.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 172.16.3.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 172.16.2.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 172.16.1.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 172.16.5.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 172.16.6.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 172.16.7.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 172.16.8.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 172.16.9.2/24
user@R1# set fe-1/2/1 unit 0 family inet address 172.16.10.2/24

user@R1# set fe-1/2/3 unit 0 family inet address 172.16.2.2/24
user@R1# set fe-1/2/3 unit 0 family inet address 172.16.3.2/24
user@R1# set fe-1/2/3 unit 0 family inet address 172.16.4.2/24
user@R1# set fe-1/2/3 unit 0 family inet address 172.16.5.2/24
user@R1# set fe-1/2/3 unit 0 family inet address 172.16.6.2/24

user@R1# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Create routing policies that reference the policies in the dynamic database.

```
[edit policy-options]
user@R1# set policy-statement dyn_policy1 dynamic-db
user@R1# set policy-statement dyn_policy2 dynamic-db
```

3. Configure BGP peering with Device R0.

```
[edit protocols bgp group to_r0]
user@R1# set neighbor 10.0.0.1 peer-as 100
```

4. Apply the dynamic database policies to the BGP peering with Device R0.

```
[edit protocols bgp group to_r0]
user@R1# set neighbor 10.0.0.1 import dyn_policy1
user@R1# set neighbor 10.0.0.1 export dyn_policy2
```

5. Configure a prefix list for prefixes learned from Device R0.

```
[edit policy-options prefix-list static_prfx2]
user@R1# set 172.16.1.0/24
user@R1# set 172.16.2.0/24
user@R1# set 172.16.3.0/24
user@R1# set 172.16.4.0/24
```

6. Configure a prefix list for prefixes learned from Device R2.

```
[edit policy-options prefix-list static_prfx1]
user@R1# set 172.16.2.0/24
```



```
user@R1# set 172.16.3.0/24
user@R1# set 172.16.4.0/24
user@R1# set 172.16.5.0/24
```

7. Configure the static database policies.

```
[edit policy-options policy-statement static_policy1]
user@R1# set term t1 from prefix-list static_prfx1
user@R1# set term t1 then accept
user@R1# set term t2 then reject

[edit policy-options policy-statement static_policy2]
user@R1# set term t1 from prefix-list static_prfx2
user@R1# set term t1 then accept
user@R1# set term t2 then reject
```

8. Configure BGP peering with Device R2.

```
[edit protocols bgp group to_R2]
user@R1# set neighbor 10.1.0.2 peer-as 300
```

9. Apply the static database policies to the BGP peering with Device R2.

```
[edit protocols bgp group to_R2]
user@R1# set import static_policy1
user@R1# set export static_policy2
```

10. (Optional) Configure the router not to reestablish the BGP peering sessions after an active nonstop routing switchover either for a specified period or until you manually reestablish the session.

    This statement is particularly useful with dynamic routing policies because the dynamic database is not synchronized with the backup Routing Engine when nonstop active routing (NSR) is enabled. As a result, if a switchover to a backup Routing Engine occurs, import and export policies running on the master Routing Engine at the time of the switchover might no longer be available. Therefore, you might want to prevent a BGP peering session from automatically being reestablished as soon as a switchover occurs.

```
[edit protocols bgp]
user@R1# set group to_r0 idle-after-switch-over 300
user@R1# set group to_R2 idle-after-switch-over 300
```

11. Configure the autonomous system (AS) number.

```
[edit routing-options]
user@R1# set routing-options autonomous-system 200
```

Results Confirm your configuration by entering the **show** command from configuration mode in the dynamic database, and the **show interfaces**, **show protocols**, **show policy-options** and **show routing-options** commands from configuration mode in the standard database. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Device R1 Dynamic

```
[edit dynamic]
user@R1# show
policy-options {
    prefix-list dyn_prfx1 {
        172.16.1.0/24;
        172.16.2.0/24;
        172.16.3.0/24;
        172.16.4.0/24;
        172.16.5.0/24;
        172.16.6.0/24;
        172.16.7.0/24;
        172.16.8.0/24;
    }
    prefix-list dyn_prfx2 {
        172.16.2.0/24;
        172.16.3.0/24;
        172.16.4.0/24;
        172.16.5.0/24;
        172.16.6.0/24;
    }
    policy-statement dyn_policy1 {
        term t1 {
            from {
                prefix-list dyn_prfx1;
            }
            then accept;
        }
        term t2 {
            then reject;
        }
    }
    policy-statement dyn_policy2 {
        term t1 {
            from {
                prefix-list dyn_prfx2;
            }
            then accept;
        }
        term t2 {
            then reject;
        }
    }
}
```

Device R1 Standard

```
[edit]
user@R1# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.2/30;
        }
    }
```



```
}
fe-1/2/1 {
   unit 0 {
      family inet {
         address 172.16.4.2/24;
         address 172.16.3.2/24;
         address 172.16.2.2/24;
         address 172.16.1.2/24;
         address 172.16.5.2/24;
         address 172.16.6.2/24;
         address 172.16.7.2/24;
         address 172.16.8.2/24;
         address 172.16.9.2/24;
         address 172.16.10.2/24;
      }
   }
}
fe-1/2/2 {
   unit 0 {
      family inet {
         address 10.1.0.1/30;
      }
   }
}
fe-1/2/3 {
   unit 0 {
      family inet {
         address 172.16.2.2/24;
         address 172.16.3.2/24;
         address 172.16.4.2/24;
         address 172.16.5.2/24;
         address 172.16.6.2/24;
      }
   }
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.2/32;
      }
   }
}

user@R1# show protocols
bgp {
   group to_r0 {
      idle-after-switch-over 300;
      neighbor 10.0.0.1 {
         import dyn_policy1;
         export dyn_policy2;
         peer-as 100;
      }
   }
   group to_R2 {
      import static_policy1;
      export static_policy2;
```

```
        idle-after-switch-over 300;
        neighbor 10.1.0.2 {
            peer-as 300;
        }
    }
}
```

```
user@R1# show policy-options
prefix-list static_prfx1 {
    172.16.2.0/24;
    172.16.3.0/24;
    172.16.4.0/24;
    172.16.5.0/24;
}
prefix-list static_prfx2 {
    172.16.1.0/24;
    172.16.2.0/24;
    172.16.3.0/24;
    172.16.4.0/24;
}
policy-statement dyn_policy1 {
    dynamic-db;
}
policy-statement dyn_policy2 {
    dynamic-db;
}
policy-statement static_policy1 {
    term t1 {
        from {
            prefix-list static_prfx1;
        }
        then accept;
    }
    term t2 {
        then reject;
    }
}
policy-statement static_policy2 {
    term t1 {
        from {
            prefix-list static_prfx2;
        }
        then accept;
    }
    term t2 {
        then reject;
    }
}
```

```
user@R1# show routing-options
autonomous-system 200;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Checking the Configured Policies on Device R1

Purpose Verify that Device R1 has the dynamic and static policies in effect.

Action From Device R1, enter the **show policy** command.

```
user@R1> show policy
Configured policies:
dyn_policy1
dyn_policy2
static_policy1
static_policy2
dyn_policy1
dyn_policy2
```

Meaning The dynamic policies are listed two times because they are configured two times, the first and central configuration in the dynamic database. The secondary configuration is in the static database, where the dynamic database is referenced, as shown here:

Configured in the Dynamic Database

```
policy-statement dyn_policy1 {
  term t1 {
    from {
      prefix-list dyn_prfx1;
    }
    then accept;
  }
  term t2 {
    then reject;
  }
}
policy-statement dyn_policy2 {
  term t1 {
    from {
      prefix-list dyn_prfx2;
    }
    then accept;
  }
```



```
  term t2 {
    then reject;
  }
}
```

Referenced from the Static Database

```
policy-statement dyn_policy1 {
  dynamic-db;
}
policy-statement dyn_policy2 {
  dynamic-db;
}
```

Checking the Routes Advertised from Device R0 to Device R1

Purpose Verify that Device R0's routing policy is working.

Action From Device R0, enter the **show route advertising-protocol bgp** command, using the neighbor address for Device R1.

```
user@R0> show route advertising-protocol bgp 10.0.0.2
inet.0: 28 destinations, 28 routes (28 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.0/24              Self                                    I
* 172.16.2.0/24              Self                                    I
* 172.16.3.0/24              Self                                    I
* 172.16.4.0/24              Self                                    I
* 172.16.5.0/24              Self                                    I
* 172.16.6.0/24              Self                                    I
* 172.16.7.0/24              Self                                    I
* 172.16.8.0/24              Self                                    I
* 172.16.9.0/24              Self                                    I
* 172.16.10.0/24             Self                                    I
* 10.0.0.0/30             Self                                    I
```

Meaning Device R0 is sending the expected routes to Device R1.

Checking the Routes That Device R1 Is Receiving from Device R0

Purpose Verify that Device R1's import routing policy is working.

Action From Device R1, enter the **show route receive-protocol bgp** command, using the neighbor address for Device R0.

```
user@R1> show route receive-protocol bgp 10.0.0.1
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.1.0/24              10.0.0.1                                100 I
  172.16.2.0/24              10.0.0.1                                100 I
  172.16.3.0/24              10.0.0.1                                100 I
  172.16.4.0/24              10.0.0.1                                100 I
  172.16.5.0/24              10.0.0.1                                100 I
```



```
  172.16.6.0/24             10.0.0.1                                  100 I
  172.16.7.0/24             10.0.0.1                                  100 I
  172.16.8.0/24             10.0.0.1                                  100 I
```

Meaning Some of the routes that are sent by Device R0 are not received by Device R1. The routes 172.16.9.0/24, 172.16.10.0/24, and 10.0.0.0/30 are missing. This is because Device R1's import policy, applied to the BGP peering session with Device R0 using the **import dyn_policy1** statement, specifically defines a prefix list limited to the following routes:

```
prefix-list dyn_prfx1 {
  172.16.1.0/24;
  172.16.2.0/24;
  172.16.3.0/24;
  172.16.4.0/24;
  172.16.5.0/24;
  172.16.6.0/24;
  172.16.7.0/24;
  172.16.8.0/24;
}
```

Checking the Routes Advertised from Device R2 to Device R1

Purpose Verify that Device R2's routing policy is working.

Action From Device R2, enter the **show route advertising-protocol bgp** command, using the neighbor address for Device R1.

```
user@R2> show route advertising-protocol bgp 10.1.0.1
inet.0: 17 destinations, 17 routes (17 active, 0 holddown, 0 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.2.0/24             Self                                     I
* 172.16.3.0/24             Self                                     I
* 172.16.4.0/24             Self                                     I
* 172.16.5.0/24             Self                                     I
* 172.16.6.0/24             Self                                     I
```

Meaning Device R2 is sending the expected routes to Device R1.

Checking the Routes That Device R1 Is Receiving from Device R2

Purpose Verify that Device R1's import routing policy is working.

Action From Device R1, enter the **show route receive-protocol bgp** command, using the neighbor address for Device R0.

```
user@R1> show route receive-protocol bgp 10.1.0.2
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
  172.16.2.0/24             10.1.0.2                                 300 I
```



```
  172.16.3.0/24              10.1.0.2                                 300 I
  172.16.4.0/24              10.1.0.2                                 300 I
  172.16.5.0/24              10.1.0.2                                 300 I
```

Meaning One of the routes that is sent by Device R2 is not received by Device R1. The route 172.16.6.0/24 is missing. This is because Device R1's import policy, applied to the BGP peering session with Device R2 using the **import static_policy1** statement, specifically defines a prefix list limited to the following routes:

```
prefix-list static_prfx1 {
  172.16.2.0/24;
  172.16.3.0/24;
  172.16.4.0/24;
  172.16.5.0/24;
}
```

Checking the Routes That Device R1 Is Advertising to Device R0

Purpose Verify that Device R1's export routing policy is working.

Action From Device R1, enter the **show route advertising-protocol bgp** command, using the neighbor address for Device R0.

```
user@R1> show route advertising-protocol bgp 10.0.0.1
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.2.0/24              Self                                    I
* 172.16.3.0/24              Self                                    I
* 172.16.4.0/24              Self                                    I
* 172.16.5.0/24              Self                                    I
* 172.16.6.0/24              Self                                    I
```

Meaning Perhaps unexpectedly, the route that Device R1 did not receive through BGP from Device R2 (172.16.6.0/24) is nonetheless being advertised by Device R1 through BGP to Device R0. This is happening for two reasons. The first reason is that route 172.16.6.0/24 is in Device R1's routing table, albeit as a direct route, as shown here:

```
user@R1> show route 172.16.6.0/24 protocol direct
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.6.0/24         *[Direct/0] 2d 22:51:41
                    > via fe-1/2/3.0
```

The second reason is that Device R1's export policy, applied to the BGP peering session with Device R0 using the **export dyn_policy2** statement, specifically defines a prefix list limited to the following routes:

```
prefix-list dyn_prfx2 {
  172.16.2.0/24;
```



```
    172.16.3.0/24;
    172.16.4.0/24;
    172.16.5.0/24;
    172.16.6.0/24;
  }
```

Note the inclusion of 172.16.6.0/24.

Checking the Routes That Device R1 Is Advertising to Device R2

Purpose Verify that Device R1's export routing policy is working.

Action From Device R1, enter the **show route advertising-protocol bgp** command, using the neighbor address for Device R2.

```
user@R1> show route advertising-protocol bgp 10.1.0.2
inet.0: 35 destinations, 51 routes (35 active, 0 holddown, 4 hidden)
  Prefix                  Nexthop              MED     Lclpref    AS path
* 172.16.1.0/24              Self                                    I
* 172.16.2.0/24              Self                                    I
* 172.16.3.0/24              Self                                    I
* 172.16.4.0/24              Self                                    I
```

Meaning Device R1 is sending the expected routes to Device R2. Device R1's export policy, applied to the BGP peering session with Device R2 using the **export static_policy2** statement, specifically defines a prefix list limited to the following routes:

```
  prefix-list static_prfx2 {
    172.16.1.0/24;
    172.16.2.0/24;
    172.16.3.0/24;
    172.16.4.0/24;
  }
```

Related Documentation

- Understanding Dynamic Routing Policies on page 523
- Example: Configuring Routing Policy Prefix Lists on page 282

CHAPTER 13

Testing Before Applying Routing Policies

Understanding Routing Policy Tests

Routing policy tests provide a method for verifying the effectiveness of your policies before applying them on the routing device. Before applying a routing policy, you can issue the **test policy** command to ensure that the policy produces the results that you expect:

user@host> **test policy** ***policy-name*** **prefix**

Keep in mind that different protocols have different default policies that get applied if the prefix does not match the configured policy. For BGP this is accept, but for RIP it is reject. The **test policy** command always uses accept as the default policy, so unless you explicitly reject all routes that you do not want to match you might see more routes matching than you want.

The default policy of the **test policy** command accepts all routes from all protocols. Test output can be misleading when you are evaluating protocol-specific conditions. For example, if you define a policy for BGP that accepts routes of a specified prefix and apply it to BGP as an export policy, BGP routes that match the prefix are advertised to BGP peers. However, if you test the same policy using the **test policy** command, the test output might indicate that non-BGP routes have been accepted.

Example: Testing a Routing Policy

Test the following policy, which looks for unwanted routes and rejects them:

```
[edit policy-options]
policy-statement reject-unwanted-routes {
  term drop-these-routes {
    from {
      route-filter 0/0 exact;
      route-filter 10/8 orlonger;
      route-filter 172.16/12 orlonger;
      route-filter 192.168/16 orlonger;
      route-filter 224/3 orlonger;
    }
    then reject;
```

```
  }
}
```

Test this policy against all routes in the routing table:

user@host> **test policy reject-unwanted-routes 0/0**

Test this policy against a specific set of routes:

user@host> **test policy reject-unwanted-routes 10.49.0.0/16**

Related Documentation

- Example: Testing a Routing Policy with Complex Regular Expressions on page 544

Example: Testing a Routing Policy with Complex Regular Expressions

This example shows how to test a routing policy using the **test policy** command to ensure that the policy produces the results that you expect before you apply it in a production environment. Regular expressions, especially complex ones, can be tricky to get right. This example shows how to use the **test policy** command to make sure that your regular expressions have the intended effect.

Requirements

No special configuration beyond device initialization is required before you configure this example.

Overview

This example shows two routing devices with an external BGP (EBGP) connection between them. Device R2 uses the BGP session to send customer routes to Device R1. These static routes have multiple community values attached.

```
user@R2> show route match-prefix 172.16.* detail

inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
172.16.1.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Reject
                Address: 0x8fd0dc4
                Next-hop reference count: 8
                State: <Active Int Ext>
                Local AS: 64511
                Age: 21:32:13
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
                Communities: 64510:1 64510:10 64510:11 64510:100 64510:111
```



```
172.16.2.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Reject
                Address: 0x8fd0dc4
                Next-hop reference count: 8
                State: <Active Int Ext>
                Local AS: 64511
                Age: 21:32:13
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
                Communities: 64510:2 64510:20 64510:22 64510:200 64510:222

172.16.3.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Reject
                Address: 0x8fd0dc4
                Next-hop reference count: 8
                State: <Active Int Ext>
                Local AS: 64511
                Age: 21:32:13
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
                Communities: 64510:3 64510:30 64510:33 64510:300 64510:333

172.16.4.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Reject
                Address: 0x8fd0dc4
                Next-hop reference count: 8
                State: <Active Int Ext>
                Local AS: 64511
                Age: 21:32:13
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
                Communities: 64510:4 64510:40 64510:44 64510:400 64510:444
```

To test a complex regular expression, Device R2 has a policy called **test-regex** that locates routes. The policy is configured like this:

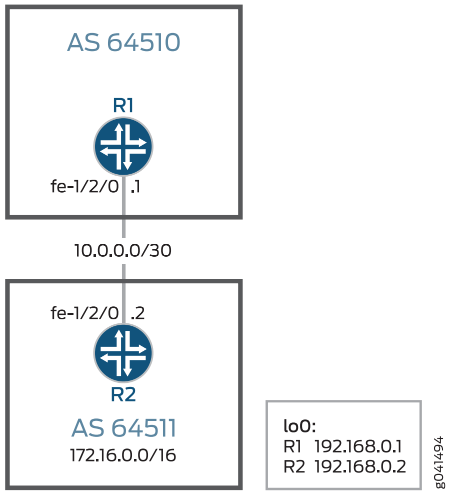
```
policy-statement test-regex {
  term find-routes {
    from community complex-regex;
    then accept;
  }
  term reject-the-rest {
    then reject;
  }
}
community complex-regex members "^64510:[13].*$";
```

This regular expression matches community values beginning with either 1 or 3.

Topology

Figure 46 on page 546 shows the sample network.

Figure 46: Routing Policy Test for Complex Regular Expressions



"CLI Quick Configuration" on page 546 shows the configuration for all of the devices in Figure 46 on page 546.

The section "Step-by-Step Procedure" on page 547 describes the steps on Device R2.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64511
set protocols bgp group ext neighbor 10.0.0.2
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext peer-as 64510
set protocols bgp group ext neighbor 10.0.0.1
set policy-options policy-statement send-static term 1 from protocol static
set policy-options policy-statement send-static term 1 then accept
set policy-options policy-statement send-static term 2 then reject
set policy-options policy-statement test-regex term find-routes from community
   complex-regex
set policy-options policy-statement test-regex term find-routes then accept
```



```
set policy-options policy-statement test-regex term reject-the-rest then reject
set policy-options community complex-regex members "^64510:[13].*$"
set routing-options static route 172.16.1.0/24 reject
set routing-options static route 172.16.1.0/24 community 64510:1
set routing-options static route 172.16.1.0/24 community 64510:10
set routing-options static route 172.16.1.0/24 community 64510:11
set routing-options static route 172.16.1.0/24 community 64510:100
set routing-options static route 172.16.1.0/24 community 64510:111
set routing-options static route 172.16.2.0/24 reject
set routing-options static route 172.16.2.0/24 community 64510:2
set routing-options static route 172.16.2.0/24 community 64510:20
set routing-options static route 172.16.2.0/24 community 64510:22
set routing-options static route 172.16.2.0/24 community 64510:200
set routing-options static route 172.16.2.0/24 community 64510:222
set routing-options static route 172.16.3.0/24 reject
set routing-options static route 172.16.3.0/24 community 64510:3
set routing-options static route 172.16.3.0/24 community 64510:30
set routing-options static route 172.16.3.0/24 community 64510:33
set routing-options static route 172.16.3.0/24 community 64510:300
set routing-options static route 172.16.3.0/24 community 64510:333
set routing-options static route 172.16.4.0/24 reject
set routing-options static route 172.16.4.0/24 community 64510:4
set routing-options static route 172.16.4.0/24 community 64510:40
set routing-options static route 172.16.4.0/24 community 64510:44
set routing-options static route 172.16.4.0/24 community 64510:400
set routing-options static route 172.16.4.0/24 community 64510:444
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64511
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set fe-1/2/0 unit 0 family inet address 10.0.0.2/30

user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
```

2. Configure BGP.

   Apply the import policy to the BGP peering session with Device R2.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set peer-as 64510
user@R2# set neighbor 10.0.0.1
```

3. Configure the routing policy that sends static routes.

```
[edit policy-options policy-statement send-static]
user@R2# set term 1 from protocol static
```

```
user@R2# set term 1 then accept
user@R2# set term 2 then reject
```

4. Configure the routing policy that tests a regular expression.

```
[edit policy-options policy-statement test-regex]
user@R2# set term find-routes from community complex-regex
user@R2# set term find-routes then accept
user@R2# set term reject-the-rest then reject

[edit policy-options community]
user@R2# set complex-regex members "^64510:[13].*$"
```

5. Configure the static routes and attaches community values.

```
[edit routing-options static route 172.16.1.0/24]
user@R2# set reject
user@R2# set community [ 64510:1 64510:10 64510:11 64510:100 64510:111 ]

[edit routing-options static route 172.16.2.0/24]
user@R2# set reject
user@R2# set community [ 64510:2 64510:20 64510:22 64510:200 64510:222 ]

[edit routing-options static route 172.16.3.0/24]
user@R2# set reject
user@R2# set community [ 64510:3 64510:30 64510:33 64510:300 64510:333 ]

[edit routing-options static route 172.16.4.0/24]
user@R2# set reject
user@R2# set community [ 64510:4 64510:40 64510:44 64510:400 64510:444 ]
```

6. Configure the autonomous system (AS) number and the router ID.

   This affects Device R2's routing table, and as no impact on Device R1 and Device R3.

```
[edit routing-options ]
user@R2# set router-id 192.168.0.2
user@R2# set autonomous-system 64511
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      address 10.0.0.2/30;
    }
  }
```



```
}
lo0 {
   unit 0 {
      family inet {
         address 192.168.0.2/32;
      }
   }
}
```

user@R2# **show protocols**

```
bgp {
   group ext {
      type external;
      peer-as 64510;
      neighbor 10.0.0.1;
   }
}
```

user@R2# **show policy-options**

```
policy-statement send-static {
   term 1 {
      from protocol static;
      then accept;
   }
   term 2 {
      then reject;
   }
}
policy-statement test-regex {
   term find-routes {
      from community complex-regex;
      then accept;
   }
   term reject-the-rest {
      then reject;
   }
}
community complex-regex members "^64510:[13].*$";
```

user@R2# **show routing-options**

```
static {
   route 172.16.1.0/24 {
      reject;
      community [ 64510:1 64510:10 64510:11 64510:100 64510:111 ];
   }
   route 172.16.2.0/24 {
      reject;
      community [ 64510:2 64510:20 64510:22 64510:200 64510:222 ];
   }
   route 172.16.3.0/24 {
      reject;
      community [ 64510:3 64510:30 64510:33 64510:300 64510:333 ];
   }
   route 172.16.4.0/24 {
      reject;
      community [ 64510:4 64510:40 64510:44 64510:400 64510:444 ];
   }
```

```
}
router-id 192.168.0.2;
autonomous-system 64511;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Test to See Which Communities Match the Regular Expression

Purpose You can test the regular expression and its policy by using the test policy ***policy-name*** command.

Action

1. On Device R2, run the **test policy test-regex 0/0** command.

```
user@R2> test policy test-regex 0/0


inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[Static/5] 1d 00:32:50
                      Reject
172.16.3.0/24      *[Static/5] 1d 00:32:50
                      Reject

Policy test-regex: 2 prefix accepted, 5 prefix rejected
```

2. On Device R2, change the regular expression to match a community value containing any number of instances of the digit 2.

```
[edit policy-options community complex-regex]
user@R2# delete members "^64510:[13].*$"
user@R2# set members "^65020:2+$"
user@R2# commit
```

3. On Device R2, rerun the **test policy test-regex 0/0** command.

```
user@R2> test policy test-regex 0/0

inet.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.2.0/24      *[Static/5] 1d 00:31:36
                      Reject

Policy test-regex: 1 prefix accepted, 6 prefix rejected
```

Meaning The 172.16.1.0 /24 and 172.16.3.0/24 routes both have communities attached that match the ^64510:[13].*$ expression. The 172.16.2.0/24 route has communities that match the ^65020:2+$ expression.

Related Documentation

- Understanding Routing Policy Tests on page 543
- Understanding How to Define BGP Communities and Extended Communities on page 361
- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions on page 309

PART 3

Configuring Firewall Filters

CHAPTER 14

Understanding How Firewall Filters Protect Your Network

Firewall Filters Overview

Firewall filters provide a means of protecting your router (and switch) from excessive traffic transiting the router (and switch) to a network destination or destined for the Routing Engine. Firewall filters that control local packets can also protect your router (and switch) from external incidents.

You can configure a firewall filter to do the following:

- Restrict traffic destined for the Routing Engine based on its source, protocol, and application.
- Limit the traffic rate of packets destined for the Routing Engine to protect against flood, or denial-of-service (DoS) attacks.

- Address special circumstances associated with fragmented packets destined for the Routing Engine. Because the device evaluates every packet against a firewall filter (including fragments), you must configure the filter to accommodate fragments that do not contain packet header information. Otherwise, the filter discards all but the first fragment of a fragmented packet.

Related Documentation

- Stateless Firewall Filter Types on page 560
- Guidelines for Configuring Firewall Filters on page 576
- Guidelines for Applying Standard Firewall Filters on page 581
- Understanding How to Use Standard Firewall Filters on page 559

Router Data Flow Overview

The Junos® operating system (Junos OS) provides a *policy framework*, which is a collection of Junos OS policies that enable you to control flows of routing information and packets within the router.

Flow of Routing Information

Routing information is the information about routes learned by the routing protocols from a router's neighbors. This information is stored in routing tables. The routing protocols advertise active routes only from the routing tables. An *active route* is a route that is chosen from all routes in the routing table to reach a destination.

To control which routes the routing protocols place in the routing tables and which routes the routing protocols advertise from the routing tables, you can configure *routing policies*, which are sets of rules that the policy framework uses to preempt default routing policies.

The Routing Engine, which runs the router's control plane software, handles the flow of routing information between the routing protocols and the routing tables and between the routing tables and the forwarding table. The Routing Engine runs the Junos OS and routing policies and stores the active router configuration, the master routing table, and the master forwarding table,

Flow of Data Packets

Data packets are chunks of data that transit the router as they are being forwarded from a source to a destination. When a router receives a data packet on an interface, it determines where to forward the packet by looking in the forwarding table for the best route to a destination. The router then forwards the data packet toward its destination through the appropriate interface.

The Packet Forwarding Engine, which is the central processing element of the router's forwarding plane, handles the flow of data packets in and out of the router's physical interfaces. Although the Packet Forwarding Engine contains Layer 3 and Layer 4 header information, it does not contain the packet data itself (the packet's payload).

Flow of Local Packets

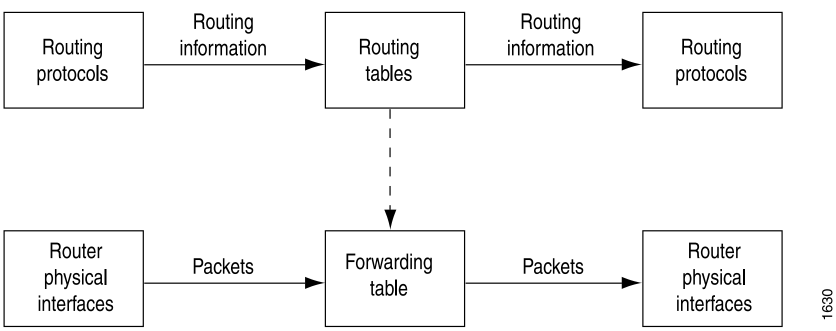
Local packets are chunks of data that are destined for or sent by the router. Local packets usually contain routing protocol data, data for IP services such as Telnet or SSH, and data for administrative protocols such as the Internet Control Message Protocol (ICMP). When the Routing Engine receives a local packet, it forwards the packet to the appropriate process or to the kernel, which are both part of the Routing Engine, or to the Packet Forwarding Engine.

The Routing Engine handles the flow of local packets from the router's physical interfaces and to the Routing Engine.

Interdependent Flows of Routing Information and Packets

Figure 47 on page 557 illustrates the flow of data through a router. Although routing information flows and packet flows are very different from one another, they are also interdependent.

Figure 47: Flows of Routing Information and Packets



Routing policies determine which routes the Routing Engine places in the forwarding table. The forwarding table, in turn, has an integral role in determining the appropriate physical interface through which to forward a packet.

Related Documentation

- Stateless Firewall Filter Overview on page 558
- *Packet Flow Through the Junos OS CoS Process Overview*
- *Understanding BGP Path Selection*
- *Understanding Route Preference Values (Administrative Distance)*
- Understanding Routing Policies on page 17

Stateless Firewall Filter Overview

This topic covers the following information:

Packet Flow Control

To influence which packets are allowed to transit the system and to apply special actions to packets as necessary, you can configure *stateless firewall filters*. A stateless firewall specifies a sequence of one or more packet-filtering rules, called *filter terms*. A filter term specifies *match conditions* to use to determine a match and *actions* to take on a matched packet. A stateless firewall filter enables you to manipulate any packet of a particular protocol family, including fragmented packets, based on evaluation of Layer 3 and Layer 4 header fields. You typically apply a stateless firewall filter to one or more interfaces that have been configured with protocol family features. You can apply a stateless firewall filter to an ingress interface, an egress interface, or both.

Data Packet Flow Control

To control the flow of data packets transiting the device as the packets are being forwarded from a source to a destination, you can apply stateless firewall filters to the input or output of the router's or switch's physical interfaces.

To enforce a specified bandwidth and maximum burst size for traffic sent or received on an interface, you can configure *policers*. Policers are a specialized type of stateless firewall filter and a primary component of the Junos OS *class-of-service* (CoS).

Local Packet Flow Control

To control the flow of local packets between the physical interfaces and the Routing Engine, you can apply stateless firewall filters to the input or output of the *loopback interface*. The loopback interface (**lo0**) is the interface to the Routing Engine and carries no data packets.

Stateless and Stateful Firewall Filters

A stateless firewall filter, also known as an *access control list* (ACL), does not statefully inspect traffic. Instead, it evaluates packet contents statically and does not keep track of the state of network connections. In contrast, a *stateful firewall filter* uses connection state information derived from other applications and past communications in the data flow to make dynamic control decisions.

The *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide* describes *stateless firewall filters*.

Purpose of Stateless Firewall Filters

The basic purpose of a stateless firewall filter is to enhance security through the use of packet filtering. Packet filtering enables you to inspect the components of incoming or outgoing packets and then perform the actions you specify on packets that match the criteria you specify. The typical use of a stateless firewall filter is to protect the Routing Engine processes and resources from malicious or untrusted packets.

Related Documentation

- Router Data Flow Overview on page 556
- Stateless Firewall Filter Types on page 560
- Controlling Network Access Using Traffic Policing Overview on page 971
- *Packet Flow Through the Junos OS CoS Process Overview*

Understanding How to Use Standard Firewall Filters

This topic covers the following information:

- Using Standard Firewall Filters to Affect Local Packets on page 559
- Using Standard Firewall Filters to Affect Data Packets on page 560

Using Standard Firewall Filters to Affect Local Packets

On a router, you can configure one physical loopback interface, **lo0**, and one or more addresses on the interface. The loopback interface is the interface to the Routing Engine, which runs and monitors all the control protocols. The loopback interface carries local packets only. Standard firewall filters applied to the loopback interface affect the local packets destined for or transmitted from the Routing Engine.

NOTE: When you create an additional loopback interface, it is important to apply a filter to it so the Routing Engine is protected. We recommend that when you apply a filter to the loopback interface, you include the **apply-groups** statement. Doing so ensures that the filter is automatically inherited on every loopback interface, including **lo0** and other loopback interfaces.

Trusted Sources

The typical use of a standard stateless firewall filter is to protect the Routing Engine processes and resources from malicious or untrusted packets. To protect the processes and resources owned by the Routing Engine, you can use a standard stateless firewall filter that specifies which protocols and services, or applications, are allowed to reach the Routing Engine. Applying this type of filter to the loopback interface ensures that the local packets are from a trusted source and protects the processes running on the Routing Engine from an external attack.

Flood Prevention

You can create standard stateless firewall filters that limit certain TCP and ICMP traffic destined for the Routing Engine. A router without this kind of protection is vulnerable to TCP and ICMP flood attacks, which are also called denial-of-service (DoS) attacks. For example:

- A TCP flood attack of SYN packets initiating connection requests can overwhelm the device until it can no longer process legitimate connection requests, resulting in denial of service.
- An ICMP flood can overload the device with so many echo requests (ping requests) that it expends all its resources responding and can no longer process valid network traffic, also resulting in denial of service.

Applying the appropriate firewall filters to the Routing Engine protects against these types of attacks.

Using Standard Firewall Filters to Affect Data Packets

Standard firewall filters that you apply to your router's transit interfaces evaluate only the user data packets that transit the router from one interface directly to another as they are being forwarded from a source to a destination. To protect the network as a whole from unauthorized access and other threats at specific interfaces, you can apply firewall filters router transit interfaces .

Related Documentation

- How Standard Firewall Filters Evaluate Packets on page 570
- Guidelines for Configuring Firewall Filters on page 576
- Guidelines for Applying Standard Firewall Filters on page 581

Stateless Firewall Filter Types

This topic covers the following information:

Firewall Filters

The Junos OS standard stateless firewall filters support a rich set of packet-matching criteria that you can use to match on specific traffic and perform specific actions, such as forwarding or dropping packets that match the criteria you specify. You can configure firewall filters to protect the local router or to protect another device that is either directly or indirectly connected to the local router. For example, you can use the filters to restrict the local packets that pass from the router's physical interfaces to the Routing Engine. Such filters are useful in protecting the IP services that run on the Routing Engine, such as Telnet, SSH, and BGP, from denial-of-service attacks.

NOTE: If you configured targeted broadcast for virtual routing and forwarding (VRF) by including the **forward-and-send-to-re** statement, any firewall filter that is configured on the Routing Engine loopback interface (**lo0**) cannot be applied to the targeted broadcast packets that are forwarded to the Routing Engine. This is because broadcast packets are forwarded as flood next hop traffic and not as local next hop traffic, and you can only apply a firewall filter to local next hop routes for traffic directed toward the Routing Engine.

Service Filters

A service filter defines packet-filtering (a set of match conditions and a set of actions) for IPv4 or IPv6 traffic. You can apply a service filter to the inbound or outbound traffic at an adaptive services interface to perform packet filtering on traffic before it is accepted for service processing. You can also apply a service filter to the traffic that is returning to the services interface after service processing to perform postservice processing.

Service filters filter IPv4 and IPv6 traffic only and can be applied to logical interfaces on Adaptive Services PICs, MultiServices PICs, and MultiServices DPCs only. Service filters are not supported on Branch SRX devices.

Simple Filters

Simple filters are supported on Gigabit Ethernet intelligent queuing (IQ2) and Enhanced Queuing Dense Port Concentrator (EQ DPC) interfaces only. Unlike standard filters, simple filters support IPv4 traffic only and have a number of restrictions. For example, you cannot configure a terminating action for a simple filter. Simple filters always accept packets. Also, simple filters can be applied only as input filters. They are not supported on outbound traffic. Simple filters are recommended for metropolitan Ethernet applications.

Related Documentation

- Stateless Firewall Filter Overview on page 558
- Stateless Firewall Filter Components on page 561

Stateless Firewall Filter Components

This topic covers the following information:

Protocol Family

Under the **firewall** statement, you can specify the protocol family for which you want to filter traffic.

Table 27 on page 562 describes the firewall filter protocol families.

Table 27: Firewall Filter Protocol Families

Type of Traffic to Be Filtered	Configuration Statement	Comments
Protocol Independent	**family any**	All protocol families configured on a logical interface.
Internet Protocol version 4 (IPv4)	**family inet**	The **family inet** statement is optional for IPv4.
Internet Protocol version 6 (IPv6)	**family inet6**	
MPLS	**family mpls**	
MPLS-tagged IPv4	**family mpls**	Supports matching on IP addresses and ports, up to five MPLS stacked labels.
MPLS-tagged IPv6	**family mpls**	Supports matching on IP addresses and ports, up to five MPLS stacked labels.
Virtual private LAN service (VPLS)	**family vpls**	
Layer 2 Circuit Cross-Connection	**family ccc**	
Layer 2 Bridging	**family bridge** (for MX Series routers) and **family ethernet-switching** (for EX Series switches)	MX Series routers and EX Series switches only.

Filter Type

Under the **family *family-name*** statement, you can specify the type and name of the filter you want to configure.

Table 28 on page 563 describes the firewall filter types.

Table 28: Filter Types

Filter Type	Configuration Statement	Description
Standard Firewall Filter	**filter** ***filter-name***	Filters the following traffic types: • Protocol independent • IPv4 • IPv6 • MPLS • MPLS-tagged IPv4 • MPLS-tagged IPv6 • VPLS • Layer 2 CCC • Layer 2 bridging (MX Series routers and EX Series switches only)
Service Filter	**service-filter** ***service-filter-name***	Defines packet-filtering to be applied to ingress or egress before it is accepted for service processing or applied to returning service traffic after service processing has completed. Filters the following traffic types: • IPv4 • IPv6 Supported at logical interfaces configured on the following hardware only: • Adaptive Services (AS) PICs on M Series and T Series routers • Multiservices (MS) PICs on M Series and T Series routers • Multiservices (MS) DPCs on MX Series routers (and EX Series switches)
Simple Filter	**simple-filter** ***simple-filter-name***	Defines packet filtering to be applied to ingress traffic only. Filters the following traffic type: • IPv4 Supported at logical interfaces configured on the following hardware only: • Gigabit Ethernet Intelligent Queuing (IQ2) PICs installed on M120, M320, or T Series routers • Enhanced Queuing Dense Port Concentrators (EQ DPCs) installed on MX Series routers (and EX Series switches)

Terms

Under the **filter**, **service-filter**, or **simple-filter** statement, you must configure at least one firewall filter *term*. A term is a named structure in which match conditions and actions are defined. Within a firewall filter, you must configure a unique name for each term.

TIP: For each protocol family on an interface, you can apply no more than one filter in each direction. If you try to apply additional filters for the same protocol family in the same direction, the last filter overwrites the previous filter. You can, however, apply filters from the same protocol family to the input and output direction of the same interface.

All stateless firewall filters contain one or more terms, and each term consists of two components—match conditions and actions. The match conditions define the values or fields that the packet must contain to be considered a match. If a packet is a match, the corresponding action is taken. By default, a packet that does not match a firewall filter is discarded.

If a packet arrives on an interface for which no firewall filter is applied for the incoming traffic on that interface, the packet is accepted by default.

NOTE: A firewall filter with a large number of terms can adversely affect both the configuration commit time and the performance of the Routing Engine.

Additionally, you can configure a stateless firewall filter within the term of another filter. This method enables you to add common terms to multiple filters without having to modify all filter definitions. You can configure one filter with the desired common terms, and configure this filter as a term in other filters. Consequently, to make a change in these common terms, you need to modify only one filter that contains the common terms, instead of multiple filters.

Match Conditions

A firewall filter term must contain at least one packet-filtering criteria, called a *match condition*, to specify the field or value that a packet must contain in order to be considered a match for the firewall filter term. For a match to occur, the packet must match all the conditions in the term. If a packet matches a firewall filter term, the router (or switch) takes the configured action on the packet.

If a firewall filter term contains multiple match conditions, a packet must meet *all* match conditions to be considered a match for the firewall filter term.

If a single match condition is configured with multiple values, such as a range of values, a packet must match only *one* of the values to be considered a match for the firewall filter term.

The scope of match conditions you can specify in a firewall filter term depends on the protocol family under which the firewall filter is configured. You can define various match conditions, including the IP source address field, IP destination address field, TCP or UDP source port field, IP protocol field, Internet Control Message Protocol (ICMP) packet type, IP options, TCP flags, incoming logical or physical interface, and outgoing logical or physical interface. These are pre-defined, or fixed, match conditions.

On MX Series 3D Universal Edge Routers with MPCs or MICs, it is possible to build flexible match conditions for IPv4, IPv6, Layer 2 bridge, CCC, and VPLS protocol families. These flexible match conditions allow a user to specify start location, byte offset, match length, and other parameters within the packet.

Each protocol family supports a different set of match conditions, and some match conditions are supported only on certain routing devices. For example, a number of match conditions for VPLS traffic are supported only on the MX Series 3D Universal Edge Routers.

In the **from** statement in a firewall filter term, you specify characteristics that the packet must have for the action in the subsequent **then** statement to be performed. The characteristics are referred to as *match conditions*. The packet must match all conditions in the **from** statement for the action to be performed, which also means that the order of the conditions in the **from** statement is not important.

If an individual match condition can specify a list of values (such as multiple source and destination addresses) or a range of numeric values, a match occurs if any of the values matches the packet.

If a filter term does not specify match conditions, the term accepts all packets and the actions specified in the term's **then** statement are optional.

NOTE:
Some of the numeric range and bit-field match conditions allow you to specify a text synonym. For a complete list of synonyms:

- If you are using the J-Web interface, select the synonym from the appropriate list.
- If you are using the CLI, type a question mark (?) after the **from** statement.

Actions

The actions specified in a firewall filter term define the actions to take for any packet that matches the conditions specified in the term.

Actions that are configured within a single term are all taken on traffic that matches the conditions configured.

BEST PRACTICE: We strongly recommend that you explicitly configure one or more actions per firewall filter term. Any packet that matches all the conditions of the term is automatically accepted unless the term specifies other or additional actions.

Firewall filter actions fall into the following categories:

Filter-Terminating Actions

A filter-terminating action halts all evaluation of a firewall filter for a specific packet. The router (or switch) performs the specified action, and no additional terms are examined.

Nonterminating Actions

Nonterminating actions are used to perform other functions on a packet, such as incrementing a counter, logging information about the packet header, sampling the packet data, or sending information to a remote host using the system log functionality.

The presence of a nonterminating action, such as **count**, **log**, or **syslog**, without an explicit terminating action, such as **accept**, **discard**, or **reject**, results in a default terminating action of **accept**. If you do not want the firewall filter action to terminate, use the **next term** action after the nonterminating action.

In this example, term 2 is never evaluated, because term 1 has the implicit default **accept** terminating action.

```
[edit firewall filter test]
term 1 {
  from {
    source-address {
      0.0.0.0/0;
    }
  }
  then {
    log;
    <accept> #By default if not specified
  }
}
term 2 {
  then {
    reject;
  }
}
```

In this example, term 2 is evaluated, because term 1 has the explicit **next term** flow control action.

```
[edit firewall filter test]
term 1 {
  from {
    source-address {
      0.0.0.0/0;
    }
  }
  then {
    log;
    next term;
  }
}
term 2 {
  then {
    reject;
  }
}
```


Flow Control Action

For standard stateless firewall filters only, the action **next term** enables the router (or switch) to perform configured actions on the packet and then evaluate the following term in the filter, rather than terminating the filter.

A maximum of 1024 **next term** actions are supported per standard stateless firewall filter configuration. If you configure a standard filter that exceeds this limit, your candidate configuration results in a commit error.

Related Documentation

- Stateless Firewall Filter Types on page 560
- Firewall Filter Flexible Match Conditions on page 667
- *Inserting a New Identifier in a Junos OS Configuration* in the *CLI User Guide*

Stateless Firewall Filter Application Points

After you define the firewall filter, you must apply it to an application point. These application points include logical interfaces, physical interfaces, routing interfaces, and routing instances.

In most cases, you can apply a firewall filter as an *input* filter or an *output* filter, or both at the same time. Input filters take action on packets being received on the specified interface, whereas output filters take action on packets that are transmitted through the specified interface.

You typically apply one filter with multiple terms to a single logical interface, to incoming traffic, outbound traffic, or both. However, there are times when you might want to chain together multiple firewall filters (with single or multiple terms) and apply them to an interface. You use an *input list* to apply multiple firewall filters to the incoming traffic on an interface. You use an *output list* to apply multiple firewall filters to the outbound traffic on an interface. You can include up to 16 filters in an input list or an output list.

There is no limit to the number of filters and counters you can set, but there are some practical considerations. More counters require more terms, and a large number of terms can take a long time to process during a commit operation. However, filters with more than 4000 terms and counters have been implemented successfully.

Table 29 on page 568 describes each point to which you can apply a firewall filter. For each application point, the table describes the types of firewall filters supported at that point, the router (or switch) hierarchy level at which the filter can be applied, and any platform-specific limitations.

Table 29: Stateless Firewall Filter Configuration and Application Summary

Filter Type	Application Point	Restrictions

Stateless firewall filter
 	
Configure by including the **filter** ***filter-name*** statement the **[edit firewall]** hierarchy level:
 	
filter *filter-name*;
 	
NOTE: If you do not include the **family** statement, the firewall filter processes IPv4 traffic by default.
	Logical interface
	
	Apply at the **[edit interfaces *interface-name* unit *unit-number* family inet]** hierarchy level by including the **input *filter-name*** or **output *filter-name*** statements:
	
	filter {
	input *filter-name*;
	output *filter-name*;
	}
	
	NOTE: A filter configured with the implicit **inet** protocol family cannot be included in an input filter list or an output filter list.
	
	NOTE: On T4000 Type 5 FPCs, a filter attached at the Layer 2 application point (that is, at the logical interface level) is unable to match with the forwarding class of a packet that is set by a Layer 3 classifier such as DSCP, DSCP V6, **inet-precedence**, and **mpls-exp**.
		Supported on the following routers:
		
		• T Series routers
		• M320 routers
		• M7i routers with the enhanced CFEB (CFEB-e)
		• M10i routers with the enhanced CFEB-e
		
		Also supported on the following Modular Port Concentrators (MPCs) on MX Series routers:
		
		• 10-Gigabit Ethernet MPC
		• 60-Gigabit Ethernet Queuing MPC
		• 60-Gigabit Ethernet Enhanced Queuing MPC
		• 100-Gigabit Ethernet MPC
		
		• Also supported on EX Series switches

Stateless firewall filter
 	
Configure at the **[edit firewall family *family-name*]** hierarchy level by including the following statement:
 	
filter *filter-name*;
 	
The ***family-name*** can be any of the following protocol families:
 	
• **any**
• **bridge**
• **ethernet-switching**
• **ccc**
• **inet**
• **inet6**
• **mpls**
• **vpls**
	Protocol family on a logical interface
	
	Apply at the **[edit interfaces *interface-name* unit *unit-number* family *family-name*]** hierarchy level by, including the **input**, **input-list**, **output**, or **output-list** statements:
	
	filter {
	input *filter-name*;
	input-list [*filter-names*];
	output *filter-name*;
	output-list [*filter-names*];
	}
		The protocol family **bridge** is supported only on MX Series routers.

Stateless firewall filter	**Routing Engine loopback interface**	

Table 29: Stateless Firewall Filter Configuration and Application Summary *(continued)*

Filter Type	Application Point	Restrictions

Service filter
 	
Configure at the **[edit firewall family (inet | inet6)]** hierarchy level by including the following statement:
 	
service-filter *service-filter-name*;
	Family inet or inet6 on a logical interface
	
	Apply at the **[edit interfaces *interface-name* unit *unit-number* family (inet | inet6)]** hierarchy level by using the **service-set** statement to apply a service filter as an input or output filter to a service set:
	
	service {
	input {
	service-set *service-set-name* service-filter *filter-name;*
	}
	output {
	service-set *service-set-name* service-filter *filter-name;*
	}
	}
	
	Configure a service set at the **[edit services]** hierarchy level by including the following statement:
	
	service-set *service-set-name*;
		Supported only on Adaptive Services (AS) and Multiservices (MS) PICs.

Postservice filter
 	
Configure at the **[edit firewall family (inet | inet6)]** hierarchy level by including the following statement:
 	
service-filter *service-filter-name*;
	Family inet or inet6 on a logical interface
	
	Apply at the **[edit interfaces *interface-name* unit *unit-number* family (inet | inet6)]** hierarchy level by including the **post-service-filter** statement to apply a service filter as an input filter:
	
	service {
	input {
	post-service-filter *filter-name*;
	}
	}
		A postservice filter is applied to traffic returning to the services interface after service processing. The filter is applied only if a service set is configured and selected.

Simple filter Configure at the **[edit firewall family inet]** hierarchy level by including the following statement: simple-filter *filter-name*	**Family inet on a logical interface** Apply at the **[edit interfaces *interface-name* unit *unit-number* family inet]** hierarchy level by including the following statement: simple-filter *simple-filter-name*;	Simple filters can only be applied as input filters. Supported on the following platforms only: • Gigabit Ethernet intelligent queuing (IQ2) PICs on the M120, M320, and T Series routers. • Enhanced Queuing Dense Port Concentrators (EQ DPC) on MX Series routers (and EX Series switches).

Table 29: Stateless Firewall Filter Configuration and Application Summary *(continued)*

Filter Type	Application Point	Restrictions

Reverse packet forwarding (RPF) check filter
 	
Configured at the **[edit firewall family (inet | inet6)]** hierarchy level by including the following statement:
 	
filter *filter-name*;
	Family inet or inet6 on a logical interface
	
	Apply at the **[edit interfaces *interface-name* unit *unit-number* family (inet | inet6)]** hierarchy level by including the following statement:
	
	rpf-check fail-filter *filter-name*
	
	to apply the stateless firewall filter as an RPF check filter.
	
	rpf-check {
	fail-filter *filter-name*;
	mode loose;
	}
		Supported on MX Series routers and EX Series switches only.

Related Documentation

- Stateless Firewall Filter Components on page 561
- Supported Standards for Filtering on page 587

How Standard Firewall Filters Evaluate Packets

This topic covers the following information:

Firewall Filter Packet Evaluation Overview

The following sequence describes how the device evaluates a packet entering or exiting an interface if the input or output traffic at a device interface is associated with a firewall filter. Packet evaluation proceeds as follows:

1. The device evaluates the packet against the terms in the firewall filter sequentially, beginning with the first term in the filter.
   - If the packet matches all the conditions specified in a term, the device performs all the actions specified in that term.
   - If the packet does not match all the conditions specified in a term, the device proceeds to the next term in the filter (if a subsequent term exists) and evaluates the packet against that term.

- If the packet does not match any term in the firewall filter, the device implicitly discards the packet.

2. Unlike service filters and simple filters, firewall filters support the **next term** action, which is neither a terminating action nor a nonterminating action but a flow control action.

   - If the matched term includes the **next term** action, the device continues evaluation of the packet at the next term within the firewall filter.
   - If the matched term does not include the **next term** action, evaluation of the packet against the given firewall filter ends at this term. The device does not evaluate the packet against any subsequent terms in this filter.

   A maximum of 1024 **next term** actions are supported per firewall filter configuration. If you configure a firewall filter that exceeds this limit, your candidate configuration results in a commit error.

3. The device stops evaluating a packet against a given firewall filter when either the packet matches a term without the **next term** action or the packet fails to match the last term in the firewall filter.

4. If a local packet arrives at a router interface that is associated with an ingress firewall filter, the filter evaluates the packet twice. The first evaluation occurs in the Packet Forwarding Engine, which is the central processing element of the router's forwarding plane, and the second evaluation occurs in the Routing Engine, which runs the router's control plane software.

   NOTE: **Local packets--chunks of data that are destined for or sent by the router itself--usually contain routing protocol data, data for IP services such as Telnet or SSH, and data for administrative protocols such as the Internet Control Message Protocol (ICMP).**

   If the first evaluation of the firewall filter modifies the incoming local packet or packet context values, the second evaluation of the firewall filter is based on the updated packet or packet context values.

   For example, suppose that the filter includes a match condition based on the forwarding class or loss priority value associated with the packet and that the filter includes an action that modifies the forwarding class or loss priority value associated with the packet. If an ingress local packet arrives at an associated interface and the filter evaluation in the Packet Forwarding Engine modifies (rather than drops) the packet, then the filter evaluation in the Routing Engine is based on the modified packet context (rather than the original packet context).

Packet Evaluation at a Single Firewall Filter

Table 30 on page 572 describes packet-filtering behaviors at a device interface associated with a single firewall filter.

Table 30: Packet Evaluation at a Single Firewall Filter

Firewall Filter Event	Action	Subsequent Action
The firewall filter term does not specify any match conditions.	The term matches all packets by default, and so the device performs the actions specified by that term.	If the term actions include the **next term** action, the device continues evaluation of the packet against the next term within the firewall filter (if a subsequent term exists).
The packet matches all conditions specified by the firewall filter term.	The device performs the actions specified by that term.	If the term actions include the **next term** action, the device continues evaluation of the packet against the next term within the firewall filter (if a subsequent term exists).
The packet matches all conditions specified by the firewall filter term, but the term does not specify any actions.	The device implicitly accepts the packet.	If the term actions include the **next term** action, the device continues evaluation of the packet against the next term within the firewall filter (if a subsequent term exists).
The packet does not match all conditions specified by the firewall filter term.	The device does not perform the actions specified by that term.	The device continues evaluation of the packet against the next term within the filter (if a subsequent term exists).

The packet does not match any term in the filter	The device implicitly discards the packet
	
	Every firewall filter configuration includes an implicit **discard** action at the end of the filter. This implicit terminating action is equivalent to including the following example term **t_explicit_discard** as the final term in the firewall filter:
	
	term t_explicit_discard {
	  then discard;
	}	

Best Practice: Explicitly Accept Any Traffic That Is Not Specifically Discarded

You might want a firewall filter to accept any traffic that the filter does not specifically discard. In this case, we recommend that you configure the firewall filter with a final term that specifies the **accept** terminating action.

In the following example snippet, configuring the **t_allow_all_else** term as the final term in the firewall filter explicitly configures the firewall filter to accept any traffic that the filter did not specifically discard :

```
term t_allow_all_else {
  then accept;
}
```

Following this best practice can simplify troubleshooting of the firewall filter.

Best Practice: Explicitly Reject Any Traffic That Is Not Specifically Accepted

On the other hand, you might want a firewall filter to reject any traffic that the firewall filter does not specifically accept. In this case, we recommend that you configure the firewall filter with a final term that specifies the **reject** terminating action.

In the following example snippet, configuring the **t_deny_all_else** term as the final term in the firewall filter explicitly configures the firewall filter to reject any traffic that the filter did not specifically accept:

```
term t_deny_all_else {
   then reject;
}
```

Following this best practice can simplify troubleshooting of the firewall filter.

Multiple Firewall Filters Attached to a Single Interface

On supported device interfaces, you can attach multiple firewall filters to a single interface. For more information, see “Understanding Multiple Firewall Filters Applied as a List” on page 844.

NOTE: On supported interfaces, you can attach a protocol-independent (**family any**) firewall filter and a protocol-specific (**family inet** or **family inet6**) firewall filter to the same interface. The protocol-independent firewall filter executes first. For more information, see “Guidelines for Applying Standard Firewall Filters” on page 581.

Single Firewall Filter Attached to Multiple Interfaces

On supported interfaces, you can associate a single firewall filter with multiple interfaces, and Junos OS creates an *interface-specific instance* of that firewall filter for each associated interface.

- Junos OS associates each interface-specific instantiation of a firewall filter with a system-generated, interface-specific name.
- For any **count** actions in the filter terms, the Packet Forwarding Engine maintains separate, interface-specific counters, and Junos OS associates each counter with a system-generated, interface-specific name.
- For any **policer** actions in the filter terms, Junos OS creates separate, interface-specific instances of the policer actions.

For more information, see “Interface-Specific Firewall Filter Instances Overview” on page 859.

Related Documentation

- Firewall Filter Match Conditions for Protocol-Independent Traffic on page 615
- Firewall Filter Terminating Actions on page 680
- How Service Filters Evaluate Packets on page 906

- How Simple Filters Evaluate Packets on page 927
- Guidelines for Configuring Firewall Filters on page 576
- Understanding How to Use Standard Firewall Filters on page 559

Understanding Firewall Filter Fast Lookup Filter

In order to enhance the speed at which specific firewall filters are processed, you can use the filter block hardware available on certain modular port concentrators (MPCs). See the MX Series Interface Module Reference manual for details.This hardware allows for an increase in the number of firewall filter operations per second that can be accomplished.

Using the **fast-lookup-filter** option in environments with hundreds or thousands of terms per filter can increase performance of those filters by utilizing the filter block hardware.

There are 4096 hardware filters available per MPC. The number of firewall filters that can be installed in hardware depends on the number of terms in each filter. One hardware filter is needed for every group of 255 terms in a firewall filter. The total number of terms supported per firewall filter is 8000. However, attaching a given firewall filter with less than 256 terms to multiple interfaces will only result in one instance of that firewall filter being installed in the filter block. This is true for interface-specific filters as well as for filter lists.

You designate specific firewall filters to be processed in the filter block hardware by including the **fast-lookup-filter** option when configuring the firewall.

When this option is used, firewall parameters are stored in the filter block hardware which accelerates the lookup process. **fast-lookup-filter** is only available for the inet and inet6 protocol families. The match conditions are limited to 5-tuples: **protocol**, **source-address**, **destination-address**, **source-port**, and **destination-port**.

Ranges, prefix lists, and the except keyword are supported within the firewall filters and terms when using this option.

NOTE: Firewall filters that are configured using the **fast-lookup-filter** option are not optimized by the firewall compiler.

Related Documentation

- *MX Series Interface Module Reference*
- *fast-lookup-filter*

Multifield Classifier for Ingress Queuing on MX Series Routers with MPC

Beginning with Junos OS Release 16.1, the multifield classifier for ingress queuing is an implementation point for firewall filters configured with specific traffic shaping actions. These filters allow you to set the forwarding class and packet loss priority for packets, or drop the packets prior to ingress queue selection. The filters are applied as ingress queue filters. Class-of-service (CoS) commands can then be used to select ingress queue, set rate limiting and so forth.

Firewall filters configured at the protocol family level are able to distinguish specific types of traffic from other types by matching on multiple fields within the packet header. The number and types of matches available depend on which protocol family is used in the filter. Before the introduction of the ingress queuing filter, these firewall filters could only be applied to traffic after the ingress queue had been selected based solely on the behavior aggregate (BA). With the introduction of the ingress queuing filter, firewall filters can be used to set forwarding classification and packet loss priority based on multiple fields within the packet header prior to forwarding queue selection. CoS functions provide traffic classification options and the ability to assign that classified traffic to specific forwarding queues.

NOTE: Ingress queuing filters are only available when the traffic manager mode is set to **ingress-and-egress** at the **[edit chassis fpc *fpc-id* pic *pic-id* traffic-manager mode]** hierarchy level.

The **ingress-queuing-filter** configuration statement is used at the **[edit interfaces *interface-name* unit *unit-number* family *family-name*]** hierarchy level to designate a previously configured firewall filter to be used as an ingress queuing filter. The following list shows which protocol families are compatible with the **ingress-queuing-filter** statement:

- **bridge**
- **ccc**
- **inet**
- **inet6**
- **mpls**
- **vpls**

The named firewall filter is a normal firewall filter that must be configured with at least one of the following actions: **accept, discard, forwarding-class,** and **loss-priority**.

Release History Table

Release	Description
16.1	Beginning with Junos OS Release 16.1, the multifield classifier for ingress queuing is an implementation point for firewall filters configured with specific traffic shaping actions.

Related Documentation

- *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*
- **ingress-queuing-filter on page 1208**
- Example: Configuring a Filter for Use as an Ingress Queuing Filter on page 741

Guidelines for Configuring Firewall Filters

This topic covers the following information:

Statement Hierarchy for Configuring Firewall Filters

To configure a standard firewall filter, you can include the following statements. For an IPv4 standard firewall filter, the **family inet** statement is optional. For an IPv6 standard firewall filter, the **family inet6** statement is mandatory.

```
firewall {
  family family-name {
    filter filter-name {
      accounting-profile name;
      instance-shared;
      interface-specific;
      physical-interface-filter;
      term term-name {
        filter filter-name;
      }
      term term-name {
        from {
          match-conditions;
          ip-version ip-version {
            match-conditions;
            protocol (tcp | udp) {
              match conditions;
            }
          }
        }
        then {
```



```
                    actions;
                }
            }
        }
    }
}
```

You can include the firewall configuration at one of the following hierarchy levels:

- **[edit]**
- **[edit logical-systems *logical-system-name*]**

NOTE: For stateless firewall filtering, you must allow the output tunnel traffic through the firewall filter applied to input traffic on the interface that is the next-hop interface toward the tunnel destination. The firewall filter affects only the packets exiting the router (or switch) by way of the tunnel.

Firewall Filter Protocol Families

A firewall filter configuration is specific to a particular protocol family. Under the **firewall** statement, include one of the following statements to specify the protocol family for which you want to filter traffic:

- **family any**—To filter protocol-independent traffic.
- **family inet**—To filter Internet Protocol version 4 (IPv4) traffic.
- **family inet6**—To filter Internet Protocol version 6 (IPv6) traffic.
- **family mpls**—To filter MPLS traffic.
- **family vpls**—To filter virtual private LAN service (VPLS) traffic.
- **family ccc**—To filter Layer 2 circuit cross-connection (CCC) traffic.
- **family bridge**—To filter Layer 2 bridging traffic for MX Series 3D Universal Edge Routers only.
- **family ethernet-switching**—To filter Layer 2 (Ethernet) traffic.

The **family *family-name*** statement is required only to specify a protocol family other than IPv4. To configure an IPv4 firewall filter, you can configure the filter at the **[edit firewall]** hierarchy level without including the **family inet** statement, because the **[edit firewall]** and **[edit firewall family inet]** hierarchy levels are equivalent.

NOTE: For bridge family filter, the *ip-protocol* match criteria is supported only for IPv4 and not for IPv6. This is applicable for line cards that support the Junos Trio chipset such as the MX 3D MPC line cards.

Firewall Filter Names and Options

Under the **family *family-name*** statement, you can include **filter *filter-name*** statements to create and name firewall filters. The filter name can contain letters, numbers, and hyphens (-) and be up to 64 characters long. To include spaces in the name, enclose the entire name in quotation marks (" ").

At the **[edit firewall family *family-name* filter *filter-name*]** hierarchy level, the following statements are optional:

- **accounting-profile**
- **instance-shared** (MX Series routers with Modular Port Concentrators (MPCS) only)
- **interface-specific**
- **physical-interface-filter**

Firewall Filter Terms

Under the **filter *filter-name*** statement, you can include **term *term-name*** statements to create and name filter terms.

- You must configure at least one term in a firewall filter.
- You must specify a unique name for each term within a firewall filter. The term name can contain letters, numbers, and hyphens (-) and can be up to 64 characters long. To include spaces in the name, enclose the entire name in quotation marks (" ").
- The order in which you specify terms within a firewall filter configuration is important. Firewall filter terms are evaluated in the order in which they are configured. By default, new terms are always added to the end of the existing filter. You can use the **insert** configuration mode command to reorder the terms of a firewall filter.

At the **[edit firewall family *family-name* filter *filter-name* term *term-name*]** hierarchy level, the **filter *filter-name*** statement is not valid in the same term as **from** or **then** statements. When included at this hierarchy level, the **filter *filter-name*** statement is used to *nest* firewall filters.

Firewall Filter Match Conditions

Firewall filter match conditions are specific to the type of traffic being filtered.

With the exception of MPLS-tagged IPv4 or IPv6 traffic, you specify the term's match conditions under the **from** statement. For MPLS-tagged IPv4 traffic, you specify the term's IPv4 address-specific match conditions under the **ip-version ipv4** statement and the term's IPv4 port-specific match conditions under the **protocol (tcp | udp)** statement.

For MPLS-tagged IPv6 traffic, you specify the term's IPv6 address-specific match conditions under the **ip-version ipv6** statement and the term's IPv6 port-specific match conditions under the **protocol (tcp | udp)** statement.

Table 31 on page 579 describes the types of traffic for which you can configure firewall filters.

Table 31: Firewall Filter Match Conditions by Protocol Family

Traffic Type	Hierarchy Level at Which Match Conditions Are Specified
Protocol-independent	**[edit firewall family any filter *filter-name* term *term-name*]** For the complete list of match conditions, see "Firewall Filter Match Conditions for Protocol-Independent Traffic" on page 615.
IPv4	**[edit firewall family inet filter *filter-name* term *term-name*]** For the complete list of match conditions, see "Firewall Filter Match Conditions for IPv4 Traffic" on page 589.
IPv6	**[edit firewall family inet6 filter *filter-name* term *term-name*]** For the complete list of match conditions, see "Firewall Filter Match Conditions for IPv6 Traffic" on page 629.
MPLS	**[edit firewall family mpls filter *filter-name* term *term-name*]** For the complete list of match conditions, see "Firewall Filter Match Conditions for MPLS Traffic" on page 639.
IPv4 addresses in MPLS flows	**[edit firewall family mpls filter *filter-name* term *term-name* ip-version ipv4]** For the complete list of match conditions, see "Firewall Filter Match Conditions for MPLS-Tagged IPv4 or IPv6 Traffic" on page 640.
IPv4 ports in MPLS flows	**[edit firewall family mpls filter *filter-name* term *term-name* ip-version ipv4 protocol (tcp | udp)]** For the complete list of match conditions, see "Firewall Filter Match Conditions for MPLS-Tagged IPv4 or IPv6 Traffic" on page 640.
IPv6 addresses in MPLS flows	**[edit firewall family mpls filter *filter-name* term *term-name* ip-version ipv6]** For the complete list of match conditions, see "Firewall Filter Match Conditions for MPLS-Tagged IPv4 or IPv6 Traffic" on page 640.
IPv6 ports in MPLS flows	**[edit firewall family mpls filter *filter-name* term *term-name* ip-version ipv6 protocol (tcp | udp)]** For the complete list of match conditions, see "Firewall Filter Match Conditions for MPLS-Tagged IPv4 or IPv6 Traffic" on page 640.
VPLS	**[edit firewall family vpls filter *filter-name* term *term-name*]** For the complete list of match conditions, see "Firewall Filter Match Conditions for VPLS Traffic" on page 643.
Layer 2 CCC	**[edit firewall family ccc filter *filter-name* term *term-name*]** For the complete list of match conditions, see "Firewall Filter Match Conditions for Layer 2 CCC Traffic" on page 654.

Table 31: Firewall Filter Match Conditions by Protocol Family *(continued)*

Traffic Type	Hierarchy Level at Which Match Conditions Are Specified
Layer 2 Bridging (MX Series routers and EX Series switches only)	**[edit firewall family bridge filter *filter-name* term *term-name*]** **[edit firewall family ethernet-switching filter *filter-name* term *term-name*]** (for EX Series switches only) For the complete list of match conditions, see "Firewall Filter Match Conditions for Layer 2 Bridging Traffic" on page 658.

If you specify an IPv6 address in a match condition (the **address**, **destination-address**, or **source-address** match conditions), use the syntax for text representations described in RFC 4291, *IP Version 6 Addressing Architecture*. For more information about IPv6 addresses, see *IPv6 Overview* and *Supported IPv6 Standards*.

Firewall Filter Actions

Under the **then** statement for a firewall filter term, you can specify the actions to be taken on a packet that matches the term.

Table 32 on page 580 summarizes the types of actions you can specify in a firewall filter term.

Table 32: Firewall Filter Action Categories

Type of Action	Description	Comment
Terminating	Halts all evaluation of a firewall filter for a specific packet. The router (or switch) performs the specified action, and no additional terms are used to examine the packet. You can specify only one *terminating action* in a firewall filter term. You can, however, specify one terminating action with one or more *nonterminating actions* in a single term. For example, within a term, you can specify **accept** with **count** and **syslog**. Regardless of the number of terms that contain terminating actions, once the system processes a terminating action within a term, processing of the entire firewall filter halts.	See "Firewall Filter Terminating Actions" on page 680.
Nonterminating	Performs other functions on a packet (such as incrementing a counter, logging information about the packet header, sampling the packet data, or sending information to a remote host using the system log functionality), but any additional terms are used to examine the packet.	All nonterminating actions include an implicit accept action. This accept action is carried out if no other terminating action is configured in the same term. See "Firewall Filter Nonterminating Actions" on page 673.

Table 32: Firewall Filter Action Categories *(continued)*

Type of Action	Description	Comment
Flow control	For standard firewall filters only, the **next term** action directs the router (or switch) to perform configured actions on the packet and then, rather than terminate the filter, use the next term in the filter to evaluate the packet. If the **next term** action is included, the matching packet is evaluated against the next term in the firewall filter. Otherwise, the matching packet is not evaluated against subsequent terms in the firewall filter. For example, when you configure a term with the nonterminating action **count**, the term's action changes from an implicit **discard** to an implicit **accept**. The **next term** action forces the continued evaluation of the firewall filter.	You cannot configure the **next term** action with a terminating action in the same filter term. However, you can configure the next term action with another nonterminating action in the same filter term. A maximum of 1024 **next term** actions are supported per standard firewall filter configuration. If you configure a standard firewall filter that exceeds this limit, your candidate configuration results in a commit error.

Related Documentation

- Guidelines for Applying Standard Firewall Filters on page 581
- Understanding How to Use Standard Firewall Filters on page 559

Guidelines for Applying Standard Firewall Filters

This topic covers the following information:

Applying Firewall Filters Overview

You can apply a standard firewall filter to a loopback interface on the router or to a physical or logical interface on the router. You can apply a firewall filter to a single interface or to multiple interfaces on the router. Table 33 on page 581 summarizes the behavior of firewall filters based on the point to which you attach the filter.

Table 33: Firewall Filter Behavior by Filter Attachment Point

Filter Attachment Point	Filter Behavior
Loopback interface	The router's loopback interface, **lo0**, is the interface to the Routing Engine and carries no data packets. When you apply a firewall filter to the loopback interface, the filter evaluates the local packets received or transmitted by the Routing Engine. NOTE: • ACX5048 and ACX5096 routers do not support the evaluation of packets transmitted by the Routing engine for loopback interface filter.
Physical interface or logical interface	When you apply a filter to a physical interface on the router or to a logical interface (or member of an aggregated Ethernet bundle defined on the interface), the filter evaluates all data packet that pass through that interface.

Table 33: Firewall Filter Behavior by Filter Attachment Point *(continued)*

Filter Attachment Point	Filter Behavior
Multiple interfaces	You can use the same firewall filter one or more times. On M Series routers, except the M120 and M320 routers, if you apply a firewall filter to multiple interfaces, the filter acts on the sum of traffic entering or exiting those interfaces. On T Series, M120, M320, and MX Series routers, interfaces are distributed among multiple packet-forwarding components. On these routers, you can configure firewall filters and service filters that, when applied to multiple interfaces, act on the individual traffic streams entering or exiting each interface, regardless of the sum of traffic on the multiple interfaces. For more information, see "Interface-Specific Firewall Filter Instances Overview" on page 859.
Single interface with protocol-independent and protocol-specific firewall filters attached	For interfaces hosted on the following hardware only, you can attach a protocol-independent (**family any**) firewall filter and a protocol-specific (**family inet** or **family inet6**) firewall filter simultaneously. The protocol-independent firewall executes first. • ACX Series Universal Access Routers • Flexible PIC Concentrators (FPCs) in M7i and M10i Multiservice Edge Routers • Modular Interface Cards (MICs) and Modular Port Concentrators (MPCs) in MX Series 3D Universal Edge Routers • T Series Core Routers NOTE: Interfaces hosted on the following hardware do not support protocol-independent firewall filters: • Forwarding Engine Boards (FEBs) in M120 routers • Enhanced III FPCs in M320 routers • FPC2 and FPC3 modules in MX Series routers • Dense Port Concentrators (DPCs) in MX Series routers • PTX Series Packet Transport Routers

Statement Hierarchy for Applying Firewall Filters

To apply a standard firewall filter to a logical interface, configure the filter statement for the logical interface defined under either the **[edit]** or **[edit logical-systems *logical-system-name*]** hierarchy level. Under the **filter** statement, you can include one or more of the following statements: **group *group-number***, **input *filter-name***, **input-list *filter-name***, **output *filter-name***, or **output-list *filter-name***. The hierarchy level at which you attach the **filter** statement depends on the filter type and device type you are configuring.

Protocol-Independent Firewall Filters on MX Series Routers

To apply a protocol-independent firewall filter to a logical interface on an MX Series router, configure the **filter** statement *directly* under the logical unit:

```
interfaces {
  interface-name {
    unit logical-unit-number {
      filter {
        group group-number;
        input filter-name;
```



```
            input-list [ filter-names ];
            output filter-name;
            output-list [ filter-names ];
          }
        }
      }
    }
```

All Other Firewall Filters on Logical Interfaces

To apply a standard firewall filter to a logical interface for all cases *other than* a protocol-independent filter on an MX Series router, configure the **filter** statement under the protocol family:

```
interfaces {
  interface-name {
    unit logical-unit-number {
      family family-name {
        ...
        filter {
          group group-number;
          input filter-name;
          input-list [ filter-names ];
          output filter-name;
          output-list [ filter-names ];
        }
      }
    }
  }
}
```

Restrictions on Applying Firewall Filters

Number of Input and Output Filters Per Logical Interface

Input filters—Although you can use the same filter multiple times, you can apply only one input filter or one input filter list to an interface.

- To specify a single firewall filter to be used to evaluate packets received on the interface, include the **input *filter-name*** statement in the **filter** stanza.
- To specify an ordered list of firewall filters to be used to evaluate packets received on the interface, include the **input-list [*filter-names*]** statement in the **filter** stanza. You can specify up to 16 firewall filters for the filter input list.

Output filters—Although you can use the same filter multiple times, you can apply only one output filter or one output filter list to an interface.

- To specify a single firewall filter to be used to evaluate packets transmitted on the interface, include the **output *filter-name*** statement in the **filter** stanza.

- To specify an ordered list of firewall filters to be used to evaluate packets transmitted on the interface, include the **output-list [*filter-names*]** statement in the **filter** stanza. You can specify up to 16 firewall filters in a filter output list.

MPLS and Layer 2 CCC Firewall Filters in Lists

The **input-list** *filter-names* and **output-list** *filter-names* statements for firewall filters for the **ccc** and **mpls** protocol families are supported on all interfaces with the exception of the following:

- Management interfaces and internal Ethernet interfaces (**fxp** or **em0**)
- Loopback interfaces (**lo0**)
- USB modem interfaces (**umd**)

Layer 2 CCC Firewall Filters on MX Series Routers and EX Series Switches

Only on MX Series routers and EX Series switches, you cannot apply a Layer 2 CCC stateless firewall filter (a firewall filter configured at the **[edit firewall filter family ccc]** hierarchy level) as an output filter. On MX Series routers and EX Series switches, firewall filters configured for the **family ccc** statement can be applied only as input filters.

Related Documentation

- family (Firewall) on page 1248
- *family (Interfaces)*
- filter (Applying to a Logical Interface) on page 1251
- filter (Configuring) on page 1252
- Guidelines for Configuring Firewall Filters on page 576
- Understanding How to Use Standard Firewall Filters on page 559

Firewall and Policing Differences Between PTX Series Packet Transport Routers and T Series Matrix Routers

This topic provides a list of firewall and policier features available on PTX Packet Transport Routers and compares them with firewall and policing features on T Series routers.

Firewall Filters

Junos OS firewall and policing software on PTX Series Packet Transport Routers supports IPv4 filters, IPv6 filters, MPLS filters, CCC filters, interface policing, LSP policing, MAC filtering, ARP policing, L2 policing, and other features. Exceptions are noted below.

- PTX Series Packet Transport Routers do not support:
  - Egress Forwarding Table Filters
  - Forwarding Table Filters for MPLS/CCC

  - Family VPLS

- PTX Series Packet Transport Routers do not support nested firewall filters. The **filter** statement at the **[edit firewall family *family-name* filter *filter-name* term *term-name*]** hierarchy level is disabled.
- Because no service PICs are present in PTX Series Packet Transport Routers, service filters are not supported for both IPv4 and IPv6 traffic. The **service-filter** statement at **[edit firewall family (inet | inet6)]** hierarchy level is disabled.
- The PTX Series Packet Transport Routers exclude simple filters. These filters are supported on Gigabit Ethernet intelligent queuing (IQ2) and Enhanced Queuing Dense Port Concentrator (EQ DPC) interfaces only. The **simple-filter** statement at the **[edit firewall family inet)]** hierarchy level is disabled.
- Physical interface filtering is not supported. The **physical-interface-filter** statement at the **[edit firewall family *family-name* filter *filter-name*]** hierarchy level is disabled.
- The prefix action feature is not supported on PTX Series Packet Transport Routers. The **prefix-action** statement at **[edit firewall family inet]** hierarchy level is disabled.
- On T Series routers, you can collect a variety of information about traffic passing through the device by setting up one or more accounting profiles that specify some common characteristics of the data. The PTX Series Packet Transport Routers do not support accounting configurations for firewall filters. The **accounting-profile** statement at the **[edit firewall family *family-name* filter *filter-name*]** hierarchy level is disabled.
- The **reject** action is not supported on the loopback (**lo0**) interface. If you apply a filter to the **lo0** interface and the filter includes a **reject** action, an error message appears.
- PTX Series Packet Transport Routers do not support aggregated ethernet logical interface match conditions. However, child link interface matching is supported.
- PTX Series Packet Transport Routers displays both counts if two different terms in a filter have the same match condition but they have different counts. T Series routers display one count only.
- PTX Series Packet Transport Routers do not have separate policer instances when a filter is bound to multiple interfaces. Use the **interface-specific** configuration statement to create the configuration.
- On PTX Series Packet Transport Routers, when an ingress interface has CCC encapsulation, packets coming in through the ingress CCC interface will not be processed by the egress filters.
- For CCC encapsulation, the PTX Series Packet Transport Routers append an extra 8 bytes for egress Layer 2 filtering. The T Series routers do not. Therefore, egress counters on PTX Series Packet Transport Routers show an extra eight bytes for each packet which impacts policer accuracy.
- On PTX Series Packet Transport Routers, output for the **show pfe statistics traffic** CLI command includes the packets discarded by DMAC and SMAC filtering. On T Series routers, the command output does not include these discarded packets because MAC filters are implemented in the PIC and not in the FPC.

- The last-fragment packet that goes through a PTX firewall cannot be matched by the **is-fragment** matching condition. This feature is supported on T Series routers.

  A possible workaround on PTX Series Packet Transport Routers is to configure two separate terms with same the actions: one term contains a match to **is-fragment** and the other term contains a match to **fragment-offset -except 0**.

- On PTX Series Packet Transport Routers, MAC pause frames are generated when packet discards exceed 100 Mbps. This occurs only for frame sizes that are less than 105 bytes.

Traffic Policiers

Junos OS firewall and policing software on PTX Series Packet Transport Routers supports IPv4 filters, IPv6 filters, MPLS filters, CCC filters, interface policing, LSP policing, MAC filtering, ARP policing, L2 policing, and other features. Exceptions are noted below.

- PTX Series Packet Transport Routers support ARP policing. T Series routers do not.
- PTX Series Packet Transport Routers do not support LSP policing.
- PTX Series Packet Transport Routers do not support the **hierarchical-policer** configuration statement. .
- PTX Series Packet Transport Routers do not support the **interface-set** configuration statement. This statement groups a number of interfaces into a single, named interface set.
- PTX Series Packet Transport Routers do not support the following policer types for both normal policers and three-color policers:
  - **logical-bandwidth-policer** — Policer uses logical interface bandwidth.
  - **physical-interface-policer** — Policer is a physical interface policer.
  - **shared-bandwidth-policer** — Share policer bandwidth among bundle links.
- When a policer action and forwarding-class, loss-priority actions are configured within the same rule (a *Multifield Classification*), the PTX Series Packet Transport Routers work differently than T Series routers. As shown below, you can configure two rules in the filter to make the PTX filter behave the same as the T Series filter:

  PTX Series configuration:

  ```
  rule-1 {
     match: {x, y, z}
     action: {forwarding-class, loss-prio, next}
  }
     rule-2 {
        match: {x, y, z}
        action: {policer}
     }
  ```

  T Series configuration:

  ```
  rule-1 {
     match: {x, y, z}
     action: {forwarding-class, loss-prio, policer}
  ```



```
}
```

Related Documentation

- *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*

Supported Standards for Filtering

The Junos OS supports the following RFCs related to filtering:

- RFC 792, *Internet Control Message Protocol*
- RFC 2460, *Internet Protocol, Version 6 (IPv6)*
- RFC 2474, *Definition of the Differentiated Services (DS) Field*
- RFC 2475, *An Architecture for Differentiated Services*
- RFC 2597, *Assured Forwarding PHB Group*
- RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*
- RFC 4291, *IP Version 6 Addressing Architecture*
- RFC 4443, *Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification*

NOTE: ACX Series routers do not support RFC 2460, RFC 4291, and RFC 4443 standards.

Related Documentation

- Firewall Filters Overview on page 555
- Service Filter Overview on page 905
- Simple Filter Overview on page 927
- Firewall Filters in Logical Systems Overview on page 787

CHAPTER 15

Firewall Filter Match Conditions and Actions

Firewall Filter Match Conditions for IPv4 Traffic

You can configure a firewall filter with match conditions for Internet Protocol version 4 (IPv4) traffic (**family inet**).

NOTE: For MX Series routers with MPCs, you need to initialize certain new firewall filters by walking the corresponding SNMP MIB, for example, **show snmp mib walk *name* ascii**. This forces Junos to learn the filter counters and ensure that the filter statistics are displayed. This guidance applies to all enhanced mode firewall filters, filters with flexible conditions, and filters with the certain terminating actions. See those topics, listed under Related Documentation, for details.

Table 34 on page 590 describes the ***match-conditions*** you can configure at the **[edit firewall family inet filter *filter-name* term *term-name* from]** hierarchy level.

Table 34: Firewall Filter Match Conditions for IPv4 Traffic

Match Condition	Description
address *address* [except]	Match the IPv4 source or destination address field unless the **except** option is included. If the option is included, do not match the IPv4 source or destination address field. NOTE: This match condition is not supported on PTX1000 routers.
ah-spi *spi-value*	(M Series routers, except M120 and M320) Match the IPsec authentication header (AH) security parameter index (SPI) value. NOTE: This match condition is not supported on PTX series routers.
ah-spi-except *spi-value*	(M Series routers, except M120 and M320) Do not match the IPsec AH SPI value. NOTE: This match condition is not supported on PTX series routers.
apply-groups	Specify which groups to inherit configuration data from. You can specify more than one group name. You must list them in order of inheritance priority. The configuration data in the first group takes priority over the data in subsequent groups.
apply-groups-except	Specify which groups not to inherit configuration data from. You can specify more than one group name.
destination-address *address* [except]	Match the IPv4 destination address field unless the **except** option is included. If the option is included, do not match the IPv4 destination address field. You cannot specify both the **address** and **destination-address** match conditions in the same term. NOTE: The **except** option is not supported on PTX1000 routers.
destination-class *class-names*	Match one or more specified destination class names (sets of destination prefixes grouped together and given a class name). For more information, see "Firewall Filter Match Conditions Based on Address Classes" on page 614. NOTE: This match condition is not supported on PTX series routers.
destination-class-except *class-names*	Do not match one or more specified destination class names. For details, see the **destination-class** match condition. NOTE: This match condition is not supported on PTX series routers.

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
destination-port *number*	Match the UDP or TCP destination port field. You cannot specify both the **port** and **destination-port** match conditions in the same term. If you configure this match condition, we recommend that you also configure the **protocol udp** or **protocol tcp** match statement in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the following text synonyms (the port numbers are also listed): **afs** (1483), **bgp** (179), **biff** (512), **bootpc** (68), **bootps** (67), **cmd** (514), **cvspserver** (2401), **dhcp** (67), **domain** (53), **eklogin** (2105), **ekshell** (2106), **exec** (512), **finger** (79), **ftp** (21), **ftp-data** (20), **http** (80), **https** (443), **ident** (113), **imap** (143), **kerberos-sec** (88), **klogin** (543), **kpasswd** (761), **krb-prop** (754), **krbupdate** (760), **kshell** (544), **ldap** (389), **ldp** (646), **login** (513), **mobileip-agent** (434), **mobilip-mn** (435), **msdp** (639), **netbios-dgm** (138), **netbios-ns** (137), **netbios-ssn** (139), **nfsd** (2049), **nntp** (119), **ntalk** (518), **ntp** (123), **pop3** (110), **pptp** (1723), **printer** (515), **radacct** (1813), **radius** (1812), **rip** (520), **rkinit** (2108), **smtp** (25), **snmp** (161), **snmptrap** (162), **snpp** (444), **socks** (1080), **ssh** (22), **sunrpc** (111), **syslog** (514), **tacacs** (49), **tacacs-ds** (65), **talk** (517), **telnet** (23), **tftp** (69), **timed** (525), **who** (513), or **xdmcp** (177).
destination-port-except *number*	Do not match the UDP or TCP destination port field. For details, see the **destination-port** match condition.
destination-prefix-list *name* **[except]**	Match destination prefixes in the specified list unless the **except** option is included. If the option is included, do not match the destination prefixes in the specified list. Specify the name of a prefix list defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level.

dscp *number*	Match the Differentiated Services code point (DSCP). The DiffServ protocol uses the type-of-service (ToS) byte in the IP header. The most significant 6 bits of this byte form the DSCP. For more information, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*.
	
	Support was added for filtering on Differentiated Services Code Point (DSCP) and forwarding class for Routing Engine sourced packets, including IS-IS packets encapsulated in generic routing encapsulation (GRE). Subsequently, when upgrading from a previous version of Junos OS where you have both a class of service (CoS) and firewall filter, and both include DSCP or forwarding class filter actions, the criteria in the firewall filter automatically takes precedence over the CoS settings. The same is true when creating new configurations; that is, where the same settings exist, the firewall filter takes precedence over the CoS, regardless of which was created first.
	
	You can specify a numeric value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix.
	
	In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed):
	
	• RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*, defines one code point: **ef** (46).
	• RFC 2597, *Assured Forwarding PHB Group*, defines 4 classes, with 3 drop precedences in each class, for a total of 12 code points:
	  ◦ **af11** (10), **af12** (12), **af13** (14)
	  ◦ **af21** (18), **af22** (20), **af23** (22)
	  ◦ **af31** (26), **af32** (28), **af33** (30)
	  ◦ **af41** (34), **af42** (36), **af43** (38)

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description	
dscp-except *number*	Do not match on the DSCP number. For more information, see the **dscp** match condition.	
esp-spi *spi-value*	Match the IPsec encapsulating security payload (ESP) SPI value. Match on this specific SPI value. You can specify the ESP SPI value in hexadecimal, binary, or decimal form. NOTE: This match condition is not supported on PTX series routers.	
esp-spi-except *spi-value*	Match the IPsec ESP SPI value. Do not match on this specific SPI value. NOTE: This match condition is not supported on PTX series routers.	
first-fragment	Match if the packet is the first fragment of a fragmented packet. Do not match if the packet is a trailing fragment of a fragmented packet. The first fragment of a fragmented packet has a fragment offset value of **0**. This match condition is an alias for the bit-field match condition **fragment-offset 0** match condition. To match both first and trailing fragments, you can use two terms that specify different match conditions: **first-fragment** and **is-fragment**.	
flexible-match-mask *value*	**bit-length**	Length of the data to be matched in bits, not needed for string input (0..128)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-mask-name	Select a flexible match from predefined template field
	mask-in-hex	Mask out bits in the packet data to be matched
	match-start	Start point to match in packet
	prefix	Value data/string to be matched

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description	
flexible-match-range *value*	**bit-length**	Length of the data to be matched in bits (0..32)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-range-name	Select a flexible match from predefined template field
	match-start	Start point to match in packet
	range	Range of values to be matched
	range-except	Do not match this range of values
forwarding-class *class*	Match the forwarding class of the packet. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**. NOTE: This match condition is not supported on PTX1000 routers. For information about forwarding classes and router-internal output queues, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.	
forwarding-class-except *class*	Do not match the forwarding class of the packet. For details, see the **forwarding-class** match condition. NOTE: This match condition is not supported on PTX1000 routers.	
fragment-flags *number*	(Ingress only) Match the three-bit IP fragmentation flags field in the IP header. In place of the numeric field value, you can specify one of the following keywords (the field values are also listed): **dont-fragment** (0x4), **more-fragments** (0x2), or **reserved** (0x8).	
fragment-offset *value*	Match the 13-bit fragment offset field in the IP header. The value is the offset, in 8-byte units, in the overall datagram message to the data fragment. Specify a numeric value, a range of values, or a set of values. An offset value of **0** indicates the first fragment of a fragmented packet. The **first-fragment** match condition is an alias for the **fragment-offset 0** match condition. To match both first and trailing fragments, you can use two terms that specify different match conditions (**first-fragment** and **is-fragment**). NOTE: This match condition is not supported on PTX1000 routers.	
fragment-offset-except *number*	Do not match the 13-bit fragment offset field. NOTE: This match condition is not supported on PTX1000 routers.	

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

<table>
<tr><th>Match Condition</th><th>Description</th></tr>
<tr><td>gre-key range</td><td>Match the gre-key field. The GRE key field is a 4 octet number inserted by the GRE encapsulator. It is an optional field for use in GRE encapsulation. The range can be a single GRE key number or a range of key numbers.

For MX Series routers with MPCs, initialize new firewall filters that include this condition by walking the corresponding SNMP MIB.</td></tr>
<tr><td>icmp-code number</td><td>Match the ICMP message code field.

NOTE: When using this match condition, you should also use the protocol icmp match condition in the same term (as shown below) to ensure that icmp packets are being evaluated.

<pre>term Allow _ICMP {
                from protocol icmp {
                    icmp-code ip-header-bad;
                    icmp-type echo-reply;
                }
                then {
                    policer ICMP_Policier;
                    count Allow_ICMP;</pre>
You must also configure the icmp-type message-type match condition in the same term. An ICMP message code provides more specific information than an ICMP message type, but the meaning of an ICMP message code is dependent on the associated ICMP message type.

In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed). The keywords are grouped by the ICMP type with which they are associated:

• parameter-problem: ip-header-bad (0), required-option-missing (1)
• redirect: redirect-for-host (1), redirect-for-network (0), redirect-for-tos-and-host (3), redirect-for-tos-and-net (2)
• time-exceeded: ttl-eq-zero-during-reassembly (1), ttl-eq-zero-during-transit (0)
• unreachable: communication-prohibited-by-filtering (13), destination-host-prohibited (10), destination-host-unknown (7), destination-network-prohibited (9), destination-network-unknown (6), fragmentation-needed (4), host-precedence-violation (14), host-unreachable (1), host-unreachable-for-TOS (12), network-unreachable (0), network-unreachable-for-TOS (11), port-unreachable (3), precedence-cutoff-in-effect (15), protocol-unreachable (2), source-host-isolated (8), source-route-failed (5)</td></tr>
<tr><td>icmp-code-except message-code</td><td>Do not match the ICMP message code field. For details, see the icmp-code match condition.</td></tr>
</table>

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

<table>
<tr><th>Match Condition</th><th>Description</th></tr>
<tr><td>icmp-type number</td><td>Match the ICMP message type field.

NOTE: When using this match condition, you should also use the protocol icmp match condition in the same term (as shown below) to ensure that icmp packets are being evaluated.

<pre>term Allow _ICMP {
                from protocol icmp {
                    icmp-type echo-reply;
                }
                then {
                    policer ICMP_Policier;
                    count Allow_ICMP;</pre>
In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): echo-reply (0), echo-request (8), info-reply (16), info-request (15), mask-request (17), mask-reply (18), parameter-problem (12), redirect (5), router-advertisement (9), router-solicit (10), source-quench (4), time-exceeded (11), timestamp (13), timestamp-reply (14), or unreachable (3).</td></tr>
<tr><td>icmp-type-except message-type</td><td>Do not match the ICMP message type field. For details, see the icmp-type match condition.</td></tr>
<tr><td>interface interface-name</td><td>Match the interface on which the packet was received.

NOTE: If you configure this match condition with an interface that does not exist, the term does not match any packet.</td></tr>
<tr><td>interface-group group-number</td><td>Match the logical interface on which the packet was received to the specified interface group or set of interface groups. For group-number, specify a single value or a range of values from 0 through 255.

To assign a logical interface to an interface group group-number, specify the group-number at the [interfaces interface-name unit number family family filter group] hierarchy level.

NOTE: This match condition is not supported on PTX series routers.

For more information, see “Filtering Packets Received on a Set of Interface Groups Overview” on page 861.</td></tr>
<tr><td>interface-group-except group-number</td><td>Do not match the logical interface on which the packet was received to the specified interface group or set of interface groups. For details, see the interface-group match condition.

NOTE: This match condition is not supported on PTX series routers.</td></tr>
<tr><td>interface-set interface-set-name</td><td>Match the interface on which the packet was received to the specified interface set.

To define an interface set, include the interface-set statement at the [edit firewall] hierarchy level.

NOTE: This match condition is not supported on PTX series routers.

For more information, see “Filtering Packets Received on an Interface Set Overview” on page 862.</td></tr>
</table>

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
ip-options ***values***	Match the 8-bit IP option field, if present, to the specified value or list of values. In place of a numeric value, you can specify one of the following text synonyms (the option values are also listed): **loose-source-route** (131), **record-route** (7), **router-alert** (148), **security** (130), **stream-id** (136),**strict-source-route** (137), or **timestamp** (68). To match *any* value for the IP option, use the text synonym **any**. To match on *multiple* values, specify the list of values within square brackets ('**[**' and '**]**'). To match a *range* of values, use the value specification **[*value1-value2*]**. For example, the match condition **ip-options [0-147]** matches on an IP options field that contains the **loose-source-route**, **record-route**, or **security** values, or any other value from 0 through 147. However, this match condition does not match on an IP options field that contains only the **router-alert** value (148). For most interfaces, a filter term that specifies an **ip-option** match on one or more *specific* IP option values (a value other than **any**) causes packets to be sent to the Routing Engine so that the kernel can parse the IP option field in the packet header. • For a firewall filter term that specifies an **ip-option** match on one or more specific IP option values, you cannot specify the **count**, **log**, or **syslog** nonterminating actions *unless* you also specify the **discard** terminating action in the same term. This behavior prevents double-counting of packets for a filter applied to a transit interface on the router. • Packets processed on the kernel might be dropped in case of a system bottleneck. To ensure that matched packets are instead sent to the Packet Forwarding Engine (where packet processing is implemented in hardware), use the **ip-options any** match condition. The 10-Gigabit Ethernet Modular Port Concentrator (MPC), 100-Gigabit Ethernet MPC, 60-Gigabit Ethernet MPC, 60-Gigabit Queuing Ethernet MPC, and 60-Gigabit Ethernet Enhanced Queuing MPC on MX Series routers are capable of parsing the IP option field of the IPv4 packet header. For interfaces configured on those MPCs, *all* packets that are matched using the **ip-options** match condition are sent to the Packet Forwarding Engine for processing. NOTE: On M and T series routers, firewall filters cannot count **ip-options** packets on a per option type and per interface basis. A limited work around is to use the **show pfe statistics ip options** command to see **ip-options** statistics on a per PFE basis. See *show pfe statistics ip* for sample output.
ip-options-except ***values***	Do not match the IP option field to the specified value or list of values. For details about specifying the ***values***, see the **ip-options** match condition.
is-fragment	Match if the packet is a trailing fragment of a fragmented packet. Do not match the first fragment of a fragmented packet. NOTE: To match both first and trailing fragments, you can use two terms that specify different match conditions (**first-fragment** and **is-fragment**).

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
loss-priority *level*	Match the packet loss priority (PLP) level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. Supported on M120 and M320 routers; M7i and M10i routers with the Enhanced CFEB (CFEB-E); and MX Series routers. For IP traffic on M320, MX Series, and T Series routers with Enhanced II Flexible PIC Concentrators (FPCs), you must include the **tri-color** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tri-color** statement is not enabled, you can only configure the **high** and **low** levels. This applies to all protocol families. NOTE: This match condition is not supported on PTX series routers. For information about the **tri-color** statement, see *Configuring and Applying Tricolor Marking Policers*. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*.
loss-priority-except *level*	Do not match the PLP level. For details, see the **loss-priority** match condition. NOTE: This match condition is not supported on PTX series routers.
packet-length *bytes*	Match the length of the received packet, in bytes. The length refers only to the IP packet, including the packet header, and does not include any Layer 2 encapsulation overhead.
packet-length-except *bytes*	Do not match the length of the received packet, in bytes. For details, see the **packet-length** match type. NOTE: This match condition is not supported on PTX1000 routers.
port *number*	Match the UDP or TCP source or destination port field. If you configure this match condition, you cannot configure the **destination-port** match condition or the **source-port** match condition in the same term. If you configure this match condition, we recommend that you also configure the **protocol udp** or **protocol tcp** match statement in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the text synonyms listed under **destination-port**.
port-except *number*	Do not match either the source or destination UDP or TCP port field. For details, see the **port** match condition.
precedence *ip-precedence-value*	Match the IP precedence field. In place of the numeric field value, you can specify one of the following text synonyms (the field values are also listed): **critical-ecp** (0xa0), **flash** (0x60), **flash-override** (0x80), **immediate** (0x40), **internet-control** (0xc0), **net-control** (0xe0), **priority** (0x20), or **routine** (0x00). You can specify precedence in hexadecimal, binary, or decimal form.

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
precedence-except *ip-precedence-value*	Do not match the IP precedence field. In place of the numeric field value, you can specify one of the following text synonyms (the field values are also listed): **critical-ecp** (0xa0), **flash** (0x60), **flash-override** (0x80), **immediate** (0x40), **internet-control** (0xc0), **net-control** (0xe0), **priority** (0x20), or **routine** (0x00). You can specify precedence in hexadecimal, binary, or decimal form.
prefix-list *name* **[except]**	Match the prefixes of the source or destination address fields to the prefixes in the specified list unless the **except** option is included. If the option is included, do not match the prefixes of the source or destination address fields to the prefixes in the specified list. The prefix list is defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level. NOTE: This match condition is not supported on PTX1000 routers.
protocol *number*	Match the IP protocol type field. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **ah** (51), **dstopts** (60), **egp** (8), **esp** (50), **fragment** (44), **gre** (47), **hop-by-hop** (0), **icmp** (1), **icmp6** (58), **icmpv6** (58), **igmp** (2), **ipip** (4), **ipv6** (41), **ospf** (89), **pim** (103), **rsvp** (46), **sctp** (132), **tcp** (6), **udp** (17), or **vrrp** (112).
protocol-except *number*	Do not match the IP protocol type field. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **ah** (51), **dstopts** (60), **egp** (8), **esp** (50), **fragment** (44), **gre** (47), **hop-by-hop** (0), **icmp** (1), **icmp6** (58), **icmpv6** (58), **igmp** (2), **ipip** (4), **ipv6** (41), **ospf** (89), **pim** (103), **rsvp** (46), **sctp** (132), **tcp** (6), **udp** (17), or **vrrp** (112).

rat-type *tech-type-value*	Match the radio-access technology (RAT) type specified in the 8-bit Tech-Type field of Proxy Mobile IPv4 (PMIPv4) access technology type extension. The technology type specifies the access technology through which the mobile device is connected to the access network.
	
	Specify a single value, a range of values, or a set of values. You can specify a technology type as a numeric value from 0 through 255 or as a system keyword.
	
	• The following numeric values are examples of well-known technology types:
	  • Numeric value 1 matches IEEE 802.3.
	  • Numeric value 2 matches IEEE 802.11a/b/g.
	  • Numeric value 3 matches IEEE 802.16e
	  • Numeric value 4 matches IEEE 802.16m.
	• Text string **eutran** matches 4G.
	• Text string **geran** matches 2G.
	• Text string **utran** matches 3G.

rat-type-except *tech-type-value*	Do not match the RAT Type.
service-filter-hit	Match a packet received from a filter where a **service-filter-hit** action was applied. NOTE: This match condition is not supported on PTX series routers.

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
source-address ***address*** **[except]**	Match the IPv4 address of the source node sending the packet unless the **except** option is included. If the option is included, do not match the IPv4 address of the source node sending the packet. You cannot specify both the **address** and **source-address** match conditions in the same term. NOTE: The **except** option is not supported on PTX1000 routers.
source-class ***class-names***	Match one or more specified source class names (sets of source prefixes grouped together and given a class name). For more information, see "Firewall Filter Match Conditions Based on Address Classes" on page 614. NOTE: This match condition is not supported on PTX series routers.
source-class-except ***class-names***	Do not match one or more specified source class names. For details, see the **source-class** match condition. NOTE: This match condition is not supported on PTX series routers.
source-port ***number***	Match the UDP or TCP source port field. You cannot specify the **port** and **source-port** match conditions in the same term. If you configure this match condition for IPv4 traffic, we recommend that you also configure the **protocol udp** or **protocol tcp** match statement in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the text synonyms listed with the **destination-port** ***number*** match condition.
source-port-except ***number***	Do not match the UDP or TCP source port field. For details, see the **source-port** match condition.
source-prefix-list ***name*** **[except]**	Match source prefixes in the specified list unless the **except** option is included. If the option is included, do not match the source prefixes in the specified list. Specify the name of a prefix list defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level.
tcp-established	Match TCP packets of an established TCP session (packets other than the first packet of a connection). This is an alias for **tcp-flags "(ack | rst)"**. This match condition does not implicitly check that the protocol is TCP. To check this, specify the **protocol tcp** match condition.

Table 34: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
tcp-flags ***value***	Match one or more of the low-order 6 bits in the 8-bit TCP flags field in the TCP header. To specify individual bit fields, you can specify the following text synonyms or hexadecimal values: • **fin** (0x01) • **syn** (0x02) • **rst** (0x04) • **push** (0x08) • **ack** (0x10) • **urgent** (0x20) In a TCP session, the SYN flag is set only in the initial packet sent, while the ACK flag is set in all packets sent after the initial packet. You can string together multiple flags using the bit-field logical operators. For combined bit-field match conditions, see the **tcp-established** and **tcp-initial** match conditions. If you configure this match condition, we recommend that you also configure the **protocol tcp** match statement in the same term to specify that the TCP protocol is being used on the port. For IPv4 traffic only, this match condition does not implicitly check whether the datagram contains the first fragment of a fragmented packet. To check for this condition for IPv4 traffic only, use the **first-fragment** match condition.
tcp-initial	Match the initial packet of a TCP connection. This is an alias for **tcp-flags "(!ack & syn)"**. This condition does not implicitly check that the protocol is TCP. If you configure this match condition, we recommend that you also configure the **protocol tcp** match condition in the same term.
ttl ***number***	Match the IPv4 time-to-live number. Specify a TTL value or a range of TTL values. For ***number***, you can specify one or more values from **0** through **255**. This match condition is supported only on M120, M320, MX Series, and T Series routers.
ttl-except ***number***	Do not match on the IPv4 TTL number. For details, see the **ttl** match condition.

Release History Table

Release	Description
13.3R7	Support was added for filtering on Differentiated Services Code Point (DSCP) and forwarding class for Routing Engine sourced packets, including IS-IS packets encapsulated in generic routing encapsulation (GRE).

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Nonterminating Actions on page 673
- Firewall Filter Match Conditions for IPv6 Traffic on page 629

- **enhanced-mode on page 1245**
- Firewall Filter Flexible Match Conditions on page 667

Firewall Filter Match Conditions Based on Numbers or Text Aliases

This topic covers the following information:

Matching on a Single Numeric Value

You can specify a firewall filter match condition based on whether a particular packet field value is a specified numeric value. In the following example, a match occurs if the packet source port number is **25**:

```
[edit firewall family inet filter filter1 term term1 from]
user@host# set source-port 25
```

Matching on a Range of Numeric Values

You can specify a firewall filter match condition based on whether a particular packet field value falls within a specified range of numeric values. In the following example, a match occurs for source ports values from **1024** through **65,535**, inclusive:

```
[edit firewall family inet filter filter2 term term1 from]
user@host# set source-port 1024-65536
```

Matching on a Text Alias for a Numeric Value

You can specify a firewall filter match condition based on whether a particular packet field value is a numeric value that you specify by using a text string as an *alias* for the numeric value. In the following example, a match occurs if the packet source port number is **25**. For the **source-port** and **destination-port** match conditions, the text alias**smtp** corresponds to the numeric value **25**.

```
[edit firewall family inet filter filter3 term term1 from]
user@host# set source-port smtp
```

Matching on a List of Numeric Values or Text Aliases

You can specify a firewall filter match condition based on whether a particular packet field value matches any one of multiple numeric values or text aliases that you specify within square brackets and delimited by spaces. In the following example, a match occurs if the packet source port number is any of the following values: **20** (which corresponds to the text aliases **ftp-data**), **25**, or any value from **1024** through **65535**.

```
[edit firewall family inet filter filter3 term term1 from]
user@host# set source-port [ smtp ftp-data 25 1024-65535 ]
```


Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Match Conditions Based on Bit-Field Values on page 602
- Firewall Filter Match Conditions Based on Address Fields on page 606
- Firewall Filter Match Conditions Based on Address Classes on page 614

Firewall Filter Match Conditions Based on Bit-Field Values

Match Conditions for Bit-Field Values

Table 35 on page 602 lists the firewall filter match conditions that are based on whether certain bit fields in a packet are set or not set. The second and third columns list the types of traffic for which the match condition is supported.

Table 35: Binary and Bit-Field Match Conditions for Firewall Filters

Bit-Field Match Condition	Match Values	Protocol Families for Standard Stateless Firewall Filters	Protocol Families for Service Filters
fragment-flags ***flags***	Hexadecimal values or text aliases for the three-bit IP fragmentation flags field in the IP header.	**family inet**	**family inet**
fragment-offset ***value***	Hexadecimal values or text aliases for the 13-bit fragment offset field in the IP header.	**family inet**	**family inet**
tcp-flags ***value***†	Hexadecimal values or text aliases for the low-order 6 bits of the 8-bit TCP flags field in the TCP header.	**family inet** **family inet6** **family vpls** **family bridge**	**family inet** **family inet6**

† The Junos OS does not automatically check the first fragment bit when matching TCP flags for IPv4 traffic. To check the first fragment bit for IPv4 traffic only, use the **first-fragment** match condition.

Match Conditions for Common Bit-Field Values or Combinations

Table 36 on page 603 describes firewall filter match conditions that are based on whether certain commonly used values or *combinations* of bit fields in a packet are set or not set.

You can use text synonyms to specify some common bit-field matches. In the previous example, you can specify **tcp-initial** as the same match condition.

NOTE:
Some of the numeric range and bit-field match conditions allow you to specify a text synonym. For a complete list of synonyms:

- If you are using the J-Web interface, select the synonym from the appropriate list.
- If you are using the CLI, type a question mark (**?**) after the **from** statement.

Table 36: Bit-Field Match Conditions for Common Combinations

Match Condition	Description	Protocol Families for Standard Stateless Firewall Filters	Protocol Families for Service Filters
first-fragment	Text alias for the bit-field match condition **fragment-offset 0**, which indicates the first fragment of a fragmented packet.	**family inet**	**family inet**
is-fragment	Text alias for the bit-field match condition **fragment-offset 0 except**, which indicates a trailing fragment of a fragmented packet.	**family inet**	**family inet**
tcp-established	Alias for the bit-field match condition **tcp-flags "(ack | rst)"**, which indicates an established TCP session, but not the first packet of a TCP connection.	**family inet** **family inet6**	—
tcp-initial	Alias for the bit-field match condition **tcp-flags "(!ack & syn)"**, which indicates the first packet of a TCP connection, but not an established TCP session.	**family inet** **family inet6**	—

Logical Operators for Bit-Field Values

Table 37 on page 604 lists the logical operators you can apply to *single* bit-field values when specifying stateless firewall filter match conditions. The operators are listed in order, from highest precedence to lowest precedence. Operations are left-associative, meaning that the operations are processed from left to right.

Table 37: Bit-Field Logical Operators

Precedence Order	Bit-Field Logical Operator	Description
1	(*complex-match-condition*)	Grouping—The complex match condition is evaluated before any operators outside the parentheses are applied.
2	! *match-condition*	Negation—A match occurs if the match condition is false.
3	*match-condition-1* & *match-condition-2* or *match-condition-1* + *match-condition-2*	Logical AND—A match occurs if both match conditions are true.
4	*match-condition-1* | *match-condition-2* or *match-condition-1* , *match-condition-2*	Logical OR—A match occurs if either match condition is true.

Matching on a Single Bit-Field Value or Text Alias

For the **fragment-flags** and **tcp-flags** bit-match conditions, you can specify firewall filter match conditions based on whether a particular bit in the packet field is set or not set.

- Numeric value to specify a single bit—You can specify a single bit-field match condition by using a numeric value that has one bit set. Depending on the match condition, you can specify a decimal value, a binary value, or a hexadecimal value. To specify a binary value, specify the number with the prefix **b**. To specify a hexadecimal value, specify the number with the prefix **0x**.

  In the following example, a match occurs if the **RST** bit in the TCP flags field is set:

  ```
  [edit firewall family inet filter filter_tcp_rst_number term term1 from]
  user@host# set tcp-flags 0x04
  ```

- Text alias to specify a single bit—You generally specify a single bit-field match condition by using a text alias enclosed in double-quotation marks (“ ”).

  In the following example, a match occurs if the **RST** bit in the TCP flags field is set:

  ```
  [edit firewall family inet filter filter_tcp_rst_alias term term1 from]
  user@host# set tcp-flags “rst”
  ```

Matching on Multiple Bit-Field Values or Text Aliases

You can specify a firewall filter match condition based on whether a particular set of bits in a packet field are set.

- Numeric values to specify multiple set bits—When you specify a numeric value whose binary representation has more than one set bit, the value is treated as a logical AND of the set bits.

  In the following example, the two match conditions are the same. A match occurs if either bit **0x01** or **0x02** is not set:

  ```
  [edit firewall family inet filter reset_or_not_initial_packet term term5 from]
  user@host# set tcp-flags "!0x3"
  user@host# set tcp-flags "!(0x01 & 0x02)"
  ```

- Text aliases that specify common bit-field matches—You can use text aliases to specify some common bit-field matches. You specify these matches as a single keyword.

  In the following example, the **tcp-established** condition, which is an alias for **"(ack | rst)"**, specifies that a match occurs on TCP packets other than the first packet of a connection:

  ```
  [edit firewall family inet filter reset_or_not_initial_packet term term6 from]
  user@host# set tcp-established
  ```

Matching on a Negated Bit-Field Value

To negate a match, precede the value with an exclamation point.

In the following example, a match occurs if the **RST** bit in the TCP flags field is *not* set:

```
[edit firewall family inet filter filter_tcp_rst term term1 from]
user@host# set tcp-flags "!rst"
```

Matching on the Logical OR of Two Bit-Field Values

You can use the *logical OR operator* (**|** or **,**) to specify that a match occurs if a bit field matches either of two bit-field values specified.

In the following example, a match occurs if the packet is *not* the initial packet in a TCP session:

```
[edit firewall family inet filter not_initial_packet term term3 from]
user@host# set tcp-flags "!syn | ack"
```

In a TCP session, the SYN flag is set only in the initial packet sent, while the ACK flag is set in all packets sent after the initial packet. In a packet that is not the initial packet in a TCP session, either the SYN flag is not set or the ACK flag is set.

Matching on the Logical AND of Two Bit-Field Values

You can use the *logical AND operator* (**&** or **+**) to specify that a match occurs if a bit field matches both of two bit-field values specified.

In the following example, a match occurs if the packet is the initial packet in a TCP session:

```
[edit firewall family inet filter initial_packet term term2 from]
user@host# set tcp-flags "syn & !ack"
```

In a TCP session, the SYN flag is set only in the initial packet sent, while the ACK flag is set in all packets sent after the initial packet. In a packet that is an initial packet in a TCP session, the SYN flag is set and the ACK flag is not set.

Grouping Bit-Field Match Conditions

You can use the *logical grouping notation* to specify that the complex match condition inside the parentheses is evaluated before any operators outside the parentheses are applied.

In the following example, a match occurs if the packet is a TCP reset or if the packet is not the initial packet in the TCP session:

```
[edit firewall family inet filter reset_or_not_initial_packet term term4 from]
user@host# set tcp-flags "!(syn & !ack) | rst"
```

In a TCP session, the SYN flag is set only in the initial packet sent, while the ACK flag is set in all packets sent after the initial packet. In a packet that is *not* the initial packet in a TCP session, the SYN flag is not set and the ACK field is set.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Match Conditions Based on Numbers or Text Aliases on page 601
- Firewall Filter Match Conditions Based on Address Fields on page 606
- Firewall Filter Match Conditions Based on Address Classes on page 614

Firewall Filter Match Conditions Based on Address Fields

You can configure firewall filter match conditions that evaluate packet address fields—IPv4 source and destination addresses, IPv6 source and destination addresses, or media access control (MAC) source and destination addresses—against specified addresses or prefix values.

Implied Match on the ’0/0 except’ Address for Firewall Filter Match Conditions Based on Address Fields

Every firewall filter match condition based on a set of addresses or address prefixes is associated with an implicit match on the address **0.0.0.0/0 except** (for IPv4 or VPLS traffic) or **0:0:0:0:0:0:0:0/0 except** (for IPv6 traffic). As a result, any packet whose specified address field does not match any of the specified addresses or address prefixes fails to match the entire term.

Matching an Address Field to a Subnet Mask or Prefix

You can specify a single match condition to match a source address or destination address that falls within a specified address prefix.

IPv4 Subnet Mask Notation

For an IPv4 address, you can specify a subnet mask value rather than a prefix length. For example:

```
[edit firewall family inet filter filter_on_dst_addr term term3 from]
user@host# set address 10.0.0.10/255.0.0.255
```

Prefix Notation

To specify the address prefix, use the notation ***prefix/prefix-length***. In the following example, a match occurs if a destination address matches the prefix **10.0.0.0/8**:

```
[edit firewall family inet filter filter_on_dst_addr term term1 from]
user@host# set destination-address 10.0.0.0/8
```

Default Prefix Length for IPv4 Addresses

If you do not specify ***/prefix-length*** for an IPv4 address, the prefix length defaults to **/32**. The following example illustrates the default prefix value:

```
[edit firewall family inet filter filter_on_dst_addr term term2 from]
user@host# set destination-address 10
user@host# show
destination-address {
  10.0.0.0/32;
}
```

Default Prefix Length for IPv6 Addresses

If you do not specify ***/prefix-length*** for an IPv6 address, the prefix length defaults to **/128**. The following example illustrates the default prefix value:

```
[edit firewall family inet6 filter filter_on_dst_addr term term1 from]
user@host# set destination-address ::10
user@host# show
destination-address {
  ::10/128;
}
```

Default Prefix Length for MAC Addresses

If you do not specify ***/prefix-length*** for a media access control (MAC) address of a VPLS, Layer 2 CCC, or Layer 2 bridging packet, the prefix length defaults to **/48**. The following example illustrates the default prefix value:

```
[edit firewall family vpls filter filter_on_dst_mac_addr term term1 from]
user@host# set destination-mac-address 01:00:0c:cc:cc:cd
user@host# show
destination-address {
   01:00:0c:cc:cc:cd/48;
}
```

Matching an Address Field to an Excluded Value

For the address-field match conditions, you can include the **except** keyword to specify that a match occurs for an address field that does not match the specified address or prefix.

Excluding IP Addresses in IPv4 or IPv6 Traffic

For the following IPv4 and IPv6 match conditions, you can include the **except** keyword to specify that a match occurs for an IP address field that does not match the specified IP address or prefix:

- **address** ***address*** **except**—A match occurs if either the source IP address or the destination IP address does not match the specified address or prefix.
- **source-address** ***address*** **except**—A match occurs if the source IP address does not match the specified address or prefix.
- **destination-address** ***address*** **except**—A match occurs if the destination IP address does not match the specified address or prefix.

In the following example, a match occurs for any IPv4 destination addresses that fall under the **192.168.10.0/8** prefix, except for addresses that fall under **192.168.0.0/16**. All other addresses implicitly do not match this condition.

```
[edit firewall family inet filter filter_on_dst_addr term term1 from]
user@host# set 192.168.0.0/16 except
user@host# set 192.168.10.0/8
user@host# show
destination-address {
   192.168.0.0/16 except;
   192.168.10.0/8;
}
```

In the following example, a match occurs for any IPv4 destination address that does not fall within the prefix **10.1.1.0/24**:

```
[edit firewall family inet filter filter_on_dst_addr term term24 from]
user@host# set destination-address 0.0.0.0/0
user@host# set destination-address 10.1.1.0/24 except
user@host# show
destination-address {
   0.0.0.0/0;
```



```
    10.1.1.0/24 except;
}
```

Excluding IP Addresses in VPLS or Layer 2 Bridging Traffic

For the following VPLS and Layer 2 bridging match conditions on MX Series routers only, you can include the **except** keyword to specify that a match occurs for an IP address field that does not match the specified IP address or prefix:

- **ip-address** ***address*** **except**—A match occurs if either the source IP address or the destination IP address does not match the specified address or prefix.
- **source-ip-address** ***address*** **except**—A match occurs if the source IP address does not match the specified address or prefix.
- **destination-ip-address** ***address*** **except**—A match occurs if the destination IP address does not match the specified address or prefix.

In the following example for filtering VPLS traffic on an MX Series router, a match occurs if the source IP address falls within the exception range of **55.0.1.0/255.0.255.0** and the destination IP address matches **5172.16.5.0/8**:

```
[edit]
firewall {
    family vpls {
        filter fvpls {
            term 1 {
                from {
                    ip-address {
                        55.0.0.0/8;
                        55.0.1.0/255.0.255.0 except;
                    }
                }
                then {
                    count from-55/8;
                    discard;
                }
            }
        }
    }
}
```

Excluding MAC Addresses in VPLS or Layer 2 Bridging Traffic

For the following VPLS or Layer 2 bridging traffic match conditions, you can include the **except** keyword to specify that a match occurs for a MAC address field that does not match the specified MAC address or prefix:

- **source-mac-address** ***address*** **except**—A match occurs if the source MAC address does not match the specified address or prefix.
- **destination-mac-address** ***address*** **except**—A match occurs if either the destination MAC address does not match the specified address or prefix.

Excluding All Addresses Requires an Explicit Match on the '0/0' Address

If you specify a firewall filter match condition that consists of one or more address-*exception* match conditions (address match conditions that use the **except** keyword) but no *matchable* address match conditions, packets that do not match any of the configured prefixes fails the overall match operation. To configure a firewall filter term of address-exception match conditions to match any address that is not in the prefix list, include an explicit match of **0/0** so that the term contain a matchable address.

For the following example firewall filter for IPv4 traffic, the **from-trusted-addresses** term fails to discard matching traffic, and the **INTRUDERS-COUNT** counter is missing from the output of the **show firewall** operational mode command:

```
[edit]
user@host# show policy-options
prefix-list TRUSTED-ADDRESSES {
  10.2.1.0/24;
  192.168.122.0/24;
}

[edit firewall family inet filter protect-RE]
user@host# show
term from-trusted-addresses {
  from {
    source-prefix-list {
      TRUSTED-ADDRESSES except;
    }
    protocol icmp;
  }
  then {
    count INTRUDERS-COUNT;
    discard;
  }
}
  term other-icmp {
    from {
      protocol icmp;
    }
    then {
      count VALID-COUNT;
      accept;
    }
  }
  term all {
    then accept;
  }
```

```
[edit]
user@host# run show firewall
Filter: protect-RE
Counters:
Name                                              Bytes              Packets
VALID-COUNT                                        2770                   70
Filter: __default_bpdu_filter__
```


To cause a filter term of address-exception match conditions to match any address that is not in the prefix list, include an explicit match of **0/0** in the set of match conditions:

```
[edit firewall family inet filter protect-RE]
user@host# show term from-trusted-addresses
from {
  source-prefix-list {
    0.0.0.0/0;
    TRUSTED-ADDRESSES except;
  }
  protocol icmp;
}
```

With the addition of the **0.0.0.0/0** source prefix address to the match condition, the **from-trusted-addresses** term discards matching traffic, and the INTRUDERS-COUNT counter displays in the output of the **show firewall** operational mode command:

```
[edit]
user@host# run show firewall
Filter: protect-RE
Counters:
Name                                                Bytes              Packets
VALID-COUNT                                          2770                   70
INTRUDERS-COUNT                                       420                    5
Filter: __default_bpdu_filter__
```

Matching Either IP Address Field to a Single Value

For IPv4 and IPv6 traffic and for VPLS and Layer 2 bridging traffic on MX Series routers only, you can use a single match condition to match a single address or prefix value to either the source or destination IP address field.

Matching Either IP Address Field in IPv4 or IPv6 Traffic

For IPv4 or IPv6 traffic, you can use a single match condition to specify the same address or prefix value as the match for either the source or destination IP address field. Instead of creating separate filter terms that specify the same address for the **source-address** and **destination-address** match conditions, you use only the **address** match condition. A match occurs if *either* the source IP address *or* the destination IP address matches the specified address or prefix.

If you use the **except** keyword with the **address** match condition, a match occurs if *both* the source IP address and the destination IP address match the specified value *before* the exception applies.

In a firewall filter term that specifies either the **source-address** or the **destination-address** match condition, you cannot also specify the **address** match condition.

Matching Either IP Address Field in VPLS or Layer 2 Bridging Traffic

For VPLS or Layer 2 bridging traffic on MX Series routers only, you can use a single match condition to specify the same address or prefix value as the match for either the source or destination IP address field. Instead of creating separate filter terms that specify the same address for the **source-ip-address** and **destination-ip-address** match conditions, you use only the **ip-address** match condition. A match occurs if *either* the source IP address *or* the destination IP address matches the specified address or prefix.

If you use the **except** keyword with the **ip-address** match condition, a match occurs if *both* the source IP address and the destination IP address match the specified value *before* the exception applies.

In a firewall filter term that specifies either the **source-ip-address** or the **destination-ip-address** match condition, you cannot also specify the **ip-address** match condition.

Matching an Address Field to Noncontiguous Prefixes

For IPv4 traffic only, specify a single match condition to match the IP source or destination address field to any prefix specified. The prefixes do not need to be contiguous. That is, the prefixes under the **source-address** or **destination-address** match condition do not need to be adjacent or neighboring to one another.

In the following example, a match occurs if a destination address matches either the **10.0.0.0/8** prefix or the **192.168.0.0/32** prefix:

```
[edit firewall family inet filter filter_on_dst_addr term term5 from]
user@host# set destination-address 10.0.0.0/8
user@host# set destination-address 192.168.0.0/32
user@host# show
destination-address {
   destination-address 10.0.0.0/8;
   destination-address 192.168.0.0/32;
}
```

The order in which you specify the prefixes within the match condition is not significant. Packets are evaluated against all the prefixes in the match condition to determine whether a match occurs. If prefixes overlap, longest-match rules are used to determine whether a match occurs. A match condition of noncontiguous prefixes includes an implicit **0/0 except** statement, which means that any prefix that does not match any prefix included in the match condition is explicitly considered not to match.

Because the prefixes are order-independent and use longest-match rules, longer prefixes subsume shorter ones as long as they are the same type (whether you specify **except** or not). This is because anything that would match the longer prefix would also match the shorter one.

Consider the following example:

```
[edit firewall family inet filter filter_on_src_addr term term1 from]
source-address {
   172.16.0.0/10;
   172.16.2.0/24 except;
   192.168.1.0;
   192.168.1.192/26 except;
   192.168.1.254;
   172.16.3.0/24; # ignored
   10.2.2.2 except; # ignored
}
```


Within the **source-address** match condition, two addresses are ignored. The **172.16.3.0/16** value is ignored because it falls under the address **172.16.0.0/10**, which is the same type. The **10.2.2.2 except** value is ignored because it is subsumed by the implicit **0.0.0.0/0 except** match value.

Suppose the following source IP address are evaluated by this firewall filter:

- Source IP address **172.16.1.2**—This address matches the **172.16.0.0/10** prefix, and thus the action in the **then** statement is taken.
- Source IP address **172.16.2.2**—This address matches the **172.16.2.0/24** prefix. Because this prefix is negated (that is, includes the **except** keyword), an explicit *mismatch* occurs. The next term in the filter is evaluated, if there is one. If there are no more terms, the packet is discarded.
- Source IP address **10.1.2.3**—This address does not match any of the prefixes included in the **source-address** condition. Instead, it matches the implicit **0.0.0.0/0 except** at the end of the list of prefixes configured under the **source-address** match condition, and is considered to be a mismatch.

  The **172.16.3.0/24** statement is ignored because it falls under the address **172.16.0.0/10**—both are the same type.

  The **10.2.2.2 except** statement is ignored because it is subsumed by the implicit **0.0.0.0/0 except** statement at the end of the list of prefixes configured under the **source-address** match condition.

BEST PRACTICE: When a firewall filter term includes the **from address** ***address*** match condition and a subsequent term includes the **from source-address** ***address*** match condition for the same address, packets might be processed by the latter term before they are evaluated by any intervening terms. As a result, packets that should be rejected by the intervening terms might be accepted instead, or packets that should be accepted might be rejected instead.

To prevent this from occurring, we recommend that you do the following. For every firewall filter term that contains the **from address** ***address*** match condition, replace that term with two separate terms: one that contains the **from source-address** ***address*** match condition, and another that contains the **from destination-address** ***address*** match condition.

Matching an Address Field to a Prefix List

You can define a list of IPv4 or IPv6 address prefixes for use in a routing policy statement or in a stateless firewall filter match condition that evaluates packet address fields.

To define a list of IPv4 or IPv6 address prefixes, include the **prefix-list** ***prefix-list*** statement.

```
prefix-list name {
  ip-addresses;
  apply-path path;
```

```
}
```

You can include the statement at the following hierarchy levels:

- **[edit policy-options]**
- **[edit logical-systems *logical-system-name* policy-options]**

After you have defined a prefix list, you can use it when specifying a firewall filter match condition based on an IPv4 or IPv6 address prefix.

```
[edit firewall family family-name filter filter-name term term-name]
from {
  source-prefix-list {
    prefix-lists;
  }
  destination-prefix-list {
    prefix-lists;
  }
}
```

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Match Conditions Based on Numbers or Text Aliases on page 601
- Firewall Filter Match Conditions Based on Bit-Field Values on page 602
- Firewall Filter Match Conditions Based on Address Classes on page 614

Firewall Filter Match Conditions Based on Address Classes

For IPv4 and IPv6 traffic only, you can use class-based firewall filter conditions to match packet fields based on source class or destination class.

- Source-Class Usage on page 614
- Destination-Class Usage on page 614
- Guidelines for Applying SCU or DCU Firewall Filters to Output Interfaces on page 615

Source-Class Usage

A *source class* is a set of source prefixes grouped together and given a class name. To configure a firewall filter term that matches an IP source address field to one or more source classes, use the **source-class *class-name*** match condition under the **[edit firewall family (inet | inet6) filter *filter-name* term *term-name* from]** hierarchy level.

Source-class usage (SCU) enables you to monitor the amount of traffic originating from a specific prefix. With this feature, usage can be tracked and customers can be billed for the traffic they receive.

Destination-Class Usage

A *destination class* is a set of destination prefixes grouped together and given a class name. To configure a firewall filter term that matches an IP destination address field to one or more destination classes, use the **destination-class *class-name*** match condition at the **[edit firewall family (inet | inet6) filter *filter-name* term *term-name* from]** hierarchy level.

Destination-class usage (DCU) enables you can track how much traffic is sent to a specific prefix in the core of the network originating from one of the specified interfaces.

Note, however, that DCU limits your ability to keep track of traffic moving in the reverse direction. It can account for all traffic that arrives on a core interface and heads toward a specific customer, but it cannot count traffic that arrives on a core interface from a specific prefix.

Guidelines for Applying SCU or DCU Firewall Filters to Output Interfaces

When applying a SCU or DCU firewall filter to an interface, keep the following guidelines in mind:

- Output interfaces—Class-based firewall filter match conditions work only for firewall filters that you apply to output interfaces. This is because the SCU and DCU are determined after route lookup occurs.
- Input interfaces—Although you can specify a source class and destination class for an input firewall filter, the counters are incremented only if the firewall filter is applied on the output interface.
- Output interfaces for tunnel traffic—SCU and DCU are not supported on the interfaces you configure as the output interface for tunnel traffic for transit packets exiting the router (or switch) through the tunnel.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Match Conditions for IPv4 Traffic on page 589
- Firewall Filter Match Conditions for IPv6 Traffic on page 629
- *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*
- Firewall Filter Match Conditions Based on Numbers or Text Aliases on page 601
- Firewall Filter Match Conditions Based on Bit-Field Values on page 602
- Firewall Filter Match Conditions Based on Address Fields on page 606

Firewall Filter Match Conditions for Protocol-Independent Traffic

You can configure a firewall filter with match conditions for protocol-independent traffic (**family any**).

To apply a protocol-independent firewall filter to a logical interface, configure the **filter** statement under the logical unit.

NOTE:
On MX Series routers, attach a protocol-independent firewall filter to a logical interface by configuring the **filter** statement *directly* under the logical unit:

- **[edit interfaces *name* unit *number* filter]**
- **[edit logical-systems *name* interfaces *name* unit *number* filter]**

On all other supported devices, attach a protocol-independent firewall filter to a logical interface by configuring the **filter** statement under the protocol family (**family any**):

- **[edit interfaces *name* unit *number* family any filter]**
- **[edit logical-systems *name* interfaces *name* unit *number* family any filter]**

Table 38 on page 616 describes the ***match-conditions*** you can configure at the **[edit firewall family any filter *filter-name* term *term-name* from]** hierarchy level.

Table 38: Firewall Filter Match Conditions for Protocol-Independent Traffic

Match Condition	Description
forwarding-class ***class***	Match the forwarding class of the packet. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**. For information about forwarding classes and router-internal output queues, see *Understanding How Forwarding Classes Assign Classes to Output Queues*. NOTE: On T4000 Type 5 FPCs, a filter attached at the Layer 2 application point (that is, at the logical interface level) is unable to match with the forwarding class of a packet that is set by a Layer 3 classifier such as DSCP, DSCP V6, **inet-precedence**, and **mpls-exp**.
forwarding-class-except ***class***	Do not match on the forwarding class. For details, see the **forwarding-class** match condition.
interface ***interface-name***	Match the interface on which the packet was received. NOTE: If you configure this match condition with an interface that does not exist, the term does not match any packet.
interface-set ***interface-set-name***	Match the interface on which the packet was received to the specified interface set. To define an interface set, include the **interface-set** statement at the **[edit firewall]** hierarchy level. For more information, see “Filtering Packets Received on an Interface Set Overview” on page 862.

Table 38: Firewall Filter Match Conditions for Protocol-Independent Traffic *(continued)*

Match Condition	Description
loss-priority ***level***	Match the packet loss priority (PLP) level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. Supported on M120 and M320 routers; M7i and M10i routers with the Enhanced CFEB (CFEB-E); and MX Series routers. For IP traffic on M320, MX Series, and T Series routers with Enhanced II Flexible PIC Concentrators (FPCs), you must include the **tri-color** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tri-color** statement is not enabled, you can only configure the **high** and **low** levels. This applies to all protocol families. NOTE: This match condition is not supported on PTX series packet transport routers. For information about the **tri-color** statement, see *Configuring and Applying Tricolor Marking Policers*. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.
loss-priority-except ***level***	Do not match the PLP level. For details, see the **loss-priority** match condition. NOTE: This match condition is not supported on PTX series packet transport routers.
packet-length ***bytes***	Match the length of the received packet, in bytes. The length refers only to the IP packet, including the packet header, and does not include any Layer 2 encapsulation overhead.
packet-length-except ***bytes***	Do not match on the received packet length, in bytes. For details, see the **packet-length** match type.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Nonterminating Actions on page 673

Firewall Filter Match Conditions for IPv4 Traffic

You can configure a firewall filter with match conditions for Internet Protocol version 4 (IPv4) traffic (**family inet**).

NOTE: For MX Series routers with MPCs, you need to initialize certain new firewall filters by walking the corresponding SNMP MIB, for example, **show snmp mib walk** ***name*** **ascii**. This forces Junos to learn the filter counters and ensure that the filter statistics are displayed. This guidance applies to all enhanced mode firewall filters, filters with flexible conditions, and filters with the certain terminating actions. See those topics, listed under Related Documentation, for details.

Table 34 on page 590 describes the ***match-conditions*** you can configure at the **[edit firewall family inet filter *filter-name* term *term-name* from]** hierarchy level.

Table 39: Firewall Filter Match Conditions for IPv4 Traffic

Match Condition	Description
address *address* [except]	Match the IPv4 source or destination address field unless the **except** option is included. If the option is included, do not match the IPv4 source or destination address field. NOTE: This match condition is not supported on PTX1000 routers.
ah-spi *spi-value*	(M Series routers, except M120 and M320) Match the IPsec authentication header (AH) security parameter index (SPI) value. NOTE: This match condition is not supported on PTX series routers.
ah-spi-except *spi-value*	(M Series routers, except M120 and M320) Do not match the IPsec AH SPI value. NOTE: This match condition is not supported on PTX series routers.
apply-groups	Specify which groups to inherit configuration data from. You can specify more than one group name. You must list them in order of inheritance priority. The configuration data in the first group takes priority over the data in subsequent groups.
apply-groups-except	Specify which groups not to inherit configuration data from. You can specify more than one group name.
destination-address *address* [except]	Match the IPv4 destination address field unless the **except** option is included. If the option is included, do not match the IPv4 destination address field. You cannot specify both the **address** and **destination-address** match conditions in the same term. NOTE: The **except** option is not supported on PTX1000 routers.
destination-class *class-names*	Match one or more specified destination class names (sets of destination prefixes grouped together and given a class name). For more information, see "Firewall Filter Match Conditions Based on Address Classes" on page 614. NOTE: This match condition is not supported on PTX series routers.
destination-class-except *class-names*	Do not match one or more specified destination class names. For details, see the **destination-class** match condition. NOTE: This match condition is not supported on PTX series routers.

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
destination-port *number*	Match the UDP or TCP destination port field. You cannot specify both the **port** and **destination-port** match conditions in the same term. If you configure this match condition, we recommend that you also configure the **protocol udp** or **protocol tcp** match statement in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the following text synonyms (the port numbers are also listed): **afs** (1483), **bgp** (179), **biff** (512), **bootpc** (68), **bootps** (67), **cmd** (514), **cvspserver** (2401), **dhcp** (67), **domain** (53), **eklogin** (2105), **ekshell** (2106), **exec** (512), **finger** (79), **ftp** (21), **ftp-data** (20), **http** (80), **https** (443), **ident** (113), **imap** (143), **kerberos-sec** (88), **klogin** (543), **kpasswd** (761), **krb-prop** (754), **krbupdate** (760), **kshell** (544), **ldap** (389), **ldp** (646), **login** (513), **mobileip-agent** (434), **mobilip-mn** (435), **msdp** (639), **netbios-dgm** (138), **netbios-ns** (137), **netbios-ssn** (139), **nfsd** (2049), **nntp** (119), **ntalk** (518), **ntp** (123), **pop3** (110), **pptp** (1723), **printer** (515), **radacct** (1813), **radius** (1812), **rip** (520), **rkinit** (2108), **smtp** (25), **snmp** (161), **snmptrap** (162), **snpp** (444), **socks** (1080), **ssh** (22), **sunrpc** (111), **syslog** (514), **tacacs** (49), **tacacs-ds** (65), **talk** (517), **telnet** (23), **tftp** (69), **timed** (525), **who** (513), or **xdmcp** (177).
destination-port-except *number*	Do not match the UDP or TCP destination port field. For details, see the **destination-port** match condition.
destination-prefix-list *name* **[except]**	Match destination prefixes in the specified list unless the **except** option is included. If the option is included, do not match the destination prefixes in the specified list. Specify the name of a prefix list defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level.

dscp *number*	Match the Differentiated Services code point (DSCP). The DiffServ protocol uses the type-of-service (ToS) byte in the IP header. The most significant 6 bits of this byte form the DSCP. For more information, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*.
	
	Support was added for filtering on Differentiated Services Code Point (DSCP) and forwarding class for Routing Engine sourced packets, including IS-IS packets encapsulated in generic routing encapsulation (GRE). Subsequently, when upgrading from a previous version of Junos OS where you have both a class of service (CoS) and firewall filter, and both include DSCP or forwarding class filter actions, the criteria in the firewall filter automatically takes precedence over the CoS settings. The same is true when creating new configurations; that is, where the same settings exist, the firewall filter takes precedence over the CoS, regardless of which was created first.
	
	You can specify a numeric value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix.
	
	In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed):
	
	• RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*, defines one code point: **ef** (46).
	• RFC 2597, *Assured Forwarding PHB Group*, defines 4 classes, with 3 drop precedences in each class, for a total of 12 code points:
	  ◦ **af11** (10), **af12** (12), **af13** (14)
	  ◦ **af21** (18), **af22** (20), **af23** (22)
	  ◦ **af31** (26), **af32** (28), **af33** (30)
	  ◦ **af41** (34), **af42** (36), **af43** (38)

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

<table>
<tr><th>Match Condition</th><th colspan="2">Description</th></tr>
<tr><td>dscp-except number</td><td colspan="2">Do not match on the DSCP number. For more information, see the dscp match condition.</td></tr>
<tr><td>esp-spi spi-value</td><td colspan="2">Match the IPsec encapsulating security payload (ESP) SPI value. Match on this specific SPI value. You can specify the ESP SPI value in hexadecimal, binary, or decimal form.

NOTE: This match condition is not supported on PTX series routers.</td></tr>
<tr><td>esp-spi-except spi-value</td><td colspan="2">Match the IPsec ESP SPI value. Do not match on this specific SPI value.

NOTE: This match condition is not supported on PTX series routers.</td></tr>
<tr><td>first-fragment</td><td colspan="2">Match if the packet is the first fragment of a fragmented packet. Do not match if the packet is a trailing fragment of a fragmented packet. The first fragment of a fragmented packet has a fragment offset value of 0.

This match condition is an alias for the bit-field match condition fragment-offset 0 match condition.

To match both first and trailing fragments, you can use two terms that specify different match conditions: first-fragment and is-fragment.</td></tr>
<tr><td rowspan="7">flexible-match-mask value</td><td>bit-length</td><td>Length of the data to be matched in bits, not needed for string input (0..128)</td></tr>
<tr><td>bit-offset</td><td>Bit offset after the (match-start + byte) offset (0..7)</td></tr>
<tr><td>byte-offset</td><td>Byte offset after the match start point</td></tr>
<tr><td>flexible-mask-name</td><td>Select a flexible match from predefined template field</td></tr>
<tr><td>mask-in-hex</td><td>Mask out bits in the packet data to be matched</td></tr>
<tr><td>match-start</td><td>Start point to match in packet</td></tr>
<tr><td>prefix</td><td>Value data/string to be matched</td></tr>
</table>

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description	
flexible-match-range *value*	**bit-length**	Length of the data to be matched in bits (0..32)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-range-name	Select a flexible match from predefined template field
	match-start	Start point to match in packet
	range	Range of values to be matched
	range-except	Do not match this range of values
forwarding-class *class*	Match the forwarding class of the packet. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**. NOTE: This match condition is not supported on PTX1000 routers. For information about forwarding classes and router-internal output queues, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.	
forwarding-class-except *class*	Do not match the forwarding class of the packet. For details, see the **forwarding-class** match condition. NOTE: This match condition is not supported on PTX1000 routers.	
fragment-flags *number*	(Ingress only) Match the three-bit IP fragmentation flags field in the IP header. In place of the numeric field value, you can specify one of the following keywords (the field values are also listed): **dont-fragment** (0x4), **more-fragments** (0x2), or **reserved** (0x8).	
fragment-offset *value*	Match the 13-bit fragment offset field in the IP header. The value is the offset, in 8-byte units, in the overall datagram message to the data fragment. Specify a numeric value, a range of values, or a set of values. An offset value of **0** indicates the first fragment of a fragmented packet. The **first-fragment** match condition is an alias for the **fragment-offset 0** match condition. To match both first and trailing fragments, you can use two terms that specify different match conditions (**first-fragment** and **is-fragment**). NOTE: This match condition is not supported on PTX1000 routers.	
fragment-offset-except *number*	Do not match the 13-bit fragment offset field. NOTE: This match condition is not supported on PTX1000 routers.	

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

<table>
<tr><th>Match Condition</th><th>Description</th></tr>
<tr><td>gre-key range</td><td>Match the gre-key field. The GRE key field is a 4 octet number inserted by the GRE encapsulator. It is an optional field for use in GRE encapsulation. The range can be a single GRE key number or a range of key numbers.

For MX Series routers with MPCs, initialize new firewall filters that include this condition by walking the corresponding SNMP MIB.</td></tr>
<tr><td>icmp-code number</td><td>Match the ICMP message code field.

NOTE: When using this match condition, you should also use the protocol icmp match condition in the same term (as shown below) to ensure that icmp packets are being evaluated.

<pre>term Allow _ICMP {
                from protocol icmp {
                    icmp-code ip-header-bad;
                    icmp-type echo-reply;
                }
                then {
                    policer ICMP_Policier;
                    count Allow_ICMP;</pre>
You must also configure the icmp-type message-type match condition in the same term. An ICMP message code provides more specific information than an ICMP message type, but the meaning of an ICMP message code is dependent on the associated ICMP message type.

In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed). The keywords are grouped by the ICMP type with which they are associated:
parameter-problem: ip-header-bad (0), required-option-missing (1)
redirect: redirect-for-host (1), redirect-for-network (0), redirect-for-tos-and-host (3), redirect-for-tos-and-net (2)
time-exceeded: ttl-eq-zero-during-reassembly (1), ttl-eq-zero-during-transit (0)
unreachable: communication-prohibited-by-filtering (13), destination-host-prohibited (10), destination-host-unknown (7), destination-network-prohibited (9), destination-network-unknown (6), fragmentation-needed (4), host-precedence-violation (14), host-unreachable (1), host-unreachable-for-TOS (12), network-unreachable (0), network-unreachable-for-TOS (11), port-unreachable (3), precedence-cutoff-in-effect (15), protocol-unreachable (2), source-host-isolated (8), source-route-failed (5)</td></tr>
<tr><td>icmp-code-except message-code</td><td>Do not match the ICMP message code field. For details, see the icmp-code match condition.</td></tr>
</table>

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description

icmp-type ***number***	Match the ICMP message type field.
	
	NOTE: When using this match condition, you should also use the **protocol icmp** match condition in the same term (as shown below) to ensure that **icmp** packets are being evaluated.
	
	`term Allow _ICMP {`
	`                from protocol icmp {`
	`                    icmp-type echo-reply;`
	`                }`
	`                then {`
	`                    policer ICMP_Policier;`
	`                    count Allow_ICMP;`
	
	In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **echo-reply** (0), **echo-request** (8), **info-reply** (16), **info-request** (15), **mask-request** (17), **mask-reply** (18), **parameter-problem** (12), **redirect** (5), **router-advertisement** (9), **router-solicit** (10), **source-quench** (4), **time-exceeded** (11), **timestamp** (13), **timestamp-reply** (14), or **unreachable** (3).

icmp-type-except ***message-type***	Do not match the ICMP message type field. For details, see the **icmp-type** match condition.
interface ***interface-name***	Match the interface on which the packet was received. NOTE: If you configure this match condition with an interface that does not exist, the term does not match any packet.
interface-group ***group-number***	Match the logical interface on which the packet was received to the specified interface group or set of interface groups. For ***group-number***, specify a single value or a range of values from 0 through 255. To assign a logical interface to an interface group ***group-number***, specify the ***group-number*** at the **[interfaces *interface-name* unit *number* family *family* filter group]** hierarchy level. NOTE: This match condition is not supported on PTX series routers. For more information, see "Filtering Packets Received on a Set of Interface Groups Overview" on page 861.
interface-group-except ***group-number***	Do not match the logical interface on which the packet was received to the specified interface group or set of interface groups. For details, see the **interface-group** match condition. NOTE: This match condition is not supported on PTX series routers.
interface-set ***interface-set-name***	Match the interface on which the packet was received to the specified interface set. To define an interface set, include the **interface-set** statement at the **[edit firewall]** hierarchy level. NOTE: This match condition is not supported on PTX series routers. For more information, see "Filtering Packets Received on an Interface Set Overview" on page 862.

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
ip-options *values*	Match the 8-bit IP option field, if present, to the specified value or list of values. In place of a numeric value, you can specify one of the following text synonyms (the option values are also listed): **loose-source-route** (131), **record-route** (7), **router-alert** (148), **security** (130), **stream-id** (136),**strict-source-route** (137), or **timestamp** (68). To match *any* value for the IP option, use the text synonym **any**. To match on *multiple* values, specify the list of values within square brackets ('**[**' and '**]**'). To match a *range* of values, use the value specification **[*value1-value2*]**. For example, the match condition **ip-options [0-147]** matches on an IP options field that contains the **loose-source-route**, **record-route**, or **security** values, or any other value from 0 through 147. However, this match condition does not match on an IP options field that contains only the **router-alert** value (148). For most interfaces, a filter term that specifies an **ip-option** match on one or more *specific* IP option values (a value other than **any**) causes packets to be sent to the Routing Engine so that the kernel can parse the IP option field in the packet header. • For a firewall filter term that specifies an **ip-option** match on one or more specific IP option values, you cannot specify the **count**, **log**, or **syslog** nonterminating actions *unless* you also specify the **discard** terminating action in the same term. This behavior prevents double-counting of packets for a filter applied to a transit interface on the router. • Packets processed on the kernel might be dropped in case of a system bottleneck. To ensure that matched packets are instead sent to the Packet Forwarding Engine (where packet processing is implemented in hardware), use the **ip-options any** match condition. The 10-Gigabit Ethernet Modular Port Concentrator (MPC), 100-Gigabit Ethernet MPC, 60-Gigabit Ethernet MPC, 60-Gigabit Queuing Ethernet MPC, and 60-Gigabit Ethernet Enhanced Queuing MPC on MX Series routers are capable of parsing the IP option field of the IPv4 packet header. For interfaces configured on those MPCs, *all* packets that are matched using the **ip-options** match condition are sent to the Packet Forwarding Engine for processing. NOTE: On M and T series routers, firewall filters cannot count **ip-options** packets on a per option type and per interface basis. A limited work around is to use the **show pfe statistics ip options** command to see **ip-options** statistics on a per PFE basis. See *show pfe statistics ip* for sample output.
ip-options-except *values*	Do not match the IP option field to the specified value or list of values. For details about specifying the ***values***, see the **ip-options** match condition.
is-fragment	Match if the packet is a trailing fragment of a fragmented packet. Do not match the first fragment of a fragmented packet. NOTE: To match both first and trailing fragments, you can use two terms that specify different match conditions (**first-fragment** and **is-fragment**).

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
loss-priority *level*	Match the packet loss priority (PLP) level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. Supported on M120 and M320 routers; M7i and M10i routers with the Enhanced CFEB (CFEB-E); and MX Series routers. For IP traffic on M320, MX Series, and T Series routers with Enhanced II Flexible PIC Concentrators (FPCs), you must include the **tri-color** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tri-color** statement is not enabled, you can only configure the **high** and **low** levels. This applies to all protocol families. NOTE: This match condition is not supported on PTX series routers. For information about the **tri-color** statement, see *Configuring and Applying Tricolor Marking Policers*. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*.
loss-priority-except *level*	Do not match the PLP level. For details, see the **loss-priority** match condition. NOTE: This match condition is not supported on PTX series routers.
packet-length *bytes*	Match the length of the received packet, in bytes. The length refers only to the IP packet, including the packet header, and does not include any Layer 2 encapsulation overhead.
packet-length-except *bytes*	Do not match the length of the received packet, in bytes. For details, see the **packet-length** match type. NOTE: This match condition is not supported on PTX1000 routers.
port *number*	Match the UDP or TCP source or destination port field. If you configure this match condition, you cannot configure the **destination-port** match condition or the **source-port** match condition in the same term. If you configure this match condition, we recommend that you also configure the **protocol udp** or **protocol tcp** match statement in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the text synonyms listed under **destination-port**.
port-except *number*	Do not match either the source or destination UDP or TCP port field. For details, see the **port** match condition.
precedence *ip-precedence-value*	Match the IP precedence field. In place of the numeric field value, you can specify one of the following text synonyms (the field values are also listed): **critical-ecp** (0xa0), **flash** (0x60), **flash-override** (0x80), **immediate** (0x40), **internet-control** (0xc0), **net-control** (0xe0), **priority** (0x20), or **routine** (0x00). You can specify precedence in hexadecimal, binary, or decimal form.

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
precedence-except *ip-precedence-value*	Do not match the IP precedence field. In place of the numeric field value, you can specify one of the following text synonyms (the field values are also listed): **critical-ecp** (0xa0), **flash** (0x60), **flash-override** (0x80), **immediate** (0x40), **internet-control** (0xc0), **net-control** (0xe0), **priority** (0x20), or **routine** (0x00). You can specify precedence in hexadecimal, binary, or decimal form.
prefix-list *name* **[except]**	Match the prefixes of the source or destination address fields to the prefixes in the specified list unless the **except** option is included. If the option is included, do not match the prefixes of the source or destination address fields to the prefixes in the specified list. The prefix list is defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level. NOTE: This match condition is not supported on PTX1000 routers.
protocol *number*	Match the IP protocol type field. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **ah** (51), **dstopts** (60), **egp** (8), **esp** (50), **fragment** (44), **gre** (47), **hop-by-hop** (0), **icmp** (1), **icmp6** (58), **icmpv6** (58), **igmp** (2), **ipip** (4), **ipv6** (41), **ospf** (89), **pim** (103), **rsvp** (46), **sctp** (132), **tcp** (6), **udp** (17), or **vrrp** (112).
protocol-except *number*	Do not match the IP protocol type field. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **ah** (51), **dstopts** (60), **egp** (8), **esp** (50), **fragment** (44), **gre** (47), **hop-by-hop** (0), **icmp** (1), **icmp6** (58), **icmpv6** (58), **igmp** (2), **ipip** (4), **ipv6** (41), **ospf** (89), **pim** (103), **rsvp** (46), **sctp** (132), **tcp** (6), **udp** (17), or **vrrp** (112).

rat-type *tech-type-value*	Match the radio-access technology (RAT) type specified in the 8-bit Tech-Type field of Proxy Mobile IPv4 (PMIPv4) access technology type extension. The technology type specifies the access technology through which the mobile device is connected to the access network.
	
	Specify a single value, a range of values, or a set of values. You can specify a technology type as a numeric value from 0 through 255 or as a system keyword.
	
	• The following numeric values are examples of well-known technology types:
	  ◦ Numeric value 1 matches IEEE 802.3.
	  ◦ Numeric value 2 matches IEEE 802.11a/b/g.
	  ◦ Numeric value 3 matches IEEE 802.16e
	  ◦ Numeric value 4 matches IEEE 802.16m.
	• Text string **eutran** matches 4G.
	• Text string **geran** matches 2G.
	• Text string **utran** matches 3G.

rat-type-except *tech-type-value*	Do not match the RAT Type.
service-filter-hit	Match a packet received from a filter where a **service-filter-hit** action was applied. NOTE: This match condition is not supported on PTX series routers.

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
source-address ***address*** **[except]**	Match the IPv4 address of the source node sending the packet unless the **except** option is included. If the option is included, do not match the IPv4 address of the source node sending the packet. You cannot specify both the **address** and **source-address** match conditions in the same term. NOTE: The **except** option is not supported on PTX1000 routers.
source-class ***class-names***	Match one or more specified source class names (sets of source prefixes grouped together and given a class name). For more information, see "Firewall Filter Match Conditions Based on Address Classes" on page 614. NOTE: This match condition is not supported on PTX series routers.
source-class-except ***class-names***	Do not match one or more specified source class names. For details, see the **source-class** match condition. NOTE: This match condition is not supported on PTX series routers.
source-port ***number***	Match the UDP or TCP source port field. You cannot specify the **port** and **source-port** match conditions in the same term. If you configure this match condition for IPv4 traffic, we recommend that you also configure the **protocol udp** or **protocol tcp** match statement in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the text synonyms listed with the **destination-port** ***number*** match condition.
source-port-except ***number***	Do not match the UDP or TCP source port field. For details, see the **source-port** match condition.
source-prefix-list ***name*** **[except]**	Match source prefixes in the specified list unless the **except** option is included. If the option is included, do not match the source prefixes in the specified list. Specify the name of a prefix list defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level.
tcp-established	Match TCP packets of an established TCP session (packets other than the first packet of a connection). This is an alias for **tcp-flags "(ack | rst)"**. This match condition does not implicitly check that the protocol is TCP. To check this, specify the **protocol tcp** match condition.

Table 39: Firewall Filter Match Conditions for IPv4 Traffic *(continued)*

Match Condition	Description
tcp-flags ***value***	Match one or more of the low-order 6 bits in the 8-bit TCP flags field in the TCP header. To specify individual bit fields, you can specify the following text synonyms or hexadecimal values: • **fin** (0x01) • **syn** (0x02) • **rst** (0x04) • **push** (0x08) • **ack** (0x10) • **urgent** (0x20) In a TCP session, the SYN flag is set only in the initial packet sent, while the ACK flag is set in all packets sent after the initial packet. You can string together multiple flags using the bit-field logical operators. For combined bit-field match conditions, see the **tcp-established** and **tcp-initial** match conditions. If you configure this match condition, we recommend that you also configure the **protocol tcp** match statement in the same term to specify that the TCP protocol is being used on the port. For IPv4 traffic only, this match condition does not implicitly check whether the datagram contains the first fragment of a fragmented packet. To check for this condition for IPv4 traffic only, use the **first-fragment** match condition.
tcp-initial	Match the initial packet of a TCP connection. This is an alias for **tcp-flags "(!ack & syn)"**. This condition does not implicitly check that the protocol is TCP. If you configure this match condition, we recommend that you also configure the **protocol tcp** match condition in the same term.
ttl ***number***	Match the IPv4 time-to-live number. Specify a TTL value or a range of TTL values. For ***number***, you can specify one or more values from **0** through **255**. This match condition is supported only on M120, M320, MX Series, and T Series routers.
ttl-except ***number***	Do not match on the IPv4 TTL number. For details, see the **ttl** match condition.

Release History Table

Release	Description
13.3R7	Support was added for filtering on Differentiated Services Code Point (DSCP) and forwarding class for Routing Engine sourced packets, including IS-IS packets encapsulated in generic routing encapsulation (GRE).

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Nonterminating Actions on page 673
- Firewall Filter Match Conditions for IPv6 Traffic on page 629

- **enhanced-mode on page 1245**
- Firewall Filter Flexible Match Conditions on page 667

Firewall Filter Match Conditions for IPv6 Traffic

You can configure a firewall filter with match conditions for Internet Protocol version 6 (IPv6) traffic (**family inet6**).

NOTE: For MX Series routers with MPCs, you need to initialize certain new firewall filters by walking the corresponding SNMP MIB, for example, **show snmp mib walk *name* ascii**. This forces Junos to learn the filter counters and ensure that the filter statistics are displayed. This guidance applies to all enhanced mode firewall filters, filters with flexible conditions, and filters with the certain terminating actions. See those topics, listed under Related Documentation, for details.

Table 40 on page 629 describes the match conditions you can configure at the **[edit firewall family inet6 filter *filter-name* term *term-name* from]** hierarchy level.

Table 40: Firewall Filter Match Conditions for IPv6 Traffic

Match Condition	Description
address *address* [except]	Match the IPv6 source or destination address field unless the **except** option is included. If the option is included, do not match the IPv6 source or destination address field.
apply-groups	Specify which groups to inherit configuration data from. You can specify more than one group name. You must list them in order of inheritance priority. The configuration data in the first group takes priority over the data in subsequent groups.
apply-groups-except	Specify which groups not to inherit configuration data from. You can specify more than one group name.
destination-address *address* [except]	Match the IPv6 destination address field unless the **except** option is included. If the option is included, do not match the IPv6 destination address field. You cannot specify both the **address** and **destination-address** match conditions in the same term.
destination-class *class-names*	Match one or more specified destination class names (sets of destination prefixes grouped together and given a class name). NOTE: This match condition is not supported on PTX series packet transport routers. For more information, see "Firewall Filter Match Conditions Based on Address Classes" on page 614.
destination-class-except *class-names*	Do not match one or more specified destination class names. For details, see the **destination-class** match condition. NOTE: This match condition is not supported on PTX series packet transport routers.

Table 40: Firewall Filter Match Conditions for IPv6 Traffic *(continued)*

Match Condition	Description
destination-port *number*	Match the UDP or TCP destination port field. You cannot specify both the **port** and **destination-port** match conditions in the same term. If you configure this match condition, we recommend that you also configure the **next-header udp** or **next-header tcp** match condition in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the following text synonyms (the port numbers are also listed): **afs** (1483), **bgp** (179), **biff** (512), **bootpc** (68), **bootps** (67), **cmd** (514), **cvspserver** (2401), **dhcp** (67), **domain** (53), **eklogin** (2105), **ekshell** (2106), **exec** (512), **finger** (79), **ftp** (21), **ftp-data** (20), **http** (80), **https** (443), **ident** (113), **imap** (143), **kerberos-sec** (88), **klogin** (543), **kpasswd** (761), **krb-prop** (754), **krbupdate** (760), **kshell** (544), **ldap** (389), **ldp** (646), **login** (513), **mobileip-agent** (434), **mobilip-mn** (435), **msdp** (639), **netbios-dgm** (138), **netbios-ns** (137), **netbios-ssn** (139), **nfsd** (2049), **nntp** (119), **ntalk** (518), **ntp** (123), **pop3** (110), **pptp** (1723), **printer** (515), **radacct** (1813), **radius** (1812), **rip** (520), **rkinit** (2108), **smtp** (25), **snmp** (161), **snmptrap** (162), **snpp** (444), **socks** (1080), **ssh** (22), **sunrpc** (111), **syslog** (514), **tacacs** (49), **tacacs-ds** (65), **talk** (517), **telnet** (23), **tftp** (69), **timed** (525), **who** (513), or **xdmcp** (177).
destination-port-except *number*	Do not match the UDP or TCP destination port field. For details, see the **destination-port** match condition.
destination-prefix-list *prefix-list-name* **[except]**	Match the IPv6 destination prefix to the specified list unless the **except** option is included. If the option is included, do not match the IPv6 destination prefix to the specified list. The prefix list is defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level.
extension-headers *header-type*	Match an extension header type that is contained in the packet by identifying a Next Header value. NOTE: This match condition is only supported on MPCs in MX Series routers. In the first fragment of a packet, the filter searches for a match in any of the extension header types. When a packet with a fragment header is found (a subsequent fragment), the filter only searches for a match of the next extension header type because the location of other extension headers is unpredictable. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **ah** (51), **destination** (60), **esp** (50), **fragment** (44), **hop-by-hop** (0), **mobility** (135), or **routing** (43). To match *any* value for the extension header option, use the text synonym **any**. For MX Series routers with MPCs, initialize new firewall filters that include this condition by walking the corresponding SNMP MIB.
extension-headers-except *header-type*	Do not match an extension header type that is contained in the packet. For details, see the **extension-headers** match condition. NOTE: This match condition is only supported on MPCs in MX Series routers.

Table 40: Firewall Filter Match Conditions for IPv6 Traffic *(continued)*

Match Condition	Description	
flexible-match-mask *value*	**bit-length**	Length of the data to be matched in bits, not needed for string input (0..128)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-mask-name	Select a flexible match from predefined template field
	mask-in-hex	Mask out bits in the packet data to be matched
	match-start	Start point to match in packet
	prefix	Value data/string to be matched
	See "Firewall Filter Flexible Match Conditions" on page 667 for details	
flexible-match-range *value*	**bit-length**	Length of the data to be matched in bits (0..32)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-range-name	Select a flexible match from predefined template field
	match-start	Start point to match in packet
	range	Range of values to be matched
	range-except	Do not match this range of values
	See "Firewall Filter Flexible Match Conditions" on page 667 for details	
forwarding-class ***class***	Match the forwarding class of the packet. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**. For information about forwarding classes and router-internal output queues, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.	
forwarding-class-except ***class***	Do not match the forwarding class of the packet. For details, see the **forwarding-class** match condition.	

Table 40: Firewall Filter Match Conditions for IPv6 Traffic *(continued)*

Match Condition	Description
hop-limit ***hop-limit***	Match the hop limit to the specified hop limit or set of hop limits. For ***hop-limit***, specify a single value or a range of values from 0 through 255. Supported on interfaces hosted on MICs or MPCs in MX Series routers only. NOTE: This match condition is supported on PTX series routers when **enhanced-mode** is configured on the router.
hop-limit-except ***hop-limit***	Do not match the hop limit to the specified hop limit or set of hop limits. For details, see the **hop-limit** match condition. Supported on interfaces hosted on MICs or MPCs in MX Series routers only. NOTE: This match condition is supported on PTX series routers when **enhanced-mode** is configured on the router.
icmp-code ***message-code***	Match the ICMP message code field. If you configure this match condition, we recommend that you also configure the **next-header icmp** or **next-header icmp6** match condition in the same term. If you configure this match condition, you must also configure the **icmp-type** ***message-type*** match condition in the same term. An ICMP message code provides more specific information than an ICMP message type, but the meaning of an ICMP message code is dependent on the associated ICMP message type. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed). The keywords are grouped by the ICMP type with which they are associated: • parameter-problem: **ip6-header-bad** (0), **unrecognized-next-header** (1), **unrecognized-option** (2) • time-exceeded: **ttl-eq-zero-during-reassembly** (1), **ttl-eq-zero-during-transit** (0) • destination-unreachable: **administratively-prohibited** (1), **address-unreachable** (3), **no-route-to-destination** (0), **port-unreachable** (4)
icmp-code-except ***message-code***	Do not match the ICMP message code field. For details, see the **icmp-code** match condition.

Table 40: Firewall Filter Match Conditions for IPv6 Traffic *(continued)*

Match Condition	Description
icmp-type ***message-type***	Match the ICMP message type field. If you configure this match condition, we recommend that you also configure the **next-header icmp** or **next-header icmp6** match condition in the same term. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **certificate-path-advertisement** (149), **certificate-path-solicitation** (148), **destination-unreachable** (1), **echo-reply** (129), **echo-request** (128), **home-agent-address-discovery-reply** (145), **home-agent-address-discovery-request** (144), **inverse-neighbor-discovery-advertisement** (142), **inverse-neighbor-discovery-solicitation** (141), **membership-query** (130), **membership-report** (131), **membership-termination** (132), **mobile-prefix-advertisement-reply** (147), **mobile-prefix-solicitation** (146), **neighbor-advertisement** (136), **neighbor-solicit** (135), **node-information-reply** (140), **node-information-request** (139), **packet-too-big** (2), **parameter-problem** (4), **private-experimentation-100** (100), **private-experimentation-101** (101), **private-experimentation-200** (200), **private-experimentation-201** (201), **redirect** (137), **router-advertisement** (134), **router-renumbering** (138), **router-solicit** (133), or **time-exceeded** (3). For **private-experimentation-201** (201), you can also specify a range of values within square brackets.
icmp-type-except ***message-type***	Do not match the ICMP message type field. For details, see the **icmp-type** match condition.
interface ***interface-name***	Match the interface on which the packet was received. NOTE: If you configure this match condition with an interface that does not exist, the term does not match any packet.
interface-group ***group-number***	Match the logical interface on which the packet was received to the specified interface group or set of interface groups. For ***group-number***, specify a single value or a range of values from **0** through **255**. To assign a logical interface to an interface group ***group-number***, specify the ***group-number*** at the **[interfaces *interface-name* unit *number* family *family* filter group]** hierarchy level. For more information, see "Filtering Packets Received on a Set of Interface Groups Overview" on page 861.
interface-group-except ***group-number***	Do not match the logical interface on which the packet was received to the specified interface group or set of interface groups. For details, see the **interface-group** match condition. NOTE: This match condition is not supported on PTX series packet transport routers.
interface-set ***interface-set-name***	Match the interface on which the packet was received to the specified interface set. To define an interface set, include the **interface-set** statement at the **[edit firewall]** hierarchy level. NOTE: This match condition is not supported on PTX series packet transport routers. For more information, see "Filtering Packets Received on an Interface Set Overview" on page 862.

Table 40: Firewall Filter Match Conditions for IPv6 Traffic *(continued)*

Match Condition	Description
ip-options *values*	Match the 8-bit IP option field, if present, to the specified value or list of values. In place of a numeric value, you can specify one of the following text synonyms (the option values are also listed): **loose-source-route** (131), **record-route** (7), **router-alert** (148), **security** (130), **stream-id** (136),**strict-source-route** (137), or **timestamp** (68). To match *any* value for the IP option, use the text synonym **any**. To match on *multiple* values, specify the list of values within square brackets ('**[**' and '**]**'). To match a *range* of values, use the value specification **[*value1-value2*]**. For example, the match condition **ip-options [0-147]** matches on an IP options field that contains the **loose-source-route**, **record-route**, or **security** values, or any other value from 0 through 147. However, this match condition does not match on an IP options field that contains only the **router-alert** value (148). For most interfaces, a filter term that specifies an **ip-option** match on one or more *specific* IP option values (a value other than **any**) causes packets to be sent to the Routing Engine so that the kernel can parse the IP option field in the packet header. • For a firewall filter term that specifies an **ip-option** match on one or more specific IP option values, you cannot specify the **count**, **log**, or **syslog** nonterminating actions *unless* you also specify the **discard** terminating action in the same term. This behavior prevents double-counting of packets for a filter applied to a transit interface on the router. • Packets processed on the kernel might be dropped in case of a system bottleneck. To ensure that matched packets are instead sent to the Packet Forwarding Engine (where packet processing is implemented in hardware), use the **ip-options any** match condition. The 10-Gigabit Ethernet Modular Port Concentrator (MPC), 100-Gigabit Ethernet MPC, 60-Gigabit Ethernet MPC, 60-Gigabit Queuing Ethernet MPC, and 60-Gigabit Ethernet Enhanced Queuing MPC on MX Series routers are capable of parsing the IP option field of the IPv4 packet header. For interfaces configured on those MPCs, *all* packets that are matched using the **ip-options** match condition are sent to the Packet Forwarding Engine for processing.
ip-options-except *values*	Do not match the IP option field to the specified value or list of values. For details about specifying the ***values***, see the **ip-options** match condition.
loss-priority *level*	Match the packet loss priority (PLP) level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. Supported on M120 and M320 routers; M7i and M10i routers with the Enhanced CFEB (CFEB-E); and MX Series routers and EX Series switches. For IP traffic on M320, MX Series, T Series routers and EX Series switches with Enhanced II Flexible PIC Concentrators (FPCs), you must include the **tri-color** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tri-color** statement is not enabled, you can only configure the **high** and **low** levels. This applies to all protocol families. NOTE: This match condition is not supported on PTX series packet transport routers. For information about the **tri-color** statement, see *Configuring and Applying Tricolor Marking Policers*. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.

Table 40: Firewall Filter Match Conditions for IPv6 Traffic *(continued)*

Match Condition	Description
loss-priority-except *level*	Do not match the PLP level. For details, see the **loss-priority** match condition. NOTE: This match condition is not supported on PTX series packet transport routers.
next-header *header-type*	Match the first 8-bit Next Header field in the packet. Support for the **next-header** firewall match condition is available in Junos OS Release 13.3R6 and later. For IPv6, we recommend that you use the **payload-protocol** term rather than the **next-header** term when configuring a firewall filter with match conditions. Although either can be used, **payload-protocol** provides the more reliable match condition because it uses the actual payload protocol to find a match, whereas **next-header** simply takes whatever appears in the first header following the IPv6 header, which may or may not be the actual protocol. In addition, if **next-header** is used with IPv6, the accelerated filter block lookup process is bypassed and the standard filter used instead. Match the first 8-bit Next Header field in the packet. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **ah** (51), **dstops** (60), **egp** (8), **esp** (50), **fragment** (44), **gre** (47), **hop-by-hop** (0), **icmp** (1), **icmp6** (58), **icmpv6** (58), **igmp** (2), **ipip** (4), **ipv6** (41), **mobility** (135), **no-next-header** (59), **ospf** (89), **pim** (103), **routing** (43), **rsvp** (46), **sctp** (132), **tcp** (6), **udp** (17), or **vrrp** (112). NOTE: **next-header icmp6** and **next-header icmpv6** match conditions perform the same function. **next-header icmp6** is the preferred option. **next-header icmpv6** is hidden in the Junos OS CLI.
next-header-except *header-type*	Do not match the 8-bit Next Header field that identifies the type of header between the IPv6 header and payload. For details, see the **next-header** match type.
packet-length *bytes*	Match the length of the received packet, in bytes. The length refers only to the IP packet, including the packet header, and does not include any Layer 2 encapsulation overhead.
packet-length-except *bytes*	Do not match the length of the received packet, in bytes. For details, see the **packet-length** match type.
payload-protocol *protocol-type*	Match the payload protocol type. In place of the ***protocol-type*** numeric value, you can specify one of the following text synonyms (the field values are also listed): specify one or a set of of the following: **ah** (51), **dstopts** (60), **egp** (8), **esp** (50), **fragment** (44), **gre** (47), **hop-by-hop** (0), **icmp** (1), **icmp6** (58, **igmp** (2), **ipip** (4), **ipv6** (41), **no-next-header**, **ospf** (89), **pim** (103), **routing**, **rsvp** (46), **sctp** (132), **tcp** (6), **udp** (17), or **vrrp** (112). You can also use the **payload-protocol** condition to match an extension header type that the Juniper Networks firmware cannot interpret. You can specify a range of extension header values within square brackets. When the firmware finds the first extension header type that it cannot interpret in a packet, the **payload-protocol** value is set to that extension header type. The firewall filter only examines the first extension header type that the firmware cannot interpret in the packet. NOTE: This match condition is only supported on MPCs on MX Series Routers. Initialize new firewall filters that include this condition by walking the corresponding SNMP MIB.
payload-protocol-except *protocol-type*	Do not match the payload protocol type. For details, see the **payload-protocol** match type. NOTE: This match condition is only supported on MPCs on MX Series Routers

Table 40: Firewall Filter Match Conditions for IPv6 Traffic *(continued)*

Match Condition	Description
port *number*	Match the UDP or TCP source or destination port field. If you configure this match condition, you cannot configure the **destination-port** match condition or the **source-port** match condition in the same term. If you configure this match condition, we recommend that you also configure the **next-header udp** or **next-header tcp** match condition in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the text synonyms listed under **destination-port**.
port-except *number*	Do not match the UDP or TCP source or destination port field. For details, see the **port** match condition.
prefix-list *prefix-list-name* **[except]**	Match the prefixes of the source or destination address fields to the prefixes in the specified list unless the **except** option is included. If the option is included, do not match the prefixes of the source or destination address fields to the prefixes in the specified list. The prefix list is defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level.
service-filter-hit	Match a packet received from a filter where a **service-filter-hit** action was applied. NOTE: This match condition is not supported on PTX series packet transport routers.
source-address *address* **[except]**	Match the IPv6 address of the source node sending the packet unless the **except** option is included. If the option is included, do not match the IPv6 address of the source node sending the packet. You cannot specify both the **address** and **source-address** match conditions in the same term.
source-class *class-names*	Match one or more specified source class names (sets of source prefixes grouped together and given a class name). NOTE: This match condition is not supported on PTX series packet transport routers. For more information, see "Firewall Filter Match Conditions Based on Address Classes" on page 614.
source-class-except *class-names*	Do not match one or more specified source class names. For details, see the **source-class** match condition. NOTE: This match condition is not supported on PTX series packet transport routers.
source-port *number*	Match the UDP or TCP source port field. You cannot specify the **port** and **source-port** match conditions in the same term. If you configure this match condition, we recommend that you also configure the **next-header udp** or **next-header tcp** match condition in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the text synonyms listed with the **destination-port** *number* match condition.
source-port-except *number*	Do not match the UDP or TCP source port field. For details, see the **source-port** match condition.

Table 40: Firewall Filter Match Conditions for IPv6 Traffic *(continued)*

Match Condition	Description
source-prefix-list ***name*** **[except]**	Match the IPv6 address prefix of the packet source field unless the **except** option is included. If the option is included, do not match the IPv6 address prefix of the packet source field. Specify a prefix list name defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level.
tcp-established	Match TCP packets other than the first packet of a connection. This is a text synonym for **tcp-flags "(ack | rst)"** (**0x14**). NOTE: This condition does not implicitly check that the protocol is TCP. To check this, specify the **protocol tcp** match condition. If you configure this match condition, we recommend that you also configure the **next-header tcp** match condition in the same term.
tcp-flags ***flags***	Match one or more of the low-order 6 bits in the 8-bit TCP flags field in the TCP header. To specify individual bit fields, you can specify the following text synonyms or hexadecimal values: • **fin** (0x01) • **syn** (0x02) • **rst** (0x04) • **push** (0x08) • **ack** (0x10) • **urgent** (0x20) In a TCP session, the SYN flag is set only in the initial packet sent, while the ACK flag is set in all packets sent after the initial packet. You can string together multiple flags using the bit-field logical operators. For combined bit-field match conditions, see the **tcp-established** and **tcp-initial** match conditions. If you configure this match condition, we recommend that you also configure the **next-header tcp** match condition in the same term to specify that the TCP protocol is being used on the port.
tcp-initial	Match the initial packet of a TCP connection. This is a text synonym for **tcp-flags "(!ack & syn)"**. This condition does not implicitly check that the protocol is TCP. If you configure this match condition, we recommend that you also configure the **next-header tcp** match condition in the same term.

Table 40: Firewall Filter Match Conditions for IPv6 Traffic *(continued)*

Match Condition	Description

traffic-class *number*	Match the 8-bit field that specifies the class-of-service (CoS) priority of the packet.
	
	This field was previously used as the type-of-service (ToS) field in IPv4.
	
	You can specify a numeric value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix.
	
	In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed):
	
	• RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*, defines one code point: **ef** (46).
	• RFC 2597, *Assured Forwarding PHB Group*, defines 4 classes, with 3 drop precedences in each class, for a total of 12 code points:
	  • **af11** (10), **af12** (12), **af13** (14)
	  • **af21** (18), **af22** (20), **af23** (22)
	  • **af31** (26), **af32** (28), **af33** (30)
	  • **af41** (34), **af42** (36), **af43** (38)

traffic-class-except *number*	Do not match the 8-bit field that specifies the CoS priority of the packet. For details, see the **traffic-class** match description.

NOTE: If you specify an IPv6 address in a match condition (the **address**, **destination-address**, or **source-address** match conditions), use the syntax for text representations described in RFC 4291, *IP Version 6 Addressing Architecture*. For more information about IPv6 addresses, see *IPv6 Overview* and *Supported IPv6 Standards*.

Release History Table

Release	Description
13.3R6	Support for the **next-header** firewall match condition is available in Junos OS Release 13.3R6 and later.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Nonterminating Actions on page 673
- Firewall Filter Match Conditions for IPv4 Traffic on page 589
- **enhanced-mode on page 1245**
- Firewall Filter Flexible Match Conditions on page 667

Firewall Filter Match Conditions for MPLS Traffic

You can configure a firewall filter with match conditions for MPLS traffic (**family mpls**).

NOTE: The **input-list** ***filter-names*** and **output-list** ***filter-names*** statements for firewall filters for the **mpls** protocol family are supported on all interfaces with the exception of management interfaces and internal Ethernet interfaces (**fxp** or **em0**), loopback interfaces (**lo0**), and USB modem interfaces (**umd**).

Table 41 on page 639 describes the ***match-conditions*** you can configure at the **[edit firewall family mpls filter** ***filter-name*** **term** ***term-name*** **from]** hierarchy level.

Table 41: Firewall Filter Match Conditions for MPLS Traffic

Match Condition	Description
apply-groups	Specify which groups to inherit configuration data from. You can specify more than one group name. You must list them in order of inheritance priority. The configuration data in the first group takes priority over the data in subsequent groups.
apply-groups-except	Specify which groups not to inherit configuration data from. You can specify more than one group name.
exp ***number***	Experimental (EXP) bit number or range of bit numbers in the MPLS header. For ***number***, you can specify one or more values from **0** through **7** in decimal, binary, or hexadecimal format. NOTE: This match condition is not supported on PTX series packet transport routers.
exp-except ***number***	Do not match on the EXP bit number or range of bit numbers in the MPLS header. For ***number***, you can specify one or more values from **0** through **7**. NOTE: This match condition is not supported on PTX series packet transport routers.
forwarding-class ***class***	Forwarding class. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.
forwarding-class-except ***class***	Do not match on the forwarding class. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.
interface ***interface-name***	Interface on which the packet was received. You can configure a match condition that matches packets based on the interface on which they were received. NOTE: If you configure this match condition with an interface that does not exist, the term does not match any packet.
interface-set ***interface-set-name***	Match the interface on which the packet was received to the specified interface set. To define an interface set, include the **interface-set** statement at the **[edit firewall]** hierarchy level. NOTE: This match condition is not supported on PTX series packet transport routers. For more information, see “Filtering Packets Received on an Interface Set Overview” on page 862.

Table 41: Firewall Filter Match Conditions for MPLS Traffic *(continued)*

Match Condition	Description
ip-version *number*	(Interfaces on Enhanced Scaling flexible PIC concentrators [FPCs] on supported T Series routers only) Inner IP version. To match MPLS-tagged IPv4 packets, match on the text synonym **ipv4**. NOTE: This match condition is not supported on PTX series packet transport routers.
loss-priority *level*	Match the packet loss priority (PLP) level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. Supported on M120 and M320 routers; M7i and M10i routers with the Enhanced CFEB (CFEB-E); and MX Series routers and EX Series switches. For IP traffic on M320, MX Series, and T Series routers with Enhanced II Flexible PIC Concentrators (FPCs), and EX Series switches, you must include the **tri-color** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tri-color** statement is not enabled, you can only configure the **high** and **low** levels. This applies to all protocol families. For information about the **tri-color** statement, see *Configuring and Applying Tricolor Marking Policers*. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.
loss-priority-except *level*	Do not match the PLP level. For details, see the **loss-priority** match condition. NOTE: This match condition is not supported on PTX series packet transport routers.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Nonterminating Actions on page 673

Firewall Filter Match Conditions for MPLS-Tagged IPv4 or IPv6 Traffic

This topic covers the following information:

- Matching on IPv4 or IPv6 Packet Header Address or Port Fields in MPLS Flows on page 640
- IP Address Match Conditions for MPLS Traffic on page 641
- IP Port Match Conditions for MPLS Traffic on page 642

Matching on IPv4 or IPv6 Packet Header Address or Port Fields in MPLS Flows

To support network-based service in a core network, you can configure a firewall filter that matches Internet Protocol version 4 (IPv4) or version 6 (IPv6) packet header fields in MPLS traffic (**family mpls**). The firewall filter can match IPv4 or IPv6 packets as an inner payload of an MPLS packet that has a single MPLS label or up to five MPLS labels stacked together. You can configure match conditions based on IPv4 addresses and IPv4 port numbers or IPv6 addresses and IPv6 port numbers in the header.

Firewall filters based on MPLS-tagged IPv4 headers are supported for interfaces on Enhanced Scaling flexible PIC concentrators (FPCs) on T320, T640, T1600, TX Matrix, and TX Matrix Plus routers and switches only. However, the firewall filters based on MPLS-tagged IPv6 headers are supported for interfaces on the Type 5 FPC on T4000 Core Routers only. The feature is not supported for the router or switch loopback interface (**lo0**), the router or switch management interface (**fxp0** or **em0**), or USB modem interfaces (**umd**).

To configure a firewall filter term that matches an address or port fields in the Layer 4 header of packets in an MPLS flow, you use the **ip-version ipv4** match condition to specify that the term is to match packets based on inner IP fields:

- To match an MPLS-tagged IPv4 packet on the source or destination address field in the IPv4 header, specify the match condition at the **[edit firewall family mpls filter *filter-name* term *term-name* from ip-version ipv4]** hierarchy level.
- To match an MPLS-tagged IPv4 packet on the source or destination port field in the Layer 4 header, specify the match condition at the **[edit firewall family mpls filter *filter-name* term *term-name* from ip-version ipv4 protocol (udp | tcp)]** hierarchy level.

To configure a firewall filter term that matches an address or port fields in the IPv6 header of packets in an MPLS flow, you use the **ip-version ipv6** match condition to specify that the term is to match packets based on inner IP fields:

- To match an MPLS-tagged IPv6 packet on the source or destination address field in the IPv6 header, specify the match condition at the **[edit firewall family mpls filter *filter-name* term *term-name* from ip-version ipv6]** hierarchy level.
- To match an MPLS-tagged IPv6 packet on the source or destination port field in the Layer 4 header, specify the match condition at the **[edit firewall family mpls filter *filter-name* term *term-name* from ip-version ipv6 protocol (udp | tcp)]** hierarchy level.

IP Address Match Conditions for MPLS Traffic

Table 42 on page 641 describes the IP address-specific match conditions you can configure at the **[edit firewall family mpls filter *filter-name* term *term-name* from ip-version *ip-version*]** hierarchy level.

Table 42: IP Address-Specific Firewall Filter Match Conditions for MPLS Traffic

Match Condition	Description
destination-address *address*	Match the address of the destination node to receive the packet.
destination-address *address* except	Do not match the address of the destination node to receive the packet.

Table 42: IP Address-Specific Firewall Filter Match Conditions for MPLS Traffic *(continued)*

Match Condition	Description
protocol *number*	Match the IP protocol type field. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **ah** (51), **dstopts** (60), **egp** (8), **esp** (50), **fragment** (44), **gre** (47), **hop-by-hop** (0), **icmp** (1), **icmp6** (58), **icmpv6** (58), **igmp** (2), **ipip** (4), **ipv6** (41), **ospf** (89), **pim** (103), **rsvp** (46), **sctp** (132), **tcp** (6), **udp** (17), or **vrrp** (112).
source-address *address*	Match the address of the source node sending the packet.
source-address *address* **except**	Do not match the address of the source node sending the packet.

IP Port Match Conditions for MPLS Traffic

Table 43 on page 642 describes the IP port-specific ***match-conditions*** you can configure at the **[edit firewall family mpls filter** ***filter-name*** **term** ***term-name*** **from ip-version** ***ip-version*** **protocol (udp | tcp)]** hierarchy level.

Table 43: IP Port-Specific Firewall Filter Match Conditions for MPLS Traffic

Match Condition	Description
destination-port *number*	Match on the UDP or TCP destination port field. In place of the numeric value, you can specify one of the following text synonyms (the port numbers are also listed): **afs** (1483), **bgp** (179), **biff** (512), **bootpc** (68), **bootps** (67), **cmd** (514), **cvspserver** (2401), **dhcp** (67), **domain** (53), **eklogin** (2105), **ekshell** (2106), **exec** (512), **finger** (79), **ftp** (21), **ftp-data** (20), **http** (80), **https** (443), **ident** (113), **imap** (143), **kerberos-sec** (88), **klogin** (543), **kpasswd** (761), **krb-prop** (754), **krbupdate** (760), **kshell** (544), **ldap** (389), **ldp** (646), **login** (513), **mobileip-agent** (434), **mobilip-mn** (435), **msdp** (639), **netbios-dgm** (138), **netbios-ns** (137), **netbios-ssn** (139), **nfsd** (2049), **nntp** (119), **ntalk** (518), **ntp** (123), **pop3** (110), **pptp** (1723), **printer** (515), **radacct** (1813), **radius** (1812), **rip** (520), **rkinit** (2108), **smtp** (25), **snmp** (161), **snmptrap** (162), **snpp** (444), **socks** (1080), **ssh** (22), **sunrpc** (111), **syslog** (514), **tacacs** (49), **tacacs-ds** (65), **talk** (517), **telnet** (23), **tftp** (69), **timed** (525), **who** (513), or **xdmcp** (177).
destination-port-except *number*	Do not match on the UDP or TCP destination port field. In place of the numeric value, you can specify one of the text synonyms listed with the **destination-port** match condition.
source-port *number*	Match on the TCP or UDP source port field. In place of the numeric field, you can specify one of the text synonyms listed under **destination-port**.

Table 43: IP Port-Specific Firewall Filter Match Conditions for MPLS Traffic *(continued)*

Match Condition	Description
source-port-except *number*	Do not match on the TCP or UDP source port field.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Nonterminating Actions on page 673

Firewall Filter Match Conditions for VPLS Traffic

In the **from** statement in the VPLS filter term, you specify conditions that the packet must match for the action in the **then** statement to be taken. All conditions in the **from** statement must match for the action to be taken. The order in which you specify match conditions is not important, because a packet must match all the conditions in a term for a match to occur.

If you specify no match conditions in a term, that term matches all packets.

An individual condition in a **from** statement can contain a list of values. For example, you can specify numeric ranges. You can also specify multiple source addresses or destination addresses. When a condition defines a list of values, a match occurs if one of the values in the list matches the packet.

Individual conditions in a **from** statement can be negated. When you negate a condition, you are defining an explicit mismatch. For example, the negated match condition for **forwarding-class** is **forwarding-class-except**. If a packet matches a negated condition, it is immediately considered not to match the **from** statement, and the next term in the filter is evaluated, if there is one. If there are no more terms, the packet is discarded.

You can configure a firewall filter with match conditions for Virtual Private LAN Service (VPLS) traffic (**family vpls**). Table 44 on page 644 describes the ***match-conditions*** you can configure at the **[edit firewall family vpls filter *filter-name* term *term-name* from]** hierarchy level.

NOTE: Not all match conditions for VPLS traffic are supported on all routing platforms or switching platforms. A number of match conditions for VPLS traffic are supported only on MX Series 3D Universal Edge Routers.

In the VPLS documentation, the word *router* in terms such as *PE router* is used to refer to any device that provides routing functions.

Table 44: Firewall Filter Match Conditions for VPLS Traffic

Match Condition	Description
destination-mac-address ***address***	Match the destination media access control (MAC) address of a VPLS packet.
destination-port ***number***	(MX Series routers and EX Series switches only) Match the UDP or TCP destination port field. You cannot specify both the **port** and **destination-port** match conditions in the same term. In place of the numeric value, you can specify one of the following text synonyms (the port numbers are also listed): **afs** (1483), **bgp** (179), **biff** (512), **bootpc** (68), **bootps** (67), **cmd** (514), **cvspserver** (2401), **dhcp** (67), **domain** (53), **eklogin** (2105), **ekshell** (2106), **exec** (512), **finger** (79), **ftp** (21), **ftp-data** (20), **http** (80), **https** (443), **ident** (113), **imap** (143), **kerberos-sec** (88), **klogin** (543), **kpasswd** (761), **krb-prop** (754), **krbupdate** (760), **kshell** (544), **ldap** (389), **ldp** (646), **login** (513), **mobileip-agent** (434), **mobilip-mn** (435), **msdp** (639), **netbios-dgm** (138), **netbios-ns** (137), **netbios-ssn** (139), **nfsd** (2049), **nntp** (119), **ntalk** (518), **ntp** (123), **pop3** (110), **pptp** (1723), **printer** (515), **radacct** (1813), **radius** (1812), **rip** (520), **rkinit** (2108), **smtp** (25), **snmp** (161), **snmptrap** (162), **snpp** (444), **socks** (1080), **ssh** (22), **sunrpc** (111), **syslog** (514), **tacacs** (49), **tacacs-ds** (65), **talk** (517), **telnet** (23), **tftp** (69), **timed** (525), **who** (513), or **xdmcp** (177).
destination-port-except ***number***	(MX Series routers and EX Series switches only) Do not match on the TCP or UDP destination port field. You cannot specify both the **port** and **destination-port** match conditions in the same term.
destination-prefix-list ***name***	(ACX Series routers, MX Series routers, and EX Series switches only) Match destination prefixes in the specified list. Specify the name of a prefix list defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level. NOTE: VPLS prefix lists support only IPv4 addresses. IPv6 addresses included in a VPLS prefix list will be discarded.
destination-prefix-list ***name*** **except**	(MX Series routers and EX Series switches only) Do not match destination prefixes in the specified list. For more information, see the **destination-prefix-list** match condition.
dscp ***number***	(MX Series routers and EX Series switches only) Match the Differentiated Services code point (DSCP). The DiffServ protocol uses the type-of-service (ToS) byte in the IP header. The most significant 6 bits of this byte form the DSCP. For more information, see the *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*. You can specify a numeric value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): • RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*, defines one code point: **ef** (46). • RFC 2597, *Assured Forwarding PHB Group*, defines 4 classes, with 3 drop precedences in each class, for a total of 12 code points: **af11** (10), **af12** (12), **af13** (14), **af21** (18), **af22** (20), **af23** (22), **af31** (26), **af32** (28), **af33** (30), **af41** (34), **af42** (36), **af43** (38)

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description	
dscp-except ***number***	(MX Series routers and EX Series switches only) Do not match on the DSCP. For details, see the **dscp** match condition.	
ether-type ***values***	Match the 2-octet IEEE 802.3 Length/EtherType field to the specified value or list of values. You can specify decimal or hexadecimal values from 0 through 65535 (0xFFFF). A value from 0 through 1500 (0x05DC) specifies the length of an Ethernet Version 1 frame. A value from 1536 (0x0600) through 65535 specifies the EtherType (nature of the MAC client protocol) of an Ethernet Version 2 frame. In place of the numeric value, you can specify one of the following text synonyms (the hexadecimal values are also listed): **aarp** (0x80F3), **appletalk** (0x809B), **arp** (0x0806), **ipv4** (0x0800), **ipv6** (0x86DD), **mpls-multicast** (0x8848), **mpls-unicast** (0x8847), **oam** (0x8902), **ppp** (0x880B), **pppoe-discovery** (0x8863), **pppoe-session** (0x8864), or **sna** (0x80D5).	
ether-type-except ***values***	Do not match the 2-octet Length/EtherType field to the specified value or list of values. For details about specifying the ***values***, see the **ether-type** match condition.	
flexible-match-mask *value*	**bit-length**	Starting in Junos OS 14.2, flexible offset filters are supported in firewall hierarchy configurations. Length of the data to be matched in bits, not needed for string input (0..128)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-mask-name	Select a flexible match from predefined template field
	mask-in-hex	Mask out bits in the packet data to be matched
	match-start	Start point to match in packet
	prefix	Value data/string to be matched

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description	
flexible-match-range *value*	**bit-length**	Length of the data to be matched in bits (0..32)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-range-name	Select a flexible match from predefined template field
	match-start	Start point to match in packet
	range	Range of values to be matched
	range-except	Do not match this range of values
forwarding-class *class*	Match the forwarding class. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.	
forwarding-class-except *class*	Do not match the forwarding class. For details, see the **forwarding-class** match condition.	
icmp-code *message-code*	Match the ICMP message code field. If you configure this match condition, we recommend that you also configure the **next-header icmp** or **next-header icmp6** match condition in the same term. If you configure this match condition, you must also configure the **icmp-type** *message-type* match condition in the same term. An ICMP message code provides more specific information than an ICMP message type, but the meaning of an ICMP message code is dependent on the associated ICMP message type. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed). The keywords are grouped by the ICMP type with which they are associated: • parameter-problem: **ip6-header-bad** (0), **unrecognized-next-header** (1), **unrecognized-option** (2) • time-exceeded: **ttl-eq-zero-during-reassembly** (1), **ttl-eq-zero-during-transit** (0) • destination-unreachable: **address-unreachable** (3), **administratively-prohibited** (1), **no-route-to-destination** (0), **port-unreachable** (4)	
icmp-code-except *message-code*	Do not match the ICMP message code field. For details, see the **icmp-code** match condition.	

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description
icmp-code ***number***	(MX Series routers and EX Series switches only) Match the ICMP message code field. If you configure this match condition, we recommend that you also configure the **ip-protocol icmp** or **ip-protocol icmp6** match condition in the same term. If you configure this match condition, you must also configure the **icmp-type** ***message-type*** match condition in the same term. An ICMP message code provides more specific information than an ICMP message type, but the meaning of an ICMP message code is dependent on the associated ICMP message type. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed). The keywords are grouped by the ICMP type with which they are associated: • parameter-problem: **ip6-header-bad** (0), **unrecognized-next-header** (1), **unrecognized-option** (2) • time-exceeded: **ttl-eq-zero-during-reassembly** (1), **ttl-eq-zero-during-transit** (0) • destination-unreachable: **address-unreachable** (3), **administratively-prohibited** (1), **no-route-to-destination** (0), **port-unreachable** (4)
icmp-code-except ***number***	(MX Series routers and EX Series switches only) Do not match on the ICMP code field. For details, see the **icmp-code** match condition.
interface ***interface-name***	Interface on which the packet was received. You can configure a match condition that matches packets based on the interface on which they were received. NOTE: If you configure this match condition with an interface that does not exist, the term does not match any packet.
interface-group ***group-number***	Match the logical interface on which the packet was received to the specified interface group or set of interface groups. For ***group-number***, specify a single value or a range of values from **0** through **255**. To assign a logical interface to an interface group ***group-number***, specify the ***group-number*** at the **[interfaces** ***interface-name*** **unit** ***number*** **family** ***family*** **filter group]** hierarchy level. For more information, see "Filtering Packets Received on a Set of Interface Groups Overview" on page 861. NOTE: This match condition is not supported on T4000 Type 5 FPCs.
interface-group-except ***group-name***	Do not match the logical interface on which the packet was received to the specified interface group or set of interface groups. For details, see the **interface-group** match condition. NOTE: This match condition is not supported on T4000 Type 5 FPCs.
interface-set ***interface-set-name***	Match the interface on which the packet was received to the specified interface set. To define an interface set, include the **interface-set** statement at the **[edit firewall]** hierarchy level. For more information, see "Filtering Packets Received on an Interface Set Overview" on page 862.
ip-address ***address***	(MX Series routers and EX Series switches only) 32-bit address that supports the standard syntax for IPv4 addresses. Note that when using this term, the match condition ether-type IPv4 must be defined on the same term.

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description
ip-destination-address ***address***	(MX Series routers and EX Series switches only) 32-bit address that is the final destination node address for the packet. Note that when using this term, the match condition ether-type IPv4 must be defined on the same term.
ip-precedence ***ip-precedence-field***	(MX Series routers and EX Series switches only) IP precedence field. In place of the numeric field value, you can specify one of the following text synonyms (the field values are also listed): **critical-ecp** (0xa0), **flash** (0x60), **flash-override** (0x80), **immediate** (0x40), **internet-control** (0xc0), **net-control** (0xe0), **priority** (0x20), or **routine** (0x00).
ip-precedence-except ***ip-precedence-field***	(MX Series routers and EX Series switches only) Do not match on the IP precedence field.
ip-protocol ***number***	(MX Series routers and EX Series switches only) IP protocol field.
ip-protocol-except ***number***	(MX Series routers and EX Series switches only) Do not match on the IP protocol field.
ip-source-address ***address***	(MX Series routers and EX Series switches only) IP address of the source node sending the packet. Note that when using this term, the match condition ether-type IPv4 must also be defined on the same term.
ipv6-source-prefix-list *named-list*	(MX Series only) Match the IPv6 source address in a *named-list*.
ipv6-address *address*	(MX Series and EX9200 only) 128-bit address that supports the standard syntax for IPv6 addresses. Starting in Junos OS 14.2, firewall family bridge IPv6 match criteria is supported on MX Series and EX9200 switches.
ipv6-destination-address *address*	((MX Series and EX9200 only) 128-bit address that is the final destination node address for this packet. Note that when using this term, the match condition **ether-type IPv6** must be defined on the same term.
ipv6-destination-prefix-list *named-list*	(MX Series only) Match the IPv6 destination addresses in a *named-list*.

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description
ipv6-next-header *protocol*	(MX Series only) Match IPv6 next header protocol type. The following list shows the supported values for *protocol*: • **ah**—IP Security authentication header • **dstopts**—IPv6 destination options • **egp**—Exterior gateway protocol • **esp**—IPSec Encapsulating Security Payload • **fragment**—IPv6 fragment header • **gre**—Generic routing encapsulation • **hop-by-hop**—IPv6 hop by hop options • **icmp**—Internet Control Message Protocol • **icmp6**—Internet Control Message Protocol Version 6 • **igmp**—Internet Group Management Protocol • **ipip**—IP in IP • **ipv6**—IPv6 in IP • **no-next-header**—IPv6 no next header • **ospf**—Open Shortest Path First • **pim**—Protocol Independent Multicast • **routing**—IPv6 routing header • **rsvp**—Resource Reservation Protocol • **sctp**—Stream Control Transmission Protocol • **tcp**—Transmission Control Protocol • **udp**—User Datagram Protocol • **vrrp**—Virtual Router Redundancy Protocol
ipv6-next-header-except *protocol*	(MX Series only) Do not match the IPv6 next header protocol type.

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description
ipv6-payload-protocol *protocol*	(MX Series only) Match IPv6 payload protocol type. The following list shows the supported values for *protocol*: • **ah**—IP Security authentication header • **dstopts**—IPv6 destination options • **egp**—Exterior gateway protocol • **esp**—IPSec Encapsulating Security Payload • **fragment**—IPv6 fragment header • **gre**—Generic routing encapsulation • **hop-by-hop**—IPv6 hop by hop options • **icmp**—Internet Control Message Protocol • **icmp6**—Internet Control Message Protocol Version 6 • **igmp**—Internet Group Management Protocol • **ipip**—IP in IP • **ipv6**—IPv6 in IP • **no-next-header**—IPv6 no next header • **ospf**—Open Shortest Path First • **pim**—Protocol Independent Multicast • **routing**—IPv6 routing header • **rsvp**—Resource Reservation Protocol • **sctp**—Stream Control Transmission Protocol • **tcp**—Transmission Control Protocol • **udp**—User Datagram Protocol • **vrrp**—Virtual Router Redundancy Protocol
ipv6-payload-protocol-except *protocol*	(MX Series only) Do not match the IPv6 payload protocol.
ipv6-prefix-list *named-list*	(MX Series only) Match the IPv6 address in a *named-list*.
ipv6-source-address *address*	(MX Series only) 128-bit address that is the originating source node address for this packet.

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description
ipv6-traffic-class *number*	(MX Series only) Differentiated Services code point (DSCP). The DiffServ protocol uses the type-of-service (ToS) byte in the IP header. The most significant 6 bits of this byte form the DSCP. For more information, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*. You can specify a numeric value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): • RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*, defines one code point: **ef** (46). • RFC 2597, *Assured Forwarding PHB Group*, defines 4 classes, with 3 drop precedences in each class, for a total of 12 code points: **af11** (10), **af12** (12), **af13** (14), **af21** (18), **af22** (20), **af23** (22), **af31** (26), **af32** (28), **af33** (30), **af41** (34), **af42** (36), **af43** (38)
ipv6-traffic-class-except *number*	Do not match the DSCP **number**.
learn-vlan-1p-priority *number*	(MX Series routers, M320 router, and EX Series switches only) Match on the IEEE 802.1p learned VLAN priority bits in the provider VLAN tag (the only tag in a single-tag frame with 802.1Q VLAN tags or the outer tag in a dual-tag frame with 802.1Q VLAN tags). Specify a single value or multiple values from **0** through **7**. Compare with the **user-vlan-1p-priority** match condition. NOTE: This match condition supports the presence of a control word for MX Series routers and the M320 router.
learn-vlan-1p-priority-except *number*	(MX Series routers, M320 router, and EX Series switches only) Do not match on the IEEE 802.1p learned VLAN priority bits. For details, see the **learn-vlan-1p-priority** match condition. NOTE: This match condition supports the presence of a control word for MX Series routers and the M320 router.
learn-vlan-dei	(MX Series routers and EX Series switches only) Match the user VLAN ID drop eligability indicator (DEI) bit.
learn-vlan-dei-except	(MX Series routers and EX Series switches only) Do not match the user VLAN ID DEI bit.
learn-vlan-id *number*	(MX Series routers and EX Series switches only) VLAN identifier used for MAC learning.
learn-vlan-id-except *number*	(MX Series routers and EX Series switches only) Do not match on the VLAN identifier used for MAC learning.

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description
loss-priority *level*	Packet loss priority (PLP) level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. Supported on M120 and M320 routers; M7i and M10i routers with the Enhanced CFEB (CFEB-E); and MX Series routers. For IP traffic on M320, MX Series, and T Series routers with Enhanced II Flexible PIC Concentrators (FPCs) and EX Series switches, you must include the **tri-color** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tri-color** statement is not enabled, you can only configure the **high** and **low** levels. This applies to all protocol families. For information about the **tri-color** statement and about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.
loss-priority-except *level*	Do not match on the packet loss priority level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*.
port *number*	(MX Series routers and EX Series switches only) TCP or UDP source or destination port. You cannot specify both the **port** match condition and either the **destination-port** or **source-port** match condition in the same term.
port-except *number*	(MX Series routers and EX Series switches only) Do not match on the TCP or UDP source or destination port. You cannot specify both the **port** match condition and either the **destination-port** or **source-port** match condition in the same term.
prefix-list *name*	(MX Series routers and EX Series switches only) Match the destination or source prefixes in the specified list. Specify the name of a prefix list defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level. NOTE: VPLS prefix lists support only IPV4 addresses. IPV6 addresses included in a VPLS prefix list will be discarded.
prefix-list *name* **except**	(MX Series routers and EX Series switches only) Do not match the destination or source prefixes in the specified list. For more information, see the **destination-prefix-list** match condition.
source-mac-address *address*	Source MAC address of a VPLS packet.
source-port *number*	(MX Series routers and EX Series switches only) TCP or UDP source port field. You cannot specify the **port** and **source-port** match conditions in the same term.
source-port-except *number*	(MX Series routers and EX Series switches only) Do not match on the TCP or UDP source port field. You cannot specify the **port** and **source-port** match conditions in the same term.

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description
source-prefix-list *name*	(ACX Series routers, MX Series routers, and EX Series switches only) Match the source prefixes in the specified prefix list. Specify a prefix list name defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level. NOTE: VPLS prefix lists support only IPV4 addresses. IPV6 addresses included in a VPLS prefix list will be discarded.
source-prefix-list *name* **except**	(MX Series routers and EX Series switches only) Do not match the source prefixes in the specified prefix list. For more information, see the **source-prefix-list** match condition.
tcp-flags *flags*	Match one or more of the low-order 6 bits in the 8-bit TCP flags field in the TCP header. To specify individual bit fields, you can specify the following text synonyms or hexadecimal values: • **fin** (0x01) • **syn** (0x02) • **rst** (0x04) • **push** (0x08) • **ack** (0x10) • **urgent** (0x20) In a TCP session, the SYN flag is set only in the initial packet sent, while the ACK flag is set in all packets sent after the initial packet. You can string together multiple flags using the bit-field logical operators. If you configure this match condition for IPv6 traffic, we recommend that you also configure the **next-header tcp** match condition in the same term to specify that the TCP protocol is being used on the port.
traffic-type *type-name*	(MX Series routers and EX Series switches only) Traffic type. Specify **broadcast**, **multicast**, **unknown-unicast**, or **known-unicast**.
traffic-type-except *type-name*	(MX Series routers and EX Series switches only) Do not match on the traffic type. Specify **broadcast**, **multicast**, **unknown-unicast**, or **known-unicast**.
user-vlan-1p-priority *number*	(MX Series routers, M320 router, and EX Series switches only) Match on the IEEE 802.1p user priority bits in the customer VLAN tag (the inner tag in a dual-tag frame with 802.1Q VLAN tags). Specify a single value or multiple values from **0** through **7**. Compare with the **learn-vlan-1p-priority** match condition. NOTE: This match condition supports the presence of a control word for MX Series routers and the M320 router.
user-vlan-1p-priority-except *number*	(MX Series routers, M320 rouer, and EX Series switches only) Do not match on the IEEE 802.1p user priority bits. For details, see the **user-vlan-1p-priority** match condition. NOTE: This match condition supports the presence of a control word for MX Series routers and the M320 router.
user-vlan-id *number*	(MX Series routers and EX Series switches only) Match the first VLAN identifier that is part of the payload.

Table 44: Firewall Filter Match Conditions for VPLS Traffic *(continued)*

Match Condition	Description
user-vlan-id-except ***number***	(MX Series routers and EX Series switches only) Do not match on the first VLAN identifier that is part of the payload.
vlan-ether-type ***value***	VLAN Ethernet type field of a VPLS packet.
vlan-ether-type-except ***value***	Do not match on the VLAN Ethernet type field of a VPLS packet.

Release History Table

Release	Description
14.2	Starting in Junos OS 14.2, flexible offset filters are supported in firewall hierarchy configurations.
14.2	Starting in Junos OS 14.2, firewall family bridge IPv6 match criteria is supported on MX Series and EX9200 switches.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Nonterminating Actions on page 673

Firewall Filter Match Conditions for Layer 2 CCC Traffic

You can configure a firewall filter with match conditions for Layer 2 circuit cross-connect (CCC) traffic (**family ccc**).

The following restrictions apply to firewall filters for Layer 2 CCC traffic:

- The **input-list** ***filter-names*** and **output-list** ***filter-names*** statements for firewall filters for the **ccc** protocol family are supported on all interfaces with the exception of management interfaces and internal Ethernet interfaces (**fxp** or **em0**), loopback interfaces (**lo0**), and USB modem interfaces (**umd**).
- Only on MX Series routers and EX Series switches, you cannot apply a Layer 2 CCC stateless firewall filter (a firewall filter configured at the **[edit firewall filter family ccc]** hierarchy level) as an output filter. On MX Series routers and EX Series switches, firewall filters configured for the **family ccc** statement can be applied only as input filters.

Table 45 on page 655 describes the ***match-conditions*** you can configure at the **[edit firewall family ccc filter** ***filter-name*** **term** ***term-name*** **from]** hierarchy level.

Table 45: Firewall Filter Match Conditions for Layer 2 CCC Traffic

Match Condition	Description	
apply-groups	Specify which groups to inherit configuration data from. You can specify more than one group name. You must list them in order of inheritance priority. The configuration data in the first group takes priority over the data in subsequent groups.	
apply-groups-except	Specify which groups not to inherit configuration data from. You can specify more than one group name.	
destination-mac-address ***address***	(MX Series routers and EX Series switches only) Match the destination media access control (MAC) address of a virtual private LAN service (VPLS) packet. To have packets correctly evaluated by this match condition when applied to egress traffic flowing over a CCC circuit from a logical interface on an I-chip DPC in a Layer 2 virtual private network (VPN) routing instance, you must make a configuration change to the Layer 2 VPN routing instance. You must explicitly disable the use of a control word for traffic flowing out over a Layer 2 circuit. The use of a control word is enabled by default for Layer 2 VPN routing instances to support the emulated virtual circuit (VC) encapsulation for Layer 2 circuits. To explicitly disable the use of a control word for Layer 2 VPNs, include the **no-control-word** statement at either of the following hierarchy levels: • **[edit routing-instances *routing-instance-name* protocols l2vpn]** • **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols l2vpn]** NOTE: This match condition is not supported on PTX series packet transport routers. For more information, see *Disabling the Control Word for Layer 2 VPNs*.	
flexible-match-mask *value*	**bit-length**	Length of the data to be matched in bits, not needed for string input (0..128)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-mask-name	Select a flexible match from predefined template field
	mask-in-hex	Mask out bits in the packet data to be matched
	match-start	Start point to match in packet
	prefix	Value data/string to be matched

Table 45: Firewall Filter Match Conditions for Layer 2 CCC Traffic *(continued)*

Match Condition	Description	
flexible-match-range *value*	**bit-length**	Length of the data to be matched in bits (0..32)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-range-name	Select a flexible match from predefined template field
	match-start	Start point to match in packet
	range	Range of values to be matched
	range-except	Do not match this range of values
forwarding-class *class*	Forwarding class. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.	
forwarding-class-except *class*	Do not match on the forwarding class. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.	
interface-group *group-number*	Match the logical interface on which the packet was received to the specified interface group or set of interface groups. For ***group-number***, specify a single value or a range of values from **0** through **255**. To assign a logical interface to an interface group ***group-number***, specify the ***group-number*** at the **[interfaces *interface-name* unit *number* family *family* filter group]** hierarchy level. NOTE: This match condition is not supported on PTX series packet transport routers. For more information, see "Filtering Packets Received on a Set of Interface Groups Overview" on page 861.	
interface-group-except *number*	Do not match the logical interface on which the packet was received to the specified interface group or set of interface groups. For details, see the **interface-group** match condition. NOTE: This match condition is not supported on PTX series packet transport routers.	
learn-vlan-1p-priority *number*	(MX Series routers, M320 router, and EX Series switches only) Match on the IEEE 802.1p learned VLAN priority bits in the provider VLAN tag (the only tag in a single-tag frame with 802.1Q VLAN tags or the outer tag in a dual-tag frame with 802.1Q VLAN tags). Specify a single value or multiple values from **0** through **7**. Compare with the **user-vlan-1p-priority** match condition. NOTE: This match condition is not supported on PTX series packet transport routers. NOTE: This match condition supports the presence of a control word for MX Series and M320 routers.	

Table 45: Firewall Filter Match Conditions for Layer 2 CCC Traffic *(continued)*

Match Condition	Description
learn-vlan-1p-priority-except *number*	(MX Series routers, M320 router, and EX Series switches only) Do not match on the IEEE 802.1p learned VLAN priority bits. For details, see the **learn-vlan-1p-priority** match condition. NOTE: This match condition is not supported on PTX series packet transport routers. NOTE: This match condition supports the presence of a control word for MX Series and M320 routers.
loss-priority *level*	Packet loss priority (PLP) level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. Supported on M120 and M320 routers; M7i and M10i routers with the Enhanced CFEB (CFEB-E); and MX Series routers and EX Series switches. For IP traffic on M320, MX Series, and T Series routers with Enhanced II Flexible PIC Concentrators (FPCs), and EX Series switches, you must include the **tri-color** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tri-color** statement is not enabled, you can only configure the **high** and **low** levels. This applies to all protocol families. For information about the **tri-color** statement, see *Configuring and Applying Tricolor Marking Policers*. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.
loss-priority-except *level*	Do not match on the packet loss priority level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. NOTE: This match condition is not supported on PTX series packet transport routers. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*.
user-vlan-1p-priority *number*	(MX Series routers, M320 router, and EX Series switches only) Match on the IEEE 802.1p user priority bits in the customer VLAN tag (the inner tag in a dual-tag frame with 802.1Q VLAN tags). Specify a single value or multiple values from **0** through **7**. Compare with the **learn-vlan-1p-priority** match condition. NOTE: This match condition is not supported on PTX series packet transport routers. NOTE: This match condition supports the presence of a control word for MX Series and M320 routers.
user-vlan-1p-priority-except *number*	(MX Series routers, M320 router, and EX Series switches only) Do not match on the IEEE 802.1p user priority bits. For details, see the **user-vlan-1p-priority** match condition. NOTE: This match condition is not supported on PTX series packet transport routers. NOTE: This match condition supports the presence of a control word for MX Series and M320 routers.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680

- Firewall Filter Nonterminating Actions on page 673

Firewall Filter Match Conditions for Layer 2 Bridging Traffic

Only on MX Series routers and EX Series switches, you can configure a standard stateless firewall filter with match conditions for Layer 2 bridging traffic (**family bridge**). Table 46 on page 658 describes the ***match-conditions*** you can configure at the **[edit firewall family bridge filter *filter-name* term *term-name* from]** hierarchy level.

Table 46: Standard Firewall Filter Match Conditions for Layer 2 Bridging (MX Series Routers and EX Series Switches Only)

Match Condition	Description
destination-mac-address ***address***	Destination media access control (MAC) address of a Layer 2 packet in a bridging environment.
destination-port ***number***	TCP or UDP destination port field. You cannot specify both the **port** and **destination-port** match conditions in the same term.
destination-port-except	Do not match the TCP/UDP destination port.
destination-prefix-list *named-list*	Match the IP destination prefixes in a *named-list*.
dscp ***number***	Differentiated Services code point (DSCP). The DiffServ protocol uses the type-of-service (ToS) byte in the IP header. The most significant 6 bits of this byte form the DSCP. For more information, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*. You can specify a numeric value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): • RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*, defines one code point: **ef** (46). • RFC 2597, *Assured Forwarding PHB Group*, defines 4 classes, with 3 drop precedences in each class, for a total of 12 code points: **af11** (10), **af12** (12), **af13** (14), **af21** (18), **af22** (20), **af23** (22), **af31** (26), **af32** (28), **af33** (30), **af41** (34), **af42** (36), **af43** (38)
dscp-except ***number***	Do not match on the DSCP number. For more information, see the **dscp-except** match condition.

Table 46: Standard Firewall Filter Match Conditions for Layer 2 Bridging (MX Series Routers and EX Series Switches Only) *(continued)*

Match Condition	Description	
ether-type ***value***	Match the 2-octet IEEE 802.3 Length/EtherType field to the specified value or list of values. You can specify decimal or hexadecimal values from 0 through 65535 (0xFFFF). A value from 0 through 1500 (0x05DC) specifies the length of an Ethernet Version 1 frame. A value from 1536 (0x0600) through 65535 specifies the EtherType (nature of the MAC client protocol) of an Ethernet Version 2 frame. In place of the numeric value, you can specify one of the following text synonyms (the hexadecimal values are also listed): **aarp** (0x80F3), **appletalk** (0x809B), **arp** (0x0806), **ipv4** (0x0800), **ipv6** (0x86DD), **mpls-multicast** (0x8848), **mpls-unicast** (0x8847), **oam** (0x8902), **ppp** (0x880B), **pppoe-discovery** (0x8863), **pppoe-session** (0x8864), **sna** (0x80D5). NOTE: When matching on ip-address or ipv6-address, the ether-type ipv4 or ipv6, respectively, must also be specified in order to limit matches to ip traffic only.	
ether-type-except ***value***	Do not match the 2-octet IEEE 802.3 Length/EtherType field to the specified value or list of values. For details about specifying the ***values***, see the **ether-type** match condition.	
flexible-match-mask *value*	**bit-length**	Length of the data to be matched in bits, not needed for string input (0..128)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-mask-name	Select a flexible match from predefined template field
	mask-in-hex	Mask out bits in the packet data to be matched
	match-start	Start point to match in packet
	prefix	Value data/string to be matched

Table 46: Standard Firewall Filter Match Conditions for Layer 2 Bridging (MX Series Routers and EX Series Switches Only) *(continued)*

Match Condition	Description	
flexible-match-range *value*	**bit-length**	Length of the data to be matched in bits (0..32)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-range-name	Select a flexible match from predefined template field
	match-start	Start point to match in packet
	range	Range of values to be matched
	range-except	Do not match this range of values
forwarding class *class*	Forwarding class. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.	
forwarding-class-except *class*	Ethernet type field of a Layer 2 packet environment. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**.	
icmp-code *message-code*	Match the ICMP message code field. If you configure this match condition, we recommend that you also configure the **ip-protocol icmp**, **ip-protocol icmp6**, or **ip-protocol icmpv6** match condition in the same term. If you configure this match condition, you must also configure the **icmp-type** ***message-type*** match condition in the same term. An ICMP message code provides more specific information than an ICMP message type, but the meaning of an ICMP message code is dependent on the associated ICMP message type. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed). The keywords are grouped by the ICMP type with which they are associated: • parameter-problem: **ip6-header-bad** (0), **unrecognized-next-header** (1), **unrecognized-option** (2) • time-exceeded: **ttl-eq-zero-during-reassembly** (1), **ttl-eq-zero-during-transit** (0) • destination-unreachable: **address-unreachable** (3), **administratively-prohibited** (1), **no-route-to-destination** (0), **port-unreachable** (4)	
icmp-code-except *message-code*	Do not match the ICMP message code field. For details, see the **icmp-code** match condition.	

Table 46: Standard Firewall Filter Match Conditions for Layer 2 Bridging (MX Series Routers and EX Series Switches Only) *(continued)*

Match Condition	Description
icmp-type ***message-type***	Match the ICMP message type field. If you configure this match condition, we recommend that you also configure the **ip-protocol icmp**, **ip-protocol icmp6**, or **ip-protocol icmpv6** match condition in the same term. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **destination-unreachable** (1), **echo-reply** (129), **echo-request** (128), **membership-query** (130), **membership-report** (131), **membership-termination** (132), **neighbor-advertisement** (136), **neighbor-solicit** (135), **node-information-reply** (140), **node-information-request** (139), **packet-too-big** (2), **parameter-problem** (4), **redirect** (137), **router-advertisement** (134), **router-renumbering** (138), **router-solicit** (133), or **time-exceeded** (3).
icmp-type-except ***message-type***	Do not match the ICMP message type field. For details, see the **icmp-type** match condition.
interface ***interface-name***	Interface on which the packet was received. You can configure a match condition that matches packets based on the interface on which they were received. NOTE: If you configure this match condition with an interface that does not exist, the term does not match any packet.
interface-group ***group-number***	Match the logical interface on which the packet was received to the specified interface group or set of interface groups. For ***group-number***, specify a single value or a range of values from **0** through **255**. To assign a logical interface to an interface group ***group-number***, specify the ***group-number*** at the **[interfaces *interface-name* unit *number* family *family* filter group]** hierarchy level. For more information, see "Filtering Packets Received on a Set of Interface Groups Overview" on page 861.
interface-group-except ***number***	Do not match the logical interface on which the packet was received to the specified interface group or set of interface groups. For details, see the **interface-group** match condition.
interface-set ***interface-set-name***	Match the interface on which the packet was received to the specified interface set. To define an interface set, include the **interface-set** statement at the **[edit firewall]** hierarchy level. For more information, see "Filtering Packets Received on an Interface Set Overview" on page 862.
ip-address ***address***	32-bit address that supports the standard syntax for IPv4 addresses. NOTE: In order to limit matches to IPv4 traffic only, the ether-type ipv4 must also be specified in the same term.
ip-destination-address ***address***	32-bit address that is the final destination node address for the packet.
ip-precedence ***ip-precedence-field***	IP precedence field. In place of the numeric field value, you can specify one of the following text synonyms (the field values are also listed): **critical-ecp** (0xa0), **flash** (0x60), **flash-override** (0x80), **immediate** (0x40), **internet-control** (0xc0), **net-control** (0xe0), **priority** (0x20), or **routine** (0x00).

Table 46: Standard Firewall Filter Match Conditions for Layer 2 Bridging (MX Series Routers and EX Series Switches Only) *(continued)*

Match Condition	Description
ip-precedence-except *ip-precedence-field*	Do not match on the IP precedence field.
ip-protocol *number*	IP protocol field.
ip-protocol-except	Do not match the IP protocol type.
ip-source-address *address*	IP address of the source node sending the packet.
ipv6-address *address*	(MX Series only) 128-bit address that supports the standard syntax for IPv6 addresses. NOTE: In order to limit matches to IPv6 traffic only, the ether-type ipv6 must also be specified in the same term.
ipv6-destination-address *address*	(MX Series only) 128-bit address that is the final destination node address for this packet.
ipv6-destination-prefix-list *named-list*	(MX Series only) Match the IPv6 destination addresses in a *named-list*.
ipv6-next-header *protocol*	(MX Series only) Match IPv6 next header protocol type. The following list shows the supported values for *protocol*: • **ah**—IP Security authentication header • **dstopts**—IPv6 destination options • **egp**—Exterior gateway protocol • **esp**—IPSec Encapsulating Security Payload • **fragment**—IPv6 fragment header • **gre**—Generic routing encapsulation • **hop-by-hop**—IPv6 hop by hop options • **icmp**—Internet Control Message Protocol • **icmp6**—Internet Control Message Protocol Version 6 • **igmp**—Internet Group Management Protocol • **ipip**—IP in IP • **ipv6**—IPv6 in IP • **no-next-header**—IPv6 no next header • **ospf**—Open Shortest Path First • **pim**—Protocol Independent Multicast • **routing**—IPv6 routing header • **rsvp**—Resource Reservation Protocol • **sctp**—Stream Control Transmission Protocol • **tcp**—Transmission Control Protocol • **udp**—User Datagram Protocol • **vrrp**—Virtual Router Redundancy Protocol

Table 46: Standard Firewall Filter Match Conditions for Layer 2 Bridging (MX Series Routers and EX Series Switches Only) *(continued)*

Match Condition	Description
ipv6-next-header-except *protocol*	(MX Series only) Do not match the IPv6 next header protocol type.
ipv6-payload-protocol *protocol*	(MX Series only) Match IPv6 payload protocol type. The following list shows the supported values for *protocol*: • **ah**—IP Security authentication header • **dstopts**—IPv6 destination options • **egp**—Exterior gateway protocol • **esp**—IPSec Encapsulating Security Payload • **fragment**—IPv6 fragment header • **gre**—Generic routing encapsulation • **hop-by-hop**—IPv6 hop by hop options • **icmp**—Internet Control Message Protocol • **icmp6**—Internet Control Message Protocol Version 6 • **igmp**—Internet Group Management Protocol • **ipip**—IP in IP • **ipv6**—IPv6 in IP • **no-next-header**—IPv6 no next header • **ospf**—Open Shortest Path First • **pim**—Protocol Independent Multicast • **routing**—IPv6 routing header • **rsvp**—Resource Reservation Protocol • **sctp**—Stream Control Transmission Protocol • **tcp**—Transmission Control Protocol • **udp**—User Datagram Protocol • **vrrp**—Virtual Router Redundancy Protocol
ipv6-payload-protocol-except *protocol*	(MX Series only) Do not match the IPv6 payload protocol.
ipv6-prefix-list *named-list*	(MX Series only) Match the IPv6 address in a *named-list*.
ipv6-source-address *address*	(MX Series only) 128-bit address that is the originating source node address for this packet.
ipv6-source-prefix-list *named-list*	(MX Series only) Match the IPv6 source address in a *named-list*.

Table 46: Standard Firewall Filter Match Conditions for Layer 2 Bridging (MX Series Routers and EX Series Switches Only) *(continued)*

Match Condition	Description
ipv6-traffic-class *number*	(MX Series only) Differentiated Services code point (DSCP). The DiffServ protocol uses the type-of-service (ToS) byte in the IP header. The most significant 6 bits of this byte form the DSCP. For more information, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*. You can specify a numeric value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): • RFC 3246, *An Expedited Forwarding PHB (Per-Hop Behavior)*, defines one code point: **ef** (46). • RFC 2597, *Assured Forwarding PHB Group*, defines 4 classes, with 3 drop precedences in each class, for a total of 12 code points: **af11** (10), **af12** (12), **af13** (14), **af21** (18), **af22** (20), **af23** (22), **af31** (26), **af32** (28), **af33** (30), **af41** (34), **af42** (36), **af43** (38)
ipv6-traffic-class-except *number*	Do not match the DSCP **number**.
isid *number*	(Supported with Provider Backbone Bridging [PBB]) Match internet service identifier.
isid-dei *number*	(Supported with PBB) Match the Internet service identifier drop eligibility indicator (DEI) bit.
isid-dei-except *number*	(Supported with PBB) Do not match the Internet service identifier DEI bit.
isid-priority-code-point *number*	(Supported with PBB) Match the Internet service identifier priority code point.
isid-priority-code-point-except *number*	(Supported with PBB) Do not match the Internet service identifier priority code point.
learn-vlan-1p-priority *value*	(MX Series routers and EX Series switches only) Match on the IEEE 802.1p learned VLAN priority bits in the provider VLAN tag (the only tag in a single-tag frame with 802.1Q VLAN tags or the outer tag in a dual-tag frame with 802.1Q VLAN tags). Specify a single value or multiple values from **0** through **7**. Compare with the **user-vlan-1p-priority** match condition.
learn-vlan-1p-priority-except *value*	(MX Series routers and EX Series switches only) Do not match on the IEEE 802.1p learned VLAN priority bits. For details, see the **learn-vlan-1p-priority** match condition.
learn-vlan-dei *number*	(Supported with bridging) Match user virtual LAN (VLAN) identifier DEI bit.
learn-vlan-dei-except *number*	(Supported with bridging) Do not match user VLAN identifier DEI bit.

Table 46: Standard Firewall Filter Match Conditions for Layer 2 Bridging (MX Series Routers and EX Series Switches Only) *(continued)*

Match Condition	Description
learn-vlan-id *number*	VLAN identifier used for MAC learning.
learn-vlan-id-except *number*	Do not match on the VLAN identifier used for MAC learning.
loss-priority *level*	Packet loss priority (PLP) level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. Supported on M120 and M320 routers; M7i and M10i routers with the Enhanced CFEB (CFEB-E); and MX Series routers and EX Series switches. For IP traffic on M320, MX Series, and T Series routers with Enhanced II Flexible PIC Concentrators (FPCs), and EX Series switches, you must include the **tri-color** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tri-color** statement is not enabled, you can only configure the **high** and **low** levels. This applies to all protocol families. For information about the **tri-color** statement, see *Configuring and Applying Tricolor Marking Policers*. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.
loss-priority-except *level*	Do not match on the packet loss priority level. Specify a single level or multiple levels: **low**, **medium-low**, **medium-high**, or **high**. For information about using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see the *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*.
port *number*	TCP or UDP source or destination port. You cannot specify both the **port** match condition and either the **destination-port** or **source-port** match conditions in the same term.
source-mac-address *address*	Source MAC address of a Layer 2 packet.
source-port *number*	TCP or UDP source port field. You cannot specify the **port** and **source-port** match conditions in the same term.
source-port-except	Do not match the TCP/UDP source port.

Table 46: Standard Firewall Filter Match Conditions for Layer 2 Bridging (MX Series Routers and EX Series Switches Only) *(continued)*

Match Condition	Description
tcp-flags *flags*	Match one or more of the low-order 6 bits in the 8-bit TCP flags field in the TCP header. To specify individual bit fields, you can specify the following text synonyms or hexadecimal values: • **fin** (0x01) • **syn** (0x02) • **rst** (0x04) • **push** (0x08) • **ack** (0x10) • **urgent** (0x20) In a TCP session, the SYN flag is set only in the initial packet sent, while the ACK flag is set in all packets sent after the initial packet. You can string together multiple flags using the bit-field logical operators. Configuring the **tcp-flags** match condition requires that you configure the **next-header-tcp** match condition.
traffic-type *type*	Traffic type. Specify **broadcast**, **multicast**, **unknown-unicast**, or **known-unicast**.
traffic-type-except *type*	Do not match on the traffic type.
user-vlan-1p-priority *value*	(MX Series routers and EX Series switches only) Match on the IEEE 802.1p user priority bits in the customer VLAN tag (the inner tag in a dual-tag frame with 802.1Q VLAN tags). Specify a single value or multiple values from **0** through **7**. Compare with the **learn-vlan-1p-priority** match condition.
user-vlan-1p-priority-except *value*	(MX Series routers and EX Series switches only) Do not match on the IEEE 802.1p user priority bits. For details, see the **user-vlan-1p-priority** match condition.
user-vlan-id *number*	(MX Series routers and EX Series switches only) Match the first VLAN identifier that is part of the payload.
user-vlan-id-except *number*	(MX Series routers and EX Series switches only) Do not match on the first VLAN identifier that is part of the payload.
vlan-ether-type *value*	VLAN Ethernet type field of a Layer 2 bridging packet.
vlan-ether-type-except *value*	Do not match on the VLAN Ethernet type field of a Layer 2 bridging packet.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Nonterminating Actions on page 673

Firewall Filter Flexible Match Conditions

Standard firewall filter match conditions vary based on the protocol family of the traffic being matched. For example, the terms available for bridge protocol traffic are different from those available for the inet or inet6 protocol families. The fields available for matching witihin each protocol family are, however, fixed or pre-defined. This means that filters can match on patterns within those pre-defined fields only.

Using flexible match conditions, firewall filters can be constructed that start the match at layer-2, layer-3, layer-4 or payload locations. From there, additional offset criteria can be specified thereby enabling pattern matches at custom, user-defined locations witihin a packet.

Flexible match filter terms are applied to MPC or MIC interfaces as either input or output filters just as any other firewall filter terms. Flexible match filter terms can also be created as templates at the **[edit firewall]** hierarchy level. These templates can then be referenced within a flexible match term.

NOTE: Flexible match conditions are only supported on MX Series routers with MPCs or MICs. For environments in which FPCs, PICs, and or DPCs are installed along with MPCs or MICs, care must be taken to ensure that flexible match firewall filter criteria are applied only to the MPC or MIC interfaces.

For MX Series routers with MPCs, you need to initialize certain new firewall filters by walking the corresponding SNMP MIB, for example, **show snmp mib walk *name* ascii**. This forces Junos to learn the filter counters and ensure that the filter statistics are displayed. This guidance applies to all enhanced mode firewall filters, filters with flexible conditions, and filters with the certain terminating actions. See those topics, listed under Related Documentation, for details.

Statement Hierarchy

Flexible match filter terms are available in three variations as shown in Table 47 on page 668. The **flexible-match** variation is configured at the **[edit firewall]** hierarchy level. It is used to define flexible match templates. The **flexible-filter-match-mask** and **flexible-match-range** are configured at the **[edit firewall family [inet|inet6|bridge|ethernet-switching|ccc|vpls] filter <*filter-name*> term <*term-name*> from]** hierarchy.

NOTE: On the EX9200 switches, you configure firewall filter flexible match conditions under **[edit firewall family ethernet-switching]** . For example: **flexible-filter-match-mask** and **flexible-match-range** are configured at the **[edit firewall family ethernet-switching filter <*filter-name*> term <*term-name*> from]** hierarchy.

Flexible Filter Match Types

Table 47: Flexible Filter Match Types

Flexible Filter Match Type	Available Attributes	Description
flexible-match	**<*name*>**	Create a flexible-match template named as the <*name*> attribute.
	bit-length	Length of the data to be matched in bits, not needed for string input (0..32)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	match-start	Start point to match in packet
flexible-match-mask	**bit-length**	Length of the data to be matched in bits, not needed for string input (0..128)
	bit-offset	Bit offset after the (match-start + byte) offset (0..7)
	byte-offset	Byte offset after the match start point
	flexible-mask-name	Select a flexible match from predefined template field. Required unless **match-start** is configured.
	mask-in-hex	Mask out bits in the packet data to be matched.
	match-start	Start point to match in packet. Required unless **flexible-mask-name** is configured.
	prefix	Value data/string to be matched.

Table 47: Flexible Filter Match Types *(continued)*

Flexible Filter Match Type	Available Attributes	Description
flexible-match-range	**bit-length**	Length of the data to be matched in bits. (0..32) Required unless **flexible-range-name** is configured.
	bit-offset	Bit offset after the (match-start + byte) offset. (0..7)
	byte-offset	Byte offset after the match start point
	flexible-range-name	Select a flexible match from predefined template.
	match-start	Start point to match in packet. Required unless **flexible-range-name** is configured.
	range	Range of values to be matched.
	range-except	Range of values to be not matched.

Flexible Filter Match Start Locations

Flexible match filter terms are constructed by giving a start location or anchor point within the packet. The start locations can be any of: layer-2, layer-3, layer-4 or payload, depending on the protocol family in use. Table 48 on page 669 shows available flexible filter match start locations by protocol family. You use these available start locations as the **match-start** locations for the flexible match filter terms.

From these start locations, specific byte and bit offsets can be utilized to allow the filter to match patterns at very specific locations within the packet.

Table 48: Flexible Filter Match Start Locations

Protocol Family	Available Start Locations
inet	**layer-3, layer-4 and payload**
inet6	**layer-3, layer-4 and payload**
bridge	**layer-2, layer-3, layer-4 and payload**
ccc	**layer-2, layer-3, layer-4 and payload**
mpls	**layer-3 and payload**
Support for MPLS added.**mpls**	**layer-3 and payload**
vpls	**layer-2, layer-3, layer-4 and payload**
ethernet-switching (EX9200 switches only)	**layer-2, layer-3, layer-4 and payload**

Flexible Filter Match Example

The following example illustrates the use and context for **flexible-match-mask**.

```
from {
    flexible-match-mask {
        flexible-mask-name <mask-name>;
        mask-in-hex <mask>;
        prefix <pattern>;
     }
}
```

The *<mask-name>* specifies for *flexible-mask-name* which predefined template is used for the flexible match condition. Templates can be defined to specify at which place (position) in the packet the flexible match condition should be executed.

The *<mask>* for *mask-in-hex* is in hexadecimal format. For example, a configured mask of **0xf0fc** specifies a match for the fist four bits in first byte (as referred by *<mask-name>*), and for the first six bits in the second byte. If the packet is IPv4 packet, and *<mask-name>* refers to first two bytes in L3 header, the search is for the IP version field and DSCP field. As another example, a configured mask **0xffc0** specifies a search for entire first byte and for two bits from the second byte. If the *<mask-name>* refers to first two bytes in L3 header, and the packet is IPv6 packet, this specifies the IP version field and DSCP in the Traffic Class field.

The *<pattern>* specified for *prefix* is an ASCII string. If first two characters are **0x**, then the string is processed as a hexadecimal number encoding appropriate bits. For example, the configured prefix **0x40c0** in combination with mask **0xf0fc** and *<mask-name>* referring first two bytes in L3 header, indicates a search for **0100** in the first four bits (version field is equal to 4) and **1100 00** in IPv4 DSCP field (DSCP is equal to cs6). Or, using the configured prefix **0x6c00** in combination with mask **0xffc0** and *<mask-name>* referring first two bytes in L3 header, specifies a search for for **0110** in the first four bits (version field is equal to 6), and **1100 00** in IPv6 DSCP field (DSCP is equal to cs6).

The first example defines a mask template that selects first two bytes (16 bits) from L3 header for flexible match:

```
firewall {
    flexible-match FM-FIRST-TWO-L3-BYTES {
        match-start layer-3;
        byte-offset 0;
        bit-offset 0;
        bit-length 16;
    }
}
```

The next example defines a mask template that selects the third through sixth byte (32 bits) of the packet payload for flexible match:

```
firewall {
    flexible-match FM-FOUR-PAYLOAD-BYTES {
        match-start payload;
        byte-offset 2;
        bit-offset 0;
        bit-length 32;
```



```
        }
    }
```

Following is an example filter demonstrating an ASCII character match, for the string *JNPR* (ASCII characters: **0x4a**, **0x4e**, **0x50**, **0x52**) in the third through sixth byte of the packet payload. The filter uses the **FM-FOUR-PAYLOAD-BYTES** mask template defined in the previous example.

```
firewall {
    family ccc filter FF-COUNT-JNPR-PACKETS {
        term JNPR-STRING {
            from {
                flexible-match-mask {
                    mask-in-hex 0xffffffff;
                    prefix JNPR;
                    flexible-mask-name FM-FOUR-PAYLOAD-BYTES;
                }
            }
            then {
                count CNT-JNPR-YES
                accept;
            }
        }
        term DEAFULT {
            then {
                count CNT-JNPR-NO
                accept;
            }
        }
    }
}
```

Following is a full example of a family ccc filter looking for DSCP equal to **cs6** and DSCP **ef**, regardless whether the encapsulated packets are IPv4 or IPv6. It uses the the **FM-FIRST-TWO-L3-BYTES** mask template defined in the first example.

```
firewall {
    family ccc filter FF-DSCP-CLASSIFY {
        term ROUTING-IPV4 {
            from {
                flexible-match-mask {
                    mask-in-hex 0xf0fc;
                    prefix 0x40c0;        # DSCP=cs6 in IPv4 header
                    flexible-mask-name FM-FIRST-TWO-L3-BYTES;
                }
            }
            then {
                count ROUTING-IPV4;
                accept;
            }
        }
        term ROUTING-IPV6 {
            from {
                flexible-match-mask {
                    mask-in-hex 0xffc0;
                    prefix 0x6c00;        # DSCP=cs6 in IPv6 header
                    flexible-mask-name FM-FIRST-TWO-L3-BYTES;
                }
            }
            then {
```

```
                    count ROUTING-IPV6;
                    accept;
                }
            }
            term VOICE-IPV4 {
                from {
                    flexible-match-mask {
                        mask-in-hex 0xf0fc;
                        prefix 0x40b8;            # DSCP=ef in IPv4 header
                        flexible-mask-name FM-FIRST-TWO-L3-BYTES;
                    }
                }
                then {
                    count VOICE-IPV4;
                    accept;
                }
            }
            term VOICE-IPV6 {
                from {
                    flexible-match-mask {
                        mask-in-hex 0xffc0;
                        prefix 0x6b80;             # DSCP=ef in IPv6 header
                        flexible-mask-name FM-FIRST-TWO-L3-BYTES;
                    }
                }
                then {
                    count VOICE-IPV6;
                    accept;
                }
            }
            term DEFAULT {
                then {
                    accept;
                }
            }
        }
    }
```

Release History Table

Release	Description
15.1	Support for MPLS added.

Related Documentation

- Firewall Filter Match Conditions for IPv4 Traffic on page 589
- Firewall Filter Match Conditions for IPv6 Traffic on page 629
- **enhanced-mode on page 1245**
- **enhanced-mode on page 1245**
- Firewall Filter Match Conditions for Layer 2 Bridging Traffic on page 658
- Firewall Filter Match Conditions for Layer 2 CCC Traffic on page 654
- Firewall Filter Match Conditions for VPLS Traffic on page 643

Firewall Filter Nonterminating Actions

Firewall filters support different sets of nonterminating actions for each protocol family.

NOTE: You cannot configure the **next term** action with a *terminating* action in the same filter term. However, you can configure the **next term** action with another *nonterminating* action in the same filter term.

Nonterminating actions carry with them an implicit accept action. In this context, *nonterminating* means that other actions can follow these actions whereas no other actions can follow a *terminating* action.

Table 49 on page 673 describes the nonterminating actions you can configure for a firewall filter term.

Table 49: Nonterminating Actions for Firewall Filters

Nonterminating Action	Description	Protocol Families
bgp-output-queue-priority priority *(expedited* **|** *(1-16))*	Assign the packet to one of the 17 prioritized BGP output queues.	• **family evpn** • **family inet** • **family inet-mdt** • **family inet-mvpn** • **family inet-vpn** • **family inet6** • **family inet6-mvpn** • **family inet6-vpn** • **family iso-vpn** • **family l2vpn** • **family route-target** • **family traffic-engineering**
count *counter-name*	Count the packet in the named counter.	• **family any** • **family bridge** • **family ccc** • **family inet** • **family inet6** • **family mpls** • **family vpls**
dont-fragment (set | clear)	Configure the value of the Don't Fragment bit (flag) in the IPv4 header to specify whether the datagram can be fragmented: • **set**—Change the flag value to one, preventing fragmentation. • **clear**—Change the flag value to zero, allowing fragmentation. NOTE: The **dont-fragment (set | clear)** actions are supported only on MPCs.	**family inet**

Table 49: Nonterminating Actions for Firewall Filters *(continued)*

Nonterminating Action	Description	Protocol Families
dscp *value*	Set the IPv4 Differentiated Services code point (DSCP) bit. You can specify a numerical value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix. The default DSCP value is best effort, that is, **be** or **0**. You can also specify one of the following text synonyms: • **af11**—Assured forwarding class 1, low drop precedence • **af12**—Assured forwarding class 1, medium drop precedence • **af13**—Assured forwarding class 1, high drop precedence • **af21**—Assured forwarding class 2, low drop precedence • **af22**—Assured forwarding class 2, medium drop precedence • **af23**—Assured forwarding class 2, high drop precedence • **af31**—Assured forwarding class 3, low drop precedence • **af32**—Assured forwarding class 3, medium drop precedence • **af33**—Assured forwarding class 3, high drop precedence • **af41**—Assured forwarding class 4, low drop precedence • **af42**—Assured forwarding class 4, medium drop precedence • **af43**—Assured forwarding class 4, high drop precedence • **be**—Best effort • **cs0**—Class selector 0 • **cs1**—Class selector 1 • **cs2**—Class selector 2 • **cs3**—Class selector 3 • **cs4**—Class selector 4 • **cs5**—Class selector 5 • **cs6**—Class selector 6 • **cs7**—Class selector 7 • **ef**—Expedited forwarding NOTE: This action is not supported on PTX Series Packet Transport Routers. NOTE: The actions **dscp 0** and **dscp be** are supported only on T320, T640, T1600, TX Matrix, TX Matrix Plus. and M320 routers and on the 10-Gigabit Ethernet Modular Port Concentrators (MPC), 60-Gigabit Ethernet MPC, 60-Gigabit Ethernet Queuing MPC, and 60-Gigabit Ethernet Enhanced Queuing MPC on MX Series routers. However, these actions are not supported on Enhanced III Flexible PIC Concentrators (FPCs) on M320 routers. NOTE: On T4000 routers, the **dscp 0** action is not supported during the interoperation between a T1600 Enhanced Scaling Type 4 FPC and a T4000 Type 5 FPC.	**family inet**

Table 49: Nonterminating Actions for Firewall Filters *(continued)*

Nonterminating Action	Description	Protocol Families
force-premium	By default, a hierarchical policer processes the traffic it receives according to the traffic's forwarding class. Premium, expedited-forwarding traffic, has priority for bandwidth over aggregate, best-effort traffic. The **force-premium** filter ensures that traffic matching the term is treated as premium traffic by a subsequent hierarchical policer, regardless of its forwarding class. This traffic is given preference over any aggregate traffic received by that policer. NOTE: The **force-premium** filter option is supported only on MPCs.	• **family any** • **family bridge** • **family ccc** • **family inet** • **family inet6** • **family VPLS**
forwarding-class ***class-name***	Classify the packet to the named forwarding class: • *forwarding-class-name* • **assured-forwarding** • **best-effort** • **expedited-forwarding** • **network-control**	• **family any** • **family bridge** • **family ccc** • **family inet** • **family inet6** • **family mpls** • **family vpls**
hierarchical-policer	Police the packet using the specified hierarchical policer	• **family any** • **family bridge** • **family ccc** • **family inet** • **family inet6** • **family mpls** • **family vpls**
ipsec-sa ***ipsec-sa***	Use the specified IPsec security association. NOTE: This action is not supported on MX Series routers, Type 5 FPCs on T4000 routers, and PTX Series Packet Transport Routers.	**family inet**
load-balance ***group-name***	Use the specified load-balancing group. NOTE: This action is not supported on MX Series routers or PTX Series Packet Transport Routers.	**family inet**
log	Log the packet header information in a buffer within the Packet Forwarding Engine. You can access this information by issuing the **show firewall log** command at the command-line interface (CLI). NOTE: The Layer 2 (L2) families log action is available only for MX Series routers with MPCs (MPC mode if the router has only MPCs, or mix mode if it has MPCs and DCPs). For MX Series routers with DPCs, the log action for L2 families is ignored if configured.	• **family bridge** • **family ccc** • **family inet** • **family inet6** • **family vpls**
logical-system ***logical-system-name***	Direct packets to a specific logical system.	• **family inet** • **family inet6**

Table 49: Nonterminating Actions for Firewall Filters *(continued)*

Nonterminating Action	Description	Protocol Families
loss-priority (high | medium-high | medium-low | low)	Set the packet loss priority (PLP) level. You cannot also configure the **three-color-policer** nonterminating action for the same firewall filter term. These two nonterminating actions are mutually exclusive. This action is supported on M120 and M320 routers; M7i and M10i routers with the Enhanced CFEB (CFEB-E); and MX Series routers. For IP traffic on M320, MX Series, and T Series routers with Enhanced II Flexible PIC Concentrators (FPCs), you must include the **tri-color** statement at the **[edit class-of-service]** hierarchy level to commit a PLP configuration with any of the four levels specified. If the **tri-color** statement is not enabled, you can only configure the **high** and **low** levels. This applies to all protocol families. For information about the **tri-color** statement and using behavior aggregate (BA) classifiers to set the PLP level of incoming packets, see *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*.	• **family any** • **family bridge** • **family ccc** • **family inet** • **family inet6** • **family mpls** • **family vpls**
next-hop-group ***group-name***	Use the specified next-hop group. We recommend that you do not use the **next-hop-group** action with the **port-mirror-instance** or **port-mirror** action in the same firewall filter.	• **family any** • **family inet**
next-interface ***interface-name***	(MX Series) Direct packets to the specified outgoing interface.	• **family inet** • **family inet6**
next-ip ***ip-address***	(MX Series) Direct packets to the specified destination IPv4 address.	**family inet**
next-ip6 ***ipv6-address***	(MX Series) Direct packets to the specified destination IPv6 address.	**family inet6**
packet-mode	Updates a bit field in the packet key buffer, which specifies traffic that will bypass flow-based forwarding. Packets with the **packet-mode** action modifier follow the packet-based forwarding path and bypass flow-based forwarding completely. Applies to SRX100, SRX210, SRX220, SRX240, and SRX650 devices only. For more information about selective stateless packet-based services, see the *Junos OS Security Configuration Guide*.	**family any**
policer ***policer-name***	Name of policer to use to rate-limit traffic.	• **family any** • **family bridge** • **family ccc** • **family inet** • **family inet6** • **family mpls** • **family vpls**

Table 49: Nonterminating Actions for Firewall Filters *(continued)*

Nonterminating Action	Description	Protocol Families
policy-map *policy-map-name*	(MX Series) Name of policy map used to assign specific rewrite rules to a specific customer.	• **family any** • **family ccc** • **family inet** • **family inet6** • **family mpls** • **family vpls**
port-mirror *instance-name*	Port-mirror the packet based on the specified family. This action is supported on M120 routers, M320 routers configured with Enhanced III FPCs, MX Series routers, and PTX Series Packet Transport Routers only. We recommend that you do not use both the **next-hop-group** and the **port-mirror** actions in the same firewall filter.	• **family any** • **family bridge** • **family ccc** • **family inet** • **family inet6** • **family vpls** • **family mpls**
port-mirror-instance *instance-name*	Port mirror a packet for an instance. This action is supported only on the MX series routers. We recommend that you do not use both the **next-hop-group** and the **port-mirror-instance** actions in the same firewall filter.	• **family any** • **family bridge** • **family ccc** • **family inet** • **family inet6** • **family vpls** • **family mpls**
prefix-action *action-name*	Count or police packets based on the specified action name. NOTE: This action is not supported on PTX Series Packet Transport Routers.	**family inet**
routing-instance *routing-instance-name*	Direct packets to the specified routing instance.	• **family inet** • **family inet6**
sample	Sample the packet. NOTE: Junos OS does not sample packets originating from the router. If you configure a filter and apply it to the output side of an interface, then only the transit packets going through that interface are sampled. Packets that are sent from the Routing Engine to the Packet Forwarding Engine are not sampled.	• **family inet** • **family inet6** • **family mpls**
service-accounting	Use the inline counting mechanism when capturing subscriber per-service statistics. Count the packet for service accounting. The count is applied to a specific named counter (**__junos-dyn-service-counter**) that RADIUS can obtain. The **service-accounting** and **service-accounting-deferred** keywords are mutually exclusive, both per-term and per-filter. NOTE: This action is not supported on T4000 Type 5 FPCs and PTX Series Packet Transport Routers.	• **family any** • **family inet** • **family inet6**

Table 49: Nonterminating Actions for Firewall Filters *(continued)*

Nonterminating Action	Description	Protocol Families
service-accounting-deferred	Use the deferred counting mechanism when capturing subscriber per-service statistics. The count is applied to a specific named counter (**__junos-dyn-service-counter**) that RADIUS can obtain. The **service-accounting** and **service-accounting-deferred** keywords are mutually exclusive, both per-term and per-filter. NOTE: This action is not supported on T4000 Type 5 FPCs and PTX Series Packet Transport Routers.	• **family any** • **family inet** • **family inet6**
service-filter-hit	(Only if the **service-filter-hit** flag is marked by a previous filter in the current type of chained filters) Direct the packet to the next type of filters. Indicate to subsequent filters in the chain that the packet was already processed. This action, coupled with the **service-filter-hit** match condition in receiving filters, helps to streamline filter processing. NOTE: This action is not supported on T4000 Type 5 FPCs and PTX Series Packet Transport Routers.	• **family any** • **family inet** • **family inet6**
syslog	Log the packet to the system log file. The syslog firewall action for existing **inet** and **inet6** families, and the **syslog** action in L2 family filters includes the following L2 information: Input interface, action, VLAN ID1, VLAN ID2, Ethernet type, source and destination MAC addresses, protocol, source and destination IP addresses, source and destination ports, and the number of packets. NOTE: The L2 families syslog action is available only for MX Series routers with MPCs (MPC mode if the router has only MPCs, or mix mode if it has MPCs and DCPs). For MX Series routers with DPCs, the syslog action for L2 families is ignored if configured.	• **family bridge** • **family ccc** • **family inet** • **family inet6** • **family vpls**
three-color-policer (single-rate | two-rate) ***policer-name***	Police the packet using the specified single-rate or two-rate three-color-policer. NOTE: You cannot also configure the **loss-priority** action for the same firewall filter term. These two actions are mutually exclusive.	• **family bridge** • **family ccc** • **family inet** • **family inet6** • **family mpls** • **family vpls**

Table 49: Nonterminating Actions for Firewall Filters *(continued)*

Nonterminating Action	Description	Protocol Families
traffic-class ***value***	Specify the traffic-class code point. You can specify a numerical value from **0** through **63**. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix. The default traffic-class value is best effort, that is, **be** or **0**. In place of the numeric value, you can specify one of the following text synonyms: • **af11**—Assured forwarding class 1, low drop precedence • **af12**—Assured forwarding class 1, medium drop precedence • **af13**—Assured forwarding class 1, high drop precedence • **af21**—Assured forwarding class 2, low drop precedence • **af22**—Assured forwarding class 2, medium drop precedence • **af23**—Assured forwarding class 2, high drop precedence • **af31**—Assured forwarding class 3, low drop precedence • **af32**—Assured forwarding class 3, medium drop precedence • **af33**—Assured forwarding class 3, high drop precedence • **af41**—Assured forwarding class 4, low drop precedence • **af42**—Assured forwarding class 4, medium drop precedence • **af43**—Assured forwarding class 4, high drop precedence • **be**—Best effort • **cs0**—Class selector 0 • **cs1**—Class selector 1 • **cs2**—Class selector 2 • **cs3**—Class selector 3 • **cs4**—Class selector 4 • **cs5**—Class selector 5 • **cs6**—Class selector 6 • **cs7**—Class selector 7 • **ef**—Expedited forwarding NOTE: The actions **traffic-class 0**and **traffic-class be** are supported only on T Series and M320 routers and on the 10-Gigabit Ethernet Modular Port Concentrator (MPC), 60-Gigabit Ethernet MPC, 60-Gigabit Ethernet Queuing MPC, and 60-Gigabit Ethernet Enhanced Queuing MPC on MX Series routers. However, these actions are not supported on Enhanced III Flexible PIC Concentrators (FPCs) on M320 routers.	**family inet6**

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Terminating Actions on page 680

Firewall Filter Terminating Actions

Firewall filters support a set of terminating actions for each protocol family. A filter-terminating action halts all evaluation of a firewall filter for a specific packet. The router performs the specified action, and no additional terms are examined.

NOTE: You cannot configure the **next term** action with a *terminating* action in the same filter term. However, you can configure the **next term** action with another *nonterminating* action in the same filter term.

For MX Series routers with MPCs, you need to initialize certain new firewall filters by walking the corresponding SNMP MIB, for example, **show snmp mib walk *name* ascii**. This forces Junos to learn the filter counters and ensure that the filter statistics are displayed. This guidance applies to all enhanced mode firewall filters, filters with flexible conditions, and filters with the certain terminating actions. See those topics, listed under Related Documentation, for details.

Table 50 on page 680 describes the terminating actions you can specify in a firewall filter term.

Table 50: Terminating Actions for Firewall Filters

Terminating Action	Description	Protocols
accept	Accept the packet.	• **family any** • **family inet** • **family inet6** • **family mpls** • **family vpls** • **family ccc** • **family bridge** • **family ethernet-switching** (for EX Series switches only)

Table 50: Terminating Actions for Firewall Filters *(continued)*

Terminating Action	Description	Protocols
decapsulate gre [routing-instance *instance-name*]	At a customer-facing interface on an MX Series router installed at the provider edge (PE) of an IPv4 transport network, enable de-encapsulation of generic routing encapsulation (GRE) packets transported through a filter-based GRE tunnel. You can configure a filter term that pairs this action with a match condition that includes a packet header match for the GRE protocol. For an IPv4 filter, include the **protocol gre** (or **protocol 47**) match condition. Attach the filter to the input of an Ethernet logical interface or aggregated Ethernet interface on a Modular Interface Card (MIC) or Modular Port Concentrator (MPC) in the router. If you commit a configuration that attaches a de-encapsulating filter to an interface that does not support filter-based GRE tunneling, the system writes a syslog warning message that the interface does not support the filter. When the interface receives a matched packet, processes that run on the Packet Forwarding Engine perform the following operations: • Remove the outer GRE header. • Forward the inner payload packet to its original destination by performing destination lookup. By default, the Packet Forwarding Engine uses the default routing instance to forward payload packets to the destination network. If the payload is MPLS, the Packet Forwarding Engine performs route lookup on the MPLS path routing table using the route label in the MPLS header. If you specify the **decapsulate** action with an optional routing instance name, the Packet Forwarding Engine performs route lookup on the routing instance, and the instance must be configured. NOTE: The **decapsulate** action that you configure at the **[edit firewall family inet filter *filter-name* term *term-name*]** hierarchy level does not process traffic with IPv4 and IPv6 options. As a result, traffic with such options is discarded by the de-encapsulation of GRE packets functionality. For more information, see "Understanding Filter-Based Tunneling Across IPv4 Networks" on page 875 and "Components of Filter-Based Tunneling Across IPv4 Networks" on page 883.	• **family inet**

Table 50: Terminating Actions for Firewall Filters *(continued)*

Terminating Action	Description	Protocols
decapsulate l2tp [routing-instance ***instance-name*** **] [forwarding-class** ***class-name*** **] [output-interface** ***interface-name*** **] [cookie** ***l2tpv3-cookie*** **] [sample]**	At a customer-facing interface on an MX Series router installed at the provider edge (PE) of an IPv4 transport network, enable de-encapsulation of Layer 2 tunneling protocol (L2TP) packets transported through a filter-based L2TP tunnel. You can configure a filter term that pairs this action with a match condition that includes a packet header match for the L2TP protocol. For IPv4 traffic, an input firewall filter **$junos-input-filter** and an output firewall filter **$junos-output-filter** are attached to the interface. Attach the filter to the input of an Ethernet logical interface or aggregated Ethernet interface on a Modular Interface Card (MIC) or Modular Port Concentrator (MPC) in the router. If you commit a configuration that attaches a de-encapsulating filter to an interface that does not support filter-based L2TP tunneling, the system writes a syslog warning message that the interface does not support the filter. The remote tunnel endpoint sends an IP tunnel packet that contains an Ethernet MAC address in the payload. If the destination MAC address of the payload packet contains the MAC address of the router, the Ethernet packet is sent in the outgoing direction towards the network, and it is processed and forwarded as though it is received on the customer port. If the source MAC address of the payload packet contains the MAC address of the router, the Ethernet packet is transmitted in the outgoing direction towards the customer port. If the tunnel does not contain the receive-cookie configured, packet injection does not happen. In such a case, any received tunnel packet is counted and dropped in the same manner in which packets that arrive with a wrong cookie are counted and dropped. The following parameters can be specified with the **decapsulate l2tp** action: • **routing-instance** ***instance-name***—By default, the Packet Forwarding Engine uses the default routing instance to forward payload packets to the destination network. If the payload is MPLS, the Packet Forwarding Engine performs route lookup on the MPLS path routing table using the route label in the MPLS header. If you specify the **decapsulate** action with an optional routing instance name, the Packet Forwarding Engine performs route lookup on the routing instance, and the instance must be configured. • **forwarding-class** ***class-name***—(Optional) Classify l2TP packets to the specified forwarding class. • **output-interface** ***interface-name***—(Optional) For L2TP tunnels, enable the packet to be duplicated and sent towards the customer or the network (based on the MAC address in the Ethernet payload). • **cookie** ***l2tpv3-cookie***—(Optional) For L2TP tunnels, specify the L2TP cookie for the duplicated packets. If the tunnel does not contain the receive-cookie configured, packet injection does not happen. In such a case, any received tunnel packet is counted and dropped in the same manner in which packets that arrive with a wrong cookie are counted and dropped. • **sample**—(Optional) Sample the packet. Junos OS does not sample packets originating from the router. If you configure a filter and apply it to the output side of an interface, then only the transit packets going through that interface are sampled. Packets that are sent from the Routing Engine to the Packet Forwarding Engine are not sampled. NOTE: The **decapsulate l2tp** action that you configure at the **[edit firewall family inet filter** ***filter-name*** **term** ***term-name*****]** hierarchy level does not process traffic with IPv4 and IPv6 options. As a result, traffic with such options is discarded by the de-encapsulation of L2TP packets functionality.	**family inet**

Table 50: Terminating Actions for Firewall Filters *(continued)*

Terminating Action	Description	Protocols
discard	Discard a packet silently, without sending an Internet Control Message Protocol (ICMP) message. Discarded packets are available for logging and sampling.	• **family any** • **family inet** • **family inet6** • **family mpls** • **family vpls** • **family ccc** • **family bridge** • **family ethernet-switching** (for EX Series switches only)
encapsulate ***template-name***	At a customer-facing interface on an MX Series router installed at the provider edge (PE) of an IPv4 transport network, enable filter-based generic routing encapsulation (GRE) tunneling using the specified tunnel template. You can configure a filter term that pairs this action with the appropriate match conditions, and then attach the filter to the input of an Ethernet logical interface or aggregated Ethernet interface on a Modular Interface Card (MIC) or Modular Port Concentrator (MPC) in the router. If you commit a configuration that attaches an encapsulating filter to an interface that does not support filter-based GRE tunneling, the system writes a syslog warning message that the interface does not support the filter. When the interface receives a matched packet, processes that run on the Packet Forwarding Engine use information in the specified tunnel template to perform the following operations: 1. Attach a GRE header (with or without a tunnel key value, as specified in the tunnel template. 2. Attach a header for the IPv4 transport protocol. 3. Forward the resulting GRE packet from the tunnel source interface to the tunnel destination (the remote PE router). The specified tunnel template must be configured using the tunnel-end-point statement under the **[edit firewall]** or **[edit logical-systems *logical-system-name* firewall]** hierarchy level. For more information, see "Understanding Filter-Based Tunneling Across IPv4 Networks" on page 875.	• **family inet** • **family inet6** • **family any** • **family mpls**

Table 50: Terminating Actions for Firewall Filters *(continued)*

Terminating Action	Description	Protocols
encapsulate ***template-name*** (for L2TP tunnels)	At a customer-facing interface on an MX Series router installed at the provider edge (PE) of an IPv4 transport network, enable filter-based L2TP tunneling using the specified tunnel template. You can configure a filter term that pairs this action with the appropriate match conditions, and then attach the filter to the input of an Ethernet logical interface or aggregated Ethernet interface on a Modular Interface Card (MIC) or Modular Port Concentrator (MPC) in the router. If you commit a configuration that attaches an encapsulating filter to an interface that does not support filter-based GRE tunneling, the system writes a syslog warning message that the interface does not support the filter. When the interface receives a matched packet, processes that run on the Packet Forwarding Engine use information in the specified tunnel template to perform the following operations: 1. Attach an L2TP header (with or without a tunnel key value, as specified in the tunnel template). 2. Attach a header for the IPv4 transport protocol. 3. Forward the resulting L2TP packet from the tunnel source interface to the tunnel destination (the remote PE router). The specified tunnel template must be configured using the **tunnel-end-point** statement under the **[edit firewall]** or **[edit logical-systems** ***logical-system-name*** **firewall]** statement hierarchy.	• **family inet**
exclude-accounting	Exclude the packet from being included in accurate accounting statistics for tunneled subscribers on an L2TP LAC. Typically used in filters that match DHCPv6 or ICMPv6 control traffic Failure to exclude these packets results in the idle-timeout detection mechanism considering these packets as data traffic, causing the timeout to never expire. (The idle timeout is configured with the **client-idle-timeout** and **client-idle-timeout-ingress-only** statements in the access profile session options.) The term excludes packets from being included in counts for both family accurate accounting and service accurate accounting. The packets are still included in the session interface statistics. The term is available for both **inet** and **inet6** families, but is used only for **inet6**.	• **family inet** • **family inet6**
logical-system ***logical-system-name***	Direct the packet to the specified logical system. NOTE: This action is not supported on PTX Series Packet Transport Routers.	• **family inet** • **family inet6**

Table 50: Terminating Actions for Firewall Filters *(continued)*

Terminating Action	Description	Protocols
reject ***message-type***	Reject the packet and return an ICMPv4 or ICMPv6 message: • If no ***message-type*** is specified, a **destination unreachable** message is returned by default. • If **tcp-reset** is specified as the ***message-type***, **tcp-reset** is returned only if the packet is a TCP packet. Otherwise, the **administratively-prohibited** message, which has a value of 13, is returned. • If any other ***message-type*** is specified, that message is returned. NOTE: Rejected packets can be sampled or logged if you configure the **sample** or **syslog** action. The ***message-type*** can be one of the following values: **address-unreachable**, **administratively-prohibited**, **bad-host-tos**, **bad-network-tos**, **beyond-scope**, **fragmentation-needed**, **host-prohibited**, **host-unknown**, **host-unreachable**, **network-prohibited**, **network-unknown**, **network-unreachable**, **no-route**, **port-unreachable**, **precedence-cutoff**, **precedence-violation**, **protocol-unreachable**, **source-host-isolated**, **source-route-failed**, or **tcp-reset**. NOTE: On PTX1000 routers, the reject action is supported on ingress interfaces only.	• **family inet** • **family inet6**
routing-instance ***instance-name***	Direct the packet to the specified routing instance.	• **family inet** • **family inet6**
topology ***topology-name***	Direct the packet to the specified topology. NOTE: This action is not supported on PTX Series Packet Transport Routers. Each routing instance (master or virtual-router) supports one default topology to which all forwarding classes are forwarded. For multitopology routing, you can configure a firewall filter on the ingress interface to match a specific forwarding class, such as expedited forwarding, with a specific topology. The traffic that matches the specified forwarding class is then added to the routing table for that topology.	• **family inet** • **family inet6**

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Firewall Filter Nonterminating Actions on page 673
- Firewall Filter Match Conditions for IPv4 Traffic on page 589
- Firewall Filter Match Conditions for IPv6 Traffic on page 629
- **enhanced-mode on page 1245**
- Firewall Filter Flexible Match Conditions on page 667

- *Firewall Filter Terminating and Nonterminating Actions for Protocol-Independent Traffic in Dynamic Service Profiles*

CHAPTER 16

Applying Firewall Filters to Routing Engine Traffic

Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List

This example shows how to configure a standard stateless firewall filter that limits certain TCP and Internet Control Message Protocol (ICMP) traffic destined for the Routing Engine by specifying a list of prefix sources that contain allowed BGP peers.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you create a stateless firewall filter that blocks all TCP connection attempts to port 179 from all requesters except BGP peers that have a specified prefix.

A source prefix list, **plist_bgp179**, is created that specifies the list of source prefixes that contain allowed BGP peers.

The stateless firewall filter **filter_bgp179** matches all packets from the source prefix list **plist_bgp179** to the destination port number 179.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set policy-options prefix-list plist_bgp179 apply-path "protocols bgp group <*> neighbor
  <*>"
set firewall family inet filter filter_bgp179 term 1 from source-address 0.0.0.0/0
set firewall family inet filter filter_bgp179 term 1 from source-prefix-list plist_bgp179 except
set firewall family inet filter filter_bgp179 term 1 from destination-port bgp
set firewall family inet filter filter_bgp179 term 1 then reject
set firewall family inet filter filter_bgp179 term 2 then accept
set interfaces lo0 unit 0 family inet filter input filter_bgp179
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
```

Configure the Filter

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the filter:

1. Expand the prefix list **bgp179** to include all prefixes pointed to by the BGP peer group defined by **protocols bgp group <*> neighbor <*>**.

   ```
   [edit policy-options prefix-list plist_bgp179]
   user@host# set apply-path " protocolsbgp group <*> neighbor <*>"
   ```

2. Define the filter term that rejects TCP connection attempts to port 179 from all requesters except the specified BGP peers.

   ```
   [edit firewall family inet filter filter_bgp179]
   user@host# set term term1 from source-address 0.0.0.0/0
   user@host# set term term1 from source-prefix-list bgp179 except
   user@host# set term term1 from destination-port bgp
   user@host# set term term1 then reject
   ```


3. Define the other filter term to accept all packets.

```
[edit firewall family inet filter filter_bgp179]
user@host# set term term2 then accept
```

4. Apply the firewall filter to the loopback interface.

```
[edit interfaces lo0 unit 0 family inet]
user@host# set filter input filter_bgp179
user@host# set address 127.0.0.1/32
```

Results

From configuration mode, confirm your configuration by entering the **show firewall**, **show interfaces**, and **show policy-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show firewall
family inet {
   filter filter_bgp179 {
      term 1 {
         from {
            source-address {
               0.0.0.0/0;
            }
            source-prefix-list {
               plist_bgp179 except;
            }
            destination-port bgp;
         }
         then {
            reject;
         }
      }
      term 2 {
         then {
            accept;
         }
      }
   }
}

user@host# show interfaces
lo0 {
   unit 0 {
      family inet {
         filter {
            input filter_bgp179;
         }
         address 127.0.0.1/32;
      }
   }
}
```

```
user@host# show policy-options
prefix-list plist_bgp179 {
   apply-path "protocols bgp group <*> neighbor <*>";
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Displaying the Firewall Filter Applied to the Loopback Interface

Purpose Verify that the firewall filter **filter_bgp179** is applied to the IPv4 input traffic at logical interface **lo0.0**.

Action Use the **show interfaces statistics operational mode** command for logical interface **lo0.0**, and include the **detail** option. Under the **Protocol inet** section of the command output section, the **Input Filters** field displays the name of the stateless firewall filter applied to the logical interface in the input direction.

```
[edit]
user@host> show interfaces statistics lo0.0 detail
  Logical interface lo0.0 (Index 321) (SNMP ifIndex 16) (Generation 130)
    Flags: SNMP-Traps Encapsulation: Unspecified
    Traffic statistics:
     Input  bytes  :                  0
     Output bytes  :                  0
     Input  packets:                  0
     Output packets:                  0
    Local statistics:
     Input  bytes  :                  0
     Output bytes  :                  0
     Input  packets:                  0
     Output packets:                  0
    Transit statistics:
     Input  bytes  :                  0                    0 bps
     Output bytes  :                  0                    0 bps
     Input  packets:                  0                    0 pps
     Output packets:                  0                    0 pps
    Protocol inet, MTU: Unlimited, Generation: 145, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Input Filters: filter_bgp179
      Addresses, Flags: Primary
        Destination: Unspecified, Local: 127.0.0.1, Broadcast: Unspecified,
Generation: 138
```

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Firewall Filter Match Conditions Based on Address Fields on page 606
- Example: Configuring a Stateless Firewall Filter to Protect Against TCP and ICMP Floods on page 719
- Example: Configuring a Filter to Accept Packets Based on IPv6 TCP Flags on page 704

• prefix-list on page 1233

Example: Configuring a Stateless Firewall Filter to Accept Traffic from Trusted Sources

This example shows how to create a stateless firewall filter that protects the Routing Engine from traffic originating from untrusted sources.

Requirements

No special configuration beyond device initialization is required before configuring stateless firewall filters.

Overview

In this example, you create a stateless firewall filter called protect-RE that discards all traffic destined for the Routing Engine except SSH and BGP protocol packets from specified trusted sources. This example includes the following firewall filter terms:

- **ssh-term**—Accepts TCP packets with a source address of **192.168.122.0/24** and a destination port that specifies SSH.
- **bgp-term**—Accepts TCP packets with a source address of **10.2.1.0/24** and a destination port that specifies BGP.
- **discard-rest-term**—For all packets that are not accepted by **ssh-term** or **bgp-term**, creates a firewall filter log and system logging records, then discards all packets.

NOTE: You can move terms within the firewall filter using the **insert** command. See *insert* in the *CLI User Guide*.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter protect-RE term ssh-term from source-address
  192.168.122.0/24
set firewall family inet filter protect-RE term ssh-term from protocol tcp
set firewall family inet filter protect-RE term ssh-term from destination-port ssh
set firewall family inet filter protect-RE term ssh-term then accept
set firewall family inet filter protect-RE term bgp-term from source-address 10.2.1.0/24
set firewall family inet filter protect-RE term bgp-term from protocol tcp
```

```
set firewall family inet filter protect-RE term bgp-term from destination-port bgp
set firewall family inet filter protect-RE term bgp-term then accept
set firewall family inet filter protect-RE term discard-rest-term then log
set firewall family inet filter protect-RE term discard-rest-term then syslog
set firewall family inet filter protect-RE term discard-rest-term then discard
set interfaces lo0 unit 0 family inet filter input protect-RE
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the stateless firewall filter:

1. Create the stateless firewall filter.

   ```
   [edit]
   user@host# edit firewall family inet filter protect-RE
   ```

2. Create the first filter term.

   ```
   [edit firewall family inet filter protect-RE]
   user@host# edit term ssh-term
   ```

3. Define the protocol, destination port, and source address match conditions for the term.

   ```
   [edit firewall family inet filter protect-RE term ssh-term]
   user@host# set from protocol tcp destination-port ssh source-address
     192.168.122.0/24
   ```

4. Define the actions for the term.

   ```
   [edit firewall family inet filter protect-RE term ssh-term]
   user@host# set then accept
   ```

5. Create the second filter term.

   ```
   [edit firewall family inet filter protect-RE]
   user@host# edit term bgp-term
   ```

6. Define the protocol, destination port, and source address match conditions for the term.

   ```
   [edit firewall family inet filter protect-RE term bgp-term]
   user@host# set from protocol tcp destination-port bgp source-address 10.2.1.0/24
   ```

7. Define the action for the term.

   ```
   [edit firewall family inet filter protect-RE term bgp-term]
   user@host# set then accept
   ```

8. Create the third filter term.

```
[edit firewall family inet filter protect-RE]
user@host# edit term discard-rest-term
```

9. Define the action for the term.

```
[edit firewall family inet filter protect-RE term discard-rest]
user@host# set then log syslog discard
```

10. Apply the filter to the input side of the Routing Engine interface.

```
[edit]
user@host# set interfaces lo0 unit 0 family inet filter input protect-RE
```

Results Confirm your configuration by entering the **show firewall** command and the **show interfaces lo0** command from configuration mode. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show firewall
family inet {
    filter protect-RE {
        term ssh-term {
            from {
                source-address {
                    192.168.122.0/24;
                }
                protocol tcp;
                destination-port ssh;
            }
            then accept;
        }
        term bgp-term {
            from {
                source-address {
                    10.2.1.0/24;
                }
                protocol tcp;
                destination-port bgp;
            }
            then accept;
        }
        term discard-rest-term {
            then {
                log;
                syslog;
                discard;
            }
        }
    }
}
```

```
user@host# show interfaces lo0
unit 0 {
```

```
    family inet {
        filter {
            input protect-RE;
        }
        address 127.0.0.1/32;
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, perform these tasks:

- Displaying Stateless Firewall Filter Configurations on page 694
- Verifying a Services, Protocols, and Trusted Sources Firewall Filter on page 694
- Displaying Stateless Firewall Filter Logs on page 695

Displaying Stateless Firewall Filter Configurations

Purpose Verify the configuration of the firewall filter.

Action From configuration mode, enter the **show firewall** command and the **show interfaces lo0** command.

Meaning Verify that the output shows the intended configuration of the firewall filter. In addition, verify that the terms are listed in the order in which you want the packets to be tested. You can move terms within a firewall filter by using the **insert** CLI command.

Verifying a Services, Protocols, and Trusted Sources Firewall Filter

Purpose Verify that the actions of the firewall filter terms are taken.

Action Send packets to the device that match the terms. In addition, verify that the filter actions are *not* taken for packets that do not match.

- Use the **ssh *host-name*** command from a host at an IP address that matches **192.168.122.0/24** to verify that you can log in to the device using only SSH from a host with this address prefix.
- Use the **show route summary** command to verify that the routing table on the device does not contain any entries with a protocol other than **Direct**, **Local**, **BGP**, or **Static**.

Sample Output

```
% ssh 192.168.249.71
```



```
%ssh host
user@host's password:
--- JUNOS 6.4-20040518.0 (JSERIES) #0: 2004-05-18 09:27:50 UTC

user@host>
```

```
user@host> show route summary
Router ID: 192.168.249.71

inet.0: 34 destinations, 34 routes (33 active, 0 holddown, 1 hidden)
              Direct:     10 routes,      9 active
               Local:      9 routes,      9 active
                 BGP:     10 routes,     10 active
              Static:      5 routes,      5 active
...
```

Meaning Verify the following information:

- You can successfully log in to the device using SSH.
- The **show route summary** command does not display a protocol other than **Direct**, **Local**, **BGP**, or **Static**.

Displaying Stateless Firewall Filter Logs

Purpose Verify that packets are being logged. If you included the **log** or **syslog** action in a term, verify that packets matching the term are recorded in the firewall log or your system logging facility.

Action From operational mode, enter the **show firewall log** command.

Sample Output

```
user@host> show firewall log
Log :
Time      Filter     Action Interface     Protocol Src Addr      Dest Addr
15:11:02  pfe        D      ge-0/0/0.0    TCP      172.17.28.19  192.168.70.71
15:11:01  pfe        D      ge-0/0/0.0    TCP      172.17.28.19  192.168.70.71
15:11:01  pfe        D      ge-0/0/0.0    TCP      172.17.28.19  192.168.70.71
15:11:01  pfe        D      ge-0/0/0.0    TCP      172.17.28.19  192.168.70.71
...
```

Meaning Each record of the output contains information about the logged packet. Verify the following information:

- Under **Time**, the time of day the packet was filtered is shown.
- The **Filter** output is always **pfe**.
- Under **Action**, the configured action of the term matches the action taken on the packet—**A** (accept), **D** (discard), **R** (reject).

- Under **Interface**, the inbound (ingress) interface on which the packet arrived is appropriate for the filter.
- Under **Protocol**, the protocol in the IP header of the packet is appropriate for the filter.
- Under **Src Addr**, the source address in the IP header of the packet is appropriate for the filter.
- Under **Dest Addr**, the destination address in the IP header of the packet is appropriate for the filter.

Related Documentation

- *show route summary*
- **show firewall on page 1591**
- **show firewall log on page 1600**
- *show interfaces (Loopback)*

Example: Configuring a Filter to Block Telnet and SSH Access

Requirements

You must have access to a remote host that has network connectivity with this device.

Overview

In this example, you create an IPv4 stateless firewall filter that logs and rejects Telnet or SSH access packets unless the packet is destined for or originates from the 192.168.1.0/24 subnet.

- To match packets destined for or originating from the **address 192.168.1.0/24** subnet, you use the **source-address 192.168.1.0/24** IPv4 match condition.
- To match packets destined for or originating from a TCP port, Telnet port, or SSH port, you use the **protocol tcp**, **port telnet**, and **telnet ssh** IPv4 match conditions.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configure the Stateless Firewall Filter on page 697
- Apply the Firewall Filter to the Loopback Interface on page 698
- Confirm and Commit Your Candidate Configuration on page 698

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

```
set firewall family inet filter local_acl term terminal_access from source-address
  192.168.1.0/24
set firewall family inet filter local_acl term terminal_access from protocol tcp
set firewall family inet filter local_acl term terminal_access from port ssh
set firewall family inet filter local_acl term terminal_access from port telnet
set firewall family inet filter local_acl term terminal_access then accept
set firewall family inet filter local_acl term terminal_access_denied from protocol tcp
set firewall family inet filter local_acl term terminal_access_denied from port ssh
set firewall family inet filter local_acl term terminal_access_denied from port telnet
set firewall family inet filter local_acl term terminal_access_denied then log
set firewall family inet filter local_acl term terminal_access_denied then reject
set firewall family inet filter local_acl term default-term then accept
set interfaces lo0 unit 0 family inet filter input local_acl
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter that selectively blocks Telnet and SSH access:

1. Create the stateless firewall filter **local_acl**.

```
[edit]
user@myhost# edit firewall family inet filter local_acl
```

2. Define the filter term **terminal_access**.

```
[edit firewall family inet filter local_acl]
user@myhost# set term terminal_access from source-address 192.168.1.0/24
user@myhost# set term terminal_access from protocol tcp
user@myhost# set term terminal_access from port ssh
user@myhost# set term terminal_access from port telnet
user@myhost# set term terminal_access then accept
```

3. Define the filter term **terminal_access_denied**.

```
[edit firewall family inet filter local_acl]
user@myhost# set term terminal_access_denied from protocol tcp
user@myhost# set term terminal_access_denied from port ssh
user@myhost# set term terminal_access_denied from port telnet
user@myhost# set term terminal_access_denied then log
user@myhost# set term terminal_access_denied then reject
user@myhost# set term default-term then accept
```


Apply the Firewall Filter to the Loopback Interface

Step-by-Step Procedure

- To apply the firewall filter to the loopback interface:

```
[edit]
user@myhost# set interfaces lo0 unit 0 family inet filter input local_acl
user@myhost# set interfaces lo0 unit 0 family inet address 127.0.0.1/32
```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@myhost# show firewall
family inet {
  filter local_acl {
    term terminal_access {
      from {
        source-address {
          192.168.1.0/24;
        }
        protocol tcp;
        port [ssh telnet];
      }
      then accept;
    }
    term terminal_access_denied {
      from {
        protocol tcp;
        port [ssh telnet];
      }
      then {
        log;
        reject;
      }
    }
    term default-term {
      then accept;
    }
  }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@myhost# show interfaces
lo0 {
```

```
        unit 0 {
            family inet {
                filter {
                    input local_acl;
                }
                source-address 127.0.0.1/32;
            }
        }
    }
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@myhost# commit
```

Verification

Confirm that the configuration is working properly.

- Verifying Accepted Packets on page 699
- Verifying Logged and Rejected Packets on page 700

Verifying Accepted Packets

Purpose Verify that the actions of the firewall filter terms are taken.

Action

1. Clear the firewall log on your router or switch.

```
user@myhost> clear firewall log
```

2. From a host at an IP address *within* the 192.168.1.0/24 subnet, use the **ssh** ***hostname*** command to verify that you can log in to the device using only SSH. This packet should be accepted, and the packet header information for this packet should not be logged in the firewall filter log buffer in the Packet Forwarding Engine.

```
user@host-A> ssh myhost
user@myhosts's password:
--- JUNOS 11.1-20101102.0 built 2010-11-02 04:48:46 UTC

% cli
user@myhost>
```

3. From a host at an IP address *within* the 192.168.1.0/24 subnet, use the **telnet** ***hostname*** command to verify that you can log in to your router or switch using only Telnet. This packet should be accepted, and the packet header information for this packet should not be logged in the firewall filter log buffer in the Packet Forwarding Engine.

```
user@host-A> telnet myhost
Trying 192.168.249.71...
Connected to myhost-fxp0.example.net.
Escape character is '^]'.
```



```
host (ttyp0)

login: user
Password:

--- JUNOS 11.1-20101102.0 built 2010-11-02 04:48:46 UTC

% cli
user@myhost>
```

4. Use the **show firewall log** command to verify that the routing table on the device does not contain any entries with a source address in the 192.168.1.0/24 subnet.

```
user@myhost> show firewall log
```

Verifying Logged and Rejected Packets

Purpose Verify that the actions of the firewall filter terms are taken.

Action

1. Clear the firewall log on your router or switch.

   ```
   user@myhost> clear firewall log
   ```

2. From a host at an IP address *outside of* the 192.168.1.0/24 subnet, use the **ssh *hostname*** command to verify that you cannot log in to the device using only SSH. This packet should be rejected, and the packet header information for this packet should be logged in the firewall filter log buffer in the Packet Forwarding Engine.

   ```
   user@host-B ssh myhost
   ssh: connect to host sugar port 22: Connection refused
   --- JUNOS 11.1-20101102.0 built 2010-11-02 04:48:46 UTC
   %
   ```

3. From a host at an IP address *outside of* the 192.168.1.0/24 subnet, use the **telnet *hostname*** command to verify that you can log in to the device using only Telnet. This packet should be rejected, and the packet header information for this packet should be logged in the firewall filter log buffer in the PFE.

   ```
   user@host-B> telnet myhost
   Trying 192.168.249.71...
   telnet: connect to address 192.168.187.3: Connection refused
   telnet: Unable to connect to remote host
   %
   ```

4. Use the **show firewall log** command to verify that the routing table on the device does not contain any entries with a source address in the 192.168.1.0/24 subnet.

   ```
   user@myhost> show firewall log

   Time     Filter    Action Interface Protocol Src Addr       Dest Addr
   18:41:25 local_acl R      fxp0.0    TCP      192.168.187.5  192.168.187.1
   18:41:25 local_acl R      fxp0.0    TCP      192.168.187.5  192.168.187.1
   18:41:25 local_acl R      fxp0.0    TCP      192.168.187.5  192.168.187.1
   ...
   18:43:06 local_acl R      fxp0.0    TCP      192.168.187.5  192.168.187.1
   18:43:06 local_acl R      fxp0.0    TCP      192.168.187.5  192.168.187.1
   18:43:06 local_acl R      fxp0.0    TCP      192.168.187.5  192.168.187.1
   ...
   ```

Example: Configuring a Filter to Block TFTP Access

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

By default, to decrease vulnerability to denial-of-service (DoS) attacks, the Junos OS filters and discards Dynamic Host Configuration Protocol (DHCP) or Bootstrap Protocol (BOOTP) packets that have a source address of 0.0.0.0 and a destination address of 255.255.255.255. This default filter is known as a unicast RPF check. However, some vendors' equipment automatically accepts these packets.

To interoperate with other vendors' equipment, you can configure a filter that checks for both of these addresses and overrides the default RPF-check filter by accepting these packets. In this example, you block Trivial File Transfer Protocol (TFTP) access, logging any attempts to establish TFTP connections.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configure the Stateless Firewall Filter on page 702
- Apply the Firewall Filter to the Loopback Interface on page 703
- Confirm and Commit Your Candidate Configuration on page 703

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter tftp_access_control term one from protocol udp
set firewall family inet filter tftp_access_control term one from port tftp
set firewall family inet filter tftp_access_control term one then log
set firewall family inet filter tftp_access_control term one then discard
set interfaces lo0 unit 0 family inet filter input tftp_access_control
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter that selectively blocks TFTP access:

1. Create the stateless firewall filter **tftp_access_control**.

```
[edit]
user@host# edit firewall family inet filter tftp_access_control
```

2. Specify a match on packets received on UDP port 69.

```
[edit firewall family inet filter tftp_access_control]
user@host# set term one from protocol udp
user@host# set term one from port tftp
```

3. Specify that matched packets be logged to the buffer on the Packet Forwarding Engine and then discarded.

```
[edit firewall family inet filter tftp_access_control]
user@host# set term one then log
user@host# set term one then discard
```

Apply the Firewall Filter to the Loopback Interface

Step-by-Step Procedure

To apply the firewall filter to the loopback interface:

- ```
  [edit]
  user@host# set interfaces lo0 unit 0 family inet filter input tftp_access_control
  user@host# set interfaces lo0 unit 0 family inet address 127.0.0.1/32
  ```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter tftp_access_control {
    term one {
      from {
        protocol udp;
        port tftp;
      }
      then {
        log;
        discard;
      }
    }
  }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
lo0 {
  unit 0 {
    family inet {
      filter {
        input tftp_access_control;
      }
```
```

```
            address 127.0.0.1/32;
          }
        }
      }
```

3. If you are done configuring the device, commit your candidate configuration.

   ```
   [edit]
   user@host# commit
   ```

Verification

Confirm that the configuration is operating properly:

Verifying Logged and Discarded Packets

Purpose Verify that the actions of the firewall filter terms are taken.

Action To

1. Clear the firewall log on your router or switch.

   user@myhost> **clear firewall log**

2. From another host, send a packet to UDP port **69** on this router or switch.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Stateless Firewall Filter to Accept Traffic from Trusted Sources on page 691
- Example: Configuring a Filter to Block Telnet and SSH Access on page 696
- Example: Configuring a Filter to Accept OSPF Packets from a Prefix on page 773
- Example: Configuring a Filter to Accept DHCP Packets Based on Address on page 771

Example: Configuring a Filter to Accept Packets Based on IPv6 TCP Flags

This example shows how to configure a standard stateless firewall filter to accept packets from a trusted source.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you create a filter that accepts packets with specific IPv6 TCP flags.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

CLI Quick Configuration

To quickly configure this example, copy the following commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet6 filter tcp_filter term 1 from next-header tcp
set firewall family inet6 filter tcp_filter term 1 from tcp-flags syn
set firewall family inet6 filter tcp_filter term 1 then count tcp_syn_pkt
set firewall family inet6 filter tcp_filter term 1 then log
set firewall family inet6 filter tcp_filter term 1 then accept
set interfaces lo0 unit 0 family inet6 filter input tcp_filter
set interfaces lo0 unit 0 family inet6 address ::10.34.1.0/120
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the firewall filter

1. Create the IPv6 stateless firewall filter **tcp_filter**.

   ```
   [edit]
   user@host# edit firewall family inet6 filter tcp_filter
   ```

2. Specify that a packet matches if it is the initial packet in a TCP session and the next header after the IPv6 header is type TCP.

   ```
   [edit firewall family inet6 filter tcp_filter]
   user@host# set term 1 from next-header tcp
   user@host# set term 1 from tcp-flags syn
   ```

3. Specify that matched packets are counted, logged to the buffer on the Packet Forwarding Engine, and accepted.

   ```
   [edit firewall family inet6 filter tcp_filter]
   user@host# set term 1 then count tcp_syn_pkt
   user@host# set term 1 then log
   user@host# set term 1 then accept
   ```


Apply the Firewall Filter to the Loopback Interface

Step-by-Step Procedure

To apply the firewall filter to the loopback interface:

- ```
  [edit]
  user@host# set interfaces lo0 unit 0 family inet6 filter input tcp_filter
  user@host# set interfaces lo0 unit 0 family inet6 address ::10.34.1.0/120
  ```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

   ```
   [edit]
   user@host# show firewall
   family inet6 {
     filter tcp_filter {
       term 1 {
         from {
           next-header tcp;
           tcp-flags syn;
         }
         then {
           count tcp_syn_pkt;
           log;
           accept;
         }
       }
     }
   }
   ```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

   ```
   [edit]
   user@host# show interfaces
   lo0 {
     unit 0 {
       family inet6 {
         filter {
           input tcp_filter;
         }
         address ::10.34.1.0/120;
       }
     }
   }
   ```


   ```

3. When you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the **show firewall** operational mode command.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Stateless Firewall Filter to Protect Against TCP and ICMP Floods on page 719
- Example: Configuring a Filter to Block TCP Access to a Port Except from Specified BGP Peers on page 713

Example: Filtering Packets Received on an Interface Set

This example shows how to configure a standard stateless firewall filter to match packets tagged for a particular interface set.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you apply a stateless firewall filter to the input of the router or switch loopback interface. The firewall filter includes a term that matches packets tagged for a particular interface set.

Topology

You create the firewall filter **L2_filter** to apply rate limits to the protocol-independent traffic received on the following interfaces:

- **fe-0/0/0.0**
- **fe-1/0/0.0**
- **fe-1/1/0.0**

NOTE: The interface type in this topic is just an example. The fe- interface type is not supported by EX Series switches.

First, for protocol-independent traffic received on **fe-0/0/0.0**, the firewall filter term **t1** applies policer **p1**.

For protocol-independent traffic received on any other Fast Ethernet interfaces, firewall filter term **t2** applies policer **p2**. To define an interface set that consists of all Fast Ethernet interfaces, you include the **interface-set *interface-set-name interface-name*** statement at the **[edit firewall]** hierarchy level. To define a packet-matching criteria based on the interface on which a packet arrives to a specified interface set, you configure a term that uses the **interface-set** firewall filter match condition.

Finally, for any other protocol-independent traffic, firewall filter term **t3** applies policer **p3**.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces fe-0/0/0 unit 0 family inet address 10.1.1.1/30
set interfaces fe-1/0/0 unit 0 family inet address 10.2.2.1/30
set interfaces fe-1/1/0 unit 0 family inet address 10.4.4.1/30
set firewall policer p1 if-exceeding bandwidth-limit 5m
set firewall policer p1 if-exceeding burst-size-limit 10m
set firewall policer p1 then discard
set firewall policer p2 if-exceeding bandwidth-limit 40m
set firewall policer p2 if-exceeding burst-size-limit 100m
set firewall policer p2 then discard
set firewall policer p3 if-exceeding bandwidth-limit 600m
set firewall policer p3 if-exceeding burst-size-limit 1g
set firewall policer p3 then discard
set firewall interface-set ifset fe-*
set firewall family any filter L2_filter term t1 from interface fe-0/0/0.0
set firewall family any filter L2_filter term t1 then count c1
set firewall family any filter L2_filter term t1 then policer p1
set firewall family any filter L2_filter term t2 from interface-set ifset
```

```
set firewall family any filter L2_filter term t2 then count c2
set firewall family any filter L2_filter term t2 then policer p2
set firewall family any filter L2_filter term t3 then count c3
set firewall family any filter L2_filter term t3 then policer p3
set interfaces lo0 unit 0 family inet address 172.16.1.157/30
set interfaces lo0 unit 0 family inet address 172.16.1.157/30
set interfaces lo0 unit 0 filter input L2_filter
```

Configuring the Interfaces for Which the Stateless Firewall Filter Terms Take Rate-Limiting Actions

Step-by-Step Procedure

To configure the interfaces for which the stateless firewall filter terms take rate-limiting actions:

1. Configure the logical interface whose input traffic will be matched by the first term of the firewall filter.

```
[edit]
user@host# set interfaces fe-0/0/0 unit 0 family inet address 10.1.1.1/30
```

2. Configure the logical interfaces whose input traffic will be matched by the second term of the firewall filter.

```
[edit ]
user@host# set interfaces fe-1/0/0 unit 0 family inet address 10.2.2.1/30
user@host# set interfaces fe-1/1/0 unit 0 family inet address 10.4.4.1/30
```

3. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Results

Confirm the configuration of the router (or switch) transit interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
fe-0/0/0 {
  unit 0 {
    family inet {
      address 10.1.1.1/30;
    }
  }
}
fe-1/0/0 {
  unit 0 {
    family inet {
      address 10.2.2.1/30;
    }
  }
}
```

```
fe-1/1/0 {
  unit 0 {
    family inet {
      address 10.4.4.1/30;
    }
  }
}
```

Configuring the Stateless Firewall Filter That Rate-Limits Protocol-Independent Traffic Based on the Interfaces on Which Packets Arrive

Step-by-Step Procedure

To configure the standard stateless firewall **L2_filter** that uses policers (**p1**, **p2**, and **p3**) to rate-limit protocol-independent traffic based on the interfaces on which the packets arrive:

1. Configure the firewall statements.

   ```
   [edit]
   user@host# edit firewall
   ```

2. Configure the policer **p1** to discard traffic that exceeds a traffic rate of **5m** bps or a burst size of **10m** bytes.

   ```
   [edit firewall]
   user@host# set policer p1 if-exceeding bandwidth-limit 5m
   user@host# set policer p1 if-exceeding burst-size-limit 10m
   user@host# set policer p1 then discard
   ```

3. Configure the policer **p2** to discard traffic that exceeds a traffic rate of **40m** bps or a burst size of **100m** bytes .

   ```
   [edit firewall]
   user@host# set policer p2 if-exceeding bandwidth-limit 40m
   user@host# set policer p2 if-exceeding burst-size-limit 100m
   user@host# set policer p2 then discard
   ```

4. Configure the policer **p3** to discard traffic that exceeds a traffic rate of **600m** bps or a burst size of **1g** bytes.

   ```
   [edit firewall]
   user@host# set policer p3 if-exceeding bandwidth-limit 600m
   user@host# set policer p3 if-exceeding burst-size-limit 1g
   user@host# set policer p3 then discard
   ```

5. Define the interface set **ifset** to be the group of all Fast Ethernet interfaces on the router.

   ```
   [edit firewall]
   user@host# set interface-set ifset fe-*
   ```

6. Create the stateless firewall filter **L2_filter**.

   ```
   [edit firewall]
   ```



```
user@host# edit family any filter L2_filter
```

7. Configure filter term **t1** to match IPv4, IPv6, or MPLS packets received on interface **fe-0/0/0.0** and use policer **p1** to rate-limit that traffic.

```
[edit firewall family any filter L2_filter]
user@host# set term t1 from interface fe-0/0/0.0
user@host# set term t1 then count c1
user@host# set term t1 then policer p1
```

8. Configure filter term **t2** to match packets received on interface-set **ifset** and use policer **p2** to rate-limit that traffic.

```
[edit firewall family any filter L2_filter]
user@host# set term t2 from interface-set ifset
user@host# set term t2 then count c2
user@host# set term t2 then policer p2
```

9. Configure filter term **t3** to use policer **p3** to rate-limit all other traffic.

```
[edit firewall family any filter L2_filter]
user@host# set term t3 then count c3
user@host# set term t3 then policer p3
```

10. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```

Results Confirm the configuration of the stateless firewall filter and the policers referenced as firewall filter actions by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family any {
  filter L2_filter {
    term t1 {
      from {
        interface fe-0/0/0.0;
      }
      then {
        policer p1;
        count c1;
      }
    }
    term t2 {
      from {
        interface-set ifset;
      }
      then {
```

```
                policer p2;
                count c2;
            }
        }
        term t3 {
            then {
                policer p3;
                count c3;
            }
        }
    }
}
policer p1 {
    if-exceeding {
        bandwidth-limit 5m;
        burst-size-limit 10m;
    }
    then discard;
}
policer p2 {
    if-exceeding {
        bandwidth-limit 40m;
        burst-size-limit 100m;
    }
    then discard;
}
policer p3 {
    if-exceeding {
        bandwidth-limit 600m;
        burst-size-limit 1g;
    }
    then discard;
}
interface-set ifset {
    fe-*;
}
```

Applying the Stateless Firewall Filter to the Routing Engine Input Interface

Step-by-Step Procedure

To apply the stateless firewall filter to the Routing Engine input interface:

1. Apply the stateless firewall filter to the Routing Engine interface in the input direction.

```
[edit]
user@host# set interfaces lo0 unit 0 family inet address 172.16.1.157/30
user@host# set interfaces lo0 unit 0 filter input L2_filter
```

2. If you are done configuring the device, commit the configuration.

```
[edit]
user@host# commit
```


Results Confirm the application of the firewall filter to the Routing Engine input interface by entering the **show interfaces** command again. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
user@host# show interfaces
fe-0/0/0 {
   ...
}
fe-1/0/0 {
   ...
}
fe-1/1/0 {
   ...
}
lo0 {
   unit 0 {
      filter {
         input L2_filter;
      }
      family inet {
         address 172.16.1.157/30;
      }
   }
}
```

Verification

To confirm that the configuration is working properly, use the show firewall **filter L2_filter** operational mode command to monitor traffic statistics about the firewall filter and three counters.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Filtering Packets Received on an Interface Set Overview on page 862

Example: Configuring a Filter to Block TCP Access to a Port Except from Specified BGP Peers

This example shows how to configure a standard stateless firewall filter that blocks all TCP connection attempts to port 179 from all requesters except from specified BGP peers.

Requirements

No special configuration beyond device initialization is required before you configure this example.

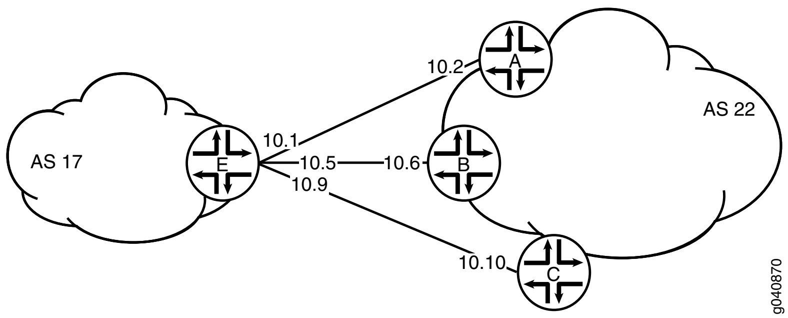
Overview

In this example, you create a stateless firewall filter that blocks all TCP connection attempts to port 179 from all requesters except the specified BGP peers.

The stateless firewall filter **filter_bgp179** matches all packets from the directly connected interfaces on Device A and Device B to the destination port number 179.

Figure 48 on page 714 shows the topology used in this example. Device C attempts to make a TCP connection to Device E. Device E blocks the connection attempt. This example shows the configuration on Device E.

Figure 48: Typical Network with BGP Peer Sessions



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device C

```
set interfaces ge-1/2/0 unit 10 description to-E
set interfaces ge-1/2/0 unit 10 family inet address 10.10.10.10/30
set protocols bgp group external-peers type external
set protocols bgp group external-peers peer-as 17
set protocols bgp group external-peers neighbor 10.10.10.9
set routing-options autonomous-system 22
```

Device E

```
set interfaces ge-1/2/0 unit 0 description to-A
set interfaces ge-1/2/0 unit 0 family inet address 10.10.10.1/30
set interfaces ge-1/2/1 unit 5 description to-B
set interfaces ge-1/2/1 unit 5 family inet address 10.10.10.5/30
set interfaces ge-1/0/0 unit 9 description to-C
set interfaces ge-1/0/0 unit 9 family inet address 10.10.10.9/30
set interfaces lo0 unit 2 family inet filter input filter_bgp179
set interfaces lo0 unit 2 family inet address 192.168.0.1/32
set protocols bgp group external-peers type external
set protocols bgp group external-peers peer-as 22
set protocols bgp group external-peers neighbor 10.10.10.2
set protocols bgp group external-peers neighbor 10.10.10.6
set protocols bgp group external-peers neighbor 10.10.10.10
set routing-options autonomous-system 17
```



```
set firewall family inet filter filter_bgp179 term 1 from source-address 10.10.10.2/32
set firewall family inet filter filter_bgp179 term 1 from source-address 10.10.10.6/32
set firewall family inet filter filter_bgp179 term 1 from destination-port bgp
set firewall family inet filter filter_bgp179 term 1 then accept
set firewall family inet filter filter_bgp179 term 2 then reject
```

Configuring Device E

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device E with a stateless firewall filter that blocks all TCP connection attempts to port 179 from all requestors except specified BGP peers:

1. Configure the interfaces.

```
user@E# set interfaces ge-1/2/0 unit 0 description to-A
user@E# set interfaces ge-1/2/0 unit 0 family inet address 10.10.10.1/30

user@E# set interfaces ge-1/2/1 unit 5 description to-B
user@E# set interfaces ge-1/2/1 unit 5 family inet address 10.10.10.5/30

user@E# set interfaces ge-1/0/0 unit 9 description to-C
user@E# set interfaces ge-1/0/0 unit 9 family inet address 10.10.10.9/30
```

2. Configure BGP.

```
[edit protocols bgp group external-peers]
user@E# set type external
user@E# set peer-as 22
user@E# set neighbor 10.10.10.2
user@E# set neighbor 10.10.10.6
user@E# set neighbor 10.10.10.10
```

3. Configure the autonomous system number.

```
[edit routing-options]
user@E# set autonomous-system 17
```

4. Define the filter term that accepts TCP connection attempts to port 179 from the specified BGP peers.

```
[edit firewall family inet filter filter_bgp179]
user@E# set term 1 from source-address 10.10.10.2/32
user@E# set term 1 from source-address 10.10.10.6/32
user@E# set term 1 from destination-port bgp
user@E# set term 1 then accept
```

5. Define the other filter term to reject packets from other sources.

```
[edit firewall family inet filter filter_bgp179]
user@E# set term 2 then reject
```

6. Apply the firewall filter to the loopback interface.

```
[edit interfaces lo0 unit 2 family inet]
user@E# set filter input filter_bgp179
user@E# set address 192.168.0.1/32
```

Results From configuration mode, confirm your configuration by entering the **show firewall**, **show interfaces**, **show protocols**, and **show routing-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@E# show firewall
family inet {
  filter filter_bgp179 {
    term 1 {
      from {
        source-address {
          10.10.10.2/32;
          10.10.10.6/32;
        }
        destination-port bgp;
      }
      then accept;
    }
    term 2 {
      then {
        reject;
      }
    }
  }
}
```

```
user@E# show interfaces
lo0 {
  unit 2 {
    family inet {
      filter {
        input filter_bgp179;
      }
      address 192.168.0.1/32;
    }
  }
}
ge-1/2/0 {
  unit 0 {
    description to-A;
    family inet {
      address 10.10.10.1/30;
    }
  }
}
ge-1/2/1 {
  unit 5 {
    description to-B;
```



```
            family inet {
                address 10.10.10.5/30;
            }
        }
    }
    ge-1/0/0 {
        unit 9 {
            description to-C;
            family inet {
                address 10.10.10.9/30;
            }
        }
    }
```

```
user@E# show protocols
bgp {
    group external-peers {
        type external;
        peer-as 22;
        neighbor 10.10.10.2;
        neighbor 10.10.10.6;
        neighbor 10.10.10.10;
    }
}
```

```
user@E# show routing-options
autonomous-system 17;
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Verifying That the Filter Is Configured

Purpose Make sure that the filter is listed in output of the **show firewall filter** command.

Action

```
user@E> show firewall filter filter_bgp179
Filter: filter_bgp179
```

Verifying the TCP Connections

Purpose Verify the TCP connections.

Action From operational mode, run the **show system connections extensive** command on Device C and Device E.

The output on Device C shows the attempt to establish a TCP connection. The output on Device E shows that connections are established with Device A and Device B only.

```
user@C> show system connections extensive | match 10.10.10

tcp4       0      0  10.10.10.9.51872     10.10.10.10.179       SYN_SENT
```

```
user@E> show system connections extensive | match 10.10.10

tcp4       0      0  10.10.10.5.179        10.10.10.6.62096     ESTABLISHED
tcp4       0      0  10.10.10.6.62096      10.10.10.5.179       ESTABLISHED
tcp4       0      0  10.10.10.1.179        10.10.10.2.61506     ESTABLISHED
tcp4       0      0  10.10.10.2.61506      10.10.10.1.179       ESTABLISHED
```

Monitoring Traffic on the Interfaces

Purpose Use the **monitor traffic** command to compare the traffic on an interface that establishes a TCP connection with the traffic on an interface that does not establish a TCP connection.

Action From operational mode, run the **monitor traffic** command on the Device E interface to Device B and on the Device E interface to Device C. The following sample output verifies that in the first example, acknowledgment (**ack**) messages are received. In the second example, **ack** messages are not received.

```
user@E> monitor traffic size 1500 interface ge-1/2/1.5
19:02:49.700912 Out IP 10.10.10.5.bgp > 10.10.10.6.62096: P
3330573561:3330573580(19) ack 915601686 win 16384 <nop,nop,timestamp 1869518816
1869504850>: BGP, length: 19
19:02:49.801244  In IP 10.10.10.6.62096 > 10.10.10.5.bgp: . ack 19 win 16384
<nop,nop,timestamp 1869518916 1869518816>
19:03:03.323018  In IP 10.10.10.6.62096 > 10.10.10.5.bgp: P 1:20(19) ack 19 win
16384 <nop,nop,timestamp 1869532439 1869518816>: BGP, length: 19
19:03:03.422418 Out IP 10.10.10.5.bgp > 10.10.10.6.62096: . ack 20 win 16384
<nop,nop,timestamp 1869532539 1869532439>
19:03:17.220162 Out IP 10.10.10.5.bgp > 10.10.10.6.62096: P 19:38(19) ack 20 win
 16384 <nop,nop,timestamp 1869546338 1869532439>: BGP, length: 19
19:03:17.320501  In IP 10.10.10.6.62096 > 10.10.10.5.bgp: . ack 38 win 16384
<nop,nop,timestamp 1869546438 1869546338>
```

```
user@E> monitor traffic size 1500 interface ge-1/0/0.9

18:54:20.175471 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0)
 win 16384 <mss 1460,nop,wscale 0,nop,nop,timestamp 1869009240 0,sackOK,eol>
18:54:23.174422 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0)
 win 16384 <mss 1460,nop,wscale 0,nop,nop,timestamp 1869012240 0,sackOK,eol>
18:54:26.374118 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0)
 win 16384 <mss 1460,nop,wscale 0,nop,nop,timestamp 1869015440 0,sackOK,eol>
18:54:29.573799 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0)
 win 16384 <mss 1460,sackOK,eol>
18:54:32.773493 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0)
 win 16384 <mss 1460,sackOK,eol>
18:54:35.973185 Out IP 10.10.10.9.61335 > 10.10.10.10.bgp: S 573929123:573929123(0)
 win 16384 <mss 1460,sackOK,eol>
```


Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Stateless Firewall Filter to Protect Against TCP and ICMP Floods on page 719
- Example: Configuring a Filter to Accept Packets Based on IPv6 TCP Flags on page 704

Example: Configuring a Stateless Firewall Filter to Protect Against TCP and ICMP Floods

This example shows how to create a stateless firewall filter that protects against TCP and ICMP denial-of-service attacks.

Requirements

No special configuration beyond device initialization is required before configuring stateless firewall filters.

Overview

In this example we create a stateless firewall filter called **protect-RE** to police TCP and ICMP packets. It uses the policers described here:

- **tcp-connection-policer**—This policer limits TCP traffic to 1,000,000 bits per second (bps) with a maximum burst size of 15,000 bytes. Traffic exceeding either limit is discarded.
- **icmp-policer**—This policer limits ICMP traffic to 1,000,000 bps with a maximum burst size of 15,000 bytes. Traffic exceeding either limit is discarded.

When specifying limits, the bandwidth limit can be from 32,000 bps to 32,000,000,000 bps and the burst-size limit can be from 1,500 bytes through 100,000,000 bytes. Use the following abbreviations when specifying limits: k (1,000), m (1,000,000), and g (1,000,000,000).

Each policer is incorporated into the action of a filter term. This example includes the following terms:

- **tcp-connection-term**—Polices certain TCP packets with a source address of 192.168.0.0/24 or 10.0.0.0/24. These addresses are defined in the **trusted-addresses** prefix list.

  Filtered packets include **tcp-established** packets The **tcp-established** match condition is an alias for the bit-field match condition **tcp-flags "(ack | rst)"**, which indicates an established TCP session, but not the first packet of a TCP connection.

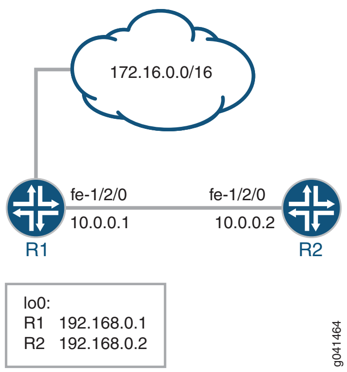
- **icmp-term**—Polices ICMP packets. All ICMP packets are counted in the **icmp-counter** counter.

NOTE: You can move terms within the firewall filter by using the **insert** command. See *insert* in the *CLI User Guide*.

You can apply a stateless firewall to the input or output sides, or both, of an interface. To filter packets transiting the device, apply the firewall filter to any non-Routing Engine interface. To filter packets originating from, or destined for, the Routing Engine, apply the firewall filter to the loopback (lo0) interface.

Figure 49 on page 720 shows the sample network.

Figure 49: Firewall Filter to Protect Against TCP and ICMP Floods



Because this firewall filter limits Routing Engine traffic to TCP packets, routing protocols that use other transport protocols for Layer 4 cannot successfully establish sessions when this filter is active. To demonstrate, this example sets up OSPF between Device R1 and Device R2.

“CLI Quick Configuration” on page 720 shows the configuration for all of the devices in Figure 49 on page 720.

The section “Step-by-Step Procedure” on page 721 describes the steps on Device R2.

Configuration

CLI Quick Configuration

To quickly configure the stateless firewall filter, copy the following commands to a text file, remove any line breaks, and then paste the commands into the CLI.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32 primary
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

```
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet filter input protect-RE
set interfaces lo0 unit 0 family inet address 192.168.0.2/32 primary
set interfaces lo0 unit 0 family inet address 172.16.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set policy-options prefix-list trusted-addresses 10.0.0.0/24
set policy-options prefix-list trusted-addresses 192.168.0.0/24
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 200
set firewall family inet filter protect-RE term tcp-connection-term from source-prefix-list
  trusted-addresses
set firewall family inet filter protect-RE term tcp-connection-term from protocol tcp
set firewall family inet filter protect-RE term tcp-connection-term from tcp-established
set firewall family inet filter protect-RE term tcp-connection-term then policer
  tcp-connection-policer
set firewall family inet filter protect-RE term tcp-connection-term then accept
set firewall family inet filter protect-RE term icmp-term from source-prefix-list
  trusted-addresses
set firewall family inet filter protect-RE term icmp-term from protocol icmp
set firewall family inet filter protect-RE term icmp-term then policer icmp-policer
set firewall family inet filter protect-RE term icmp-term then count icmp-counter
set firewall family inet filter protect-RE term icmp-term then accept
set firewall policer tcp-connection-policer filter-specific
set firewall policer tcp-connection-policer if-exceeding bandwidth-limit 1m
set firewall policer tcp-connection-policer if-exceeding burst-size-limit 15k
set firewall policer tcp-connection-policer then discard
set firewall policer icmp-policer filter-specific
set firewall policer icmp-policer if-exceeding bandwidth-limit 1m
set firewall policer icmp-policer if-exceeding burst-size-limit 15k
set firewall policer icmp-policer then discard
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure stateless firewall filter to discard :

1. Configure the device interfaces.

```
[edit interfaces fe-1/2/0 unit 0 family inet ]
user@R2# set address 10.0.0.2/30

[edit interfaces lo0 unit 0 family inet]
```

user@R2# **set address 192.168.0.2/32 primary**
user@R2# **set address 172.16.0.2/32**

2. Configure the BGP peering session.

   [edit protocols bgp group ext]
   user@R2# **set type external**
   user@R2# **set export send-direct**
   user@R2# **set neighbor 10.0.0.1 peer-as 100**

3. Configure the autonomous system (AS) number and router ID.

   [edit routing-options]
   user@R2# **set autonomous-system 200**
   user@R2# **set router-id 192.168.0.2**

4. Configure OSPF.

   [edit protocols ospf area 0.0.0.0]
   user@R2# **set interface lo0.0 passive**
   user@R2# **set interface fe-1/2/0.0**

5. Define the list of trusted addresses.

   [edit policy-options prefix-list trusted-addresses]
   user@R2# **set 10.0.0.0/24**
   user@R2# **set 192.168.0.0/24**

6. Configure a policy to advertise direct routes.

   [edit policy-options policy-statement send-direct term 1]
   user@R2# **set from protocol direct**
   user@R2# **set then accept**

7. Configure the TCP policer.

   [edit firewall policer tcp-connection-policer]
   user@R2# **set filter-specific**
   user@R2# **set if-exceeding bandwidth-limit 1m**
   user@R2# **set if-exceeding burst-size-limit 15k**
   user@R2# **set then discard**

8. Create the ICMP policer.

   [edit firewall policer icmp-policer]
   user@R2# **set filter-specific**
   user@R2# **set if-exceeding bandwidth-limit 1m**
   user@R2# **set if-exceeding burst-size-limit 15k**
   user@R2# **set then discard**

9. Configure the TCP filter rules.

   [edit firewall family inet filter protect-RE term tcp-connection-term]


```
user@R2# set from source-prefix-list trusted-addresses
user@R2# set from protocol tcp
user@R2# set from tcp-established
user@R2# set then policer tcp-connection-policer
user@R2# set then accept
```

10. Configure the ICMP filter rules.

```
[edit firewall family inet filter protect-RE term icmp-term]
user@R2# set from source-prefix-list trusted-addresses
user@R2# set from protocol icmp
user@R2# set then policer icmp-policer
user@R2# set then count icmp-counter
user@R2# set then accept
```

11. Apply the filter to the loopback interface.

```
[edit interfaces lo0 unit 0]
user@R2# set family inet filter input protect-RE
```

Results Confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, **show routing-options**, and **show firewall** commands from configuration mode. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      address 10.0.0.2/30;
    }
  }
}
lo0 {
  unit 0 {
    family inet {
      filter {
        input protect-RE;
      }
      address 192.168.0.2/32 {
        primary;
      }
      address 172.16.0.2/32;
    }
  }
}
```

```
user@R2# show protocols
bgp {
  group ext {
    type external;
    export send-direct;
    neighbor 10.0.0.1 {
```

```
            peer-as 100;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface fe-1/2/0.0;
    }
}
```

```
user@R2# show policy-options
prefix-list trusted-addresses {
    10.0.0.0/24;
    192.168.0.0/24;
}
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

```
user@R2# show routing-options
router-id 192.168.0.2;
autonomous-system 200;
```

```
user@R2# show firewall
family inet {
    filter protect-RE {
        term tcp-connection-term {
            from {
                source-prefix-list {
                    trusted-addresses;
                }
                protocol tcp;
                tcp-established;
            }
            then {
                policer tcp-connection-policer;
                accept;
            }
        }
        term icmp-term {
            from {
                source-prefix-list {
                    trusted-addresses;
                }
                protocol icmp;
            }
            then {
                policer icmp-policer;
                count icmp-counter;
                accept;
            }
```



```
        }
      }
    }
    policer tcp-connection-policer {
      filter-specific;
      if-exceeding {
        bandwidth-limit 1m;
        burst-size-limit 15k;
      }
      then discard;
    }
    policer icmp-policer {
      filter-specific;
      if-exceeding {
        bandwidth-limit 1m;
        burst-size-limit 15k;
      }
      then discard;
    }
    }
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

NOTE: To verify the TCP policer, you can use a packet generation tool. This task is not shown here.

Displaying Stateless Firewall Filter That Are in Effect

Purpose Verify the configuration of the firewall filter.

Action From operational mode, enter the **show firewall** command.

```
user@R2> show firewall
Filter: protect-RE
Counters:
Name                                                Bytes              Packets
icmp-counter                                            0                    0
Policers:
Name                                                Bytes              Packets
icmp-policer                                                         0
tcp-connection-policer                                               0
```

Meaning The output shows the filter, the counter, and the policers that are in effect on Device R2.

Using telnet to Verify the tcp-established Condition in the TCP Firewall Filter

Purpose Make sure that telnet traffic works as expected.

Action Verify that the device can establish only TCP sessions with hosts that meet the **from tcp-established** condition.

1. From Device R2, make sure that the BGP session with Device R1 is established.

```
user@R2> show bgp summary | match down
Groups: 1 Peers: 1 Down peers: 0
```

2. From Device R2, telnet to Device R1.

```
user@R2> telnet 192.168.0.1
Trying 192.168.0.1...
Connected to R1.example.net.
Escape character is '^]'.

R1 (ttyp4)

login:
```

3. From Device R1, telnet to Device R2.

```
user@R1> telnet 192.168.0.2
Trying 192.168.0.2...
telnet: connect to address 192.168.0.2: Operation timed out
telnet: Unable to connect to remote host
```

4. On Device R2, deactivate the **from tcp-established** match condition.

   [edit firewall family inet filter protect-RE term tcp-connection-term]
   user@R2# **deactivate from tcp-established**
   user@R2# **commit**

5. From Device R1, try again to telnet to Device R2.

```
user@R1> telnet 192.168.0.1
Trying 192.168.0.2...
Connected to R2.example.net.
Escape character is '^]'.

R2 (ttyp4)

login:
```

Meaning Verify the following information:

- As expected , the BGP session is established. The **from tcp-established** match condition is not expected to block BGP session establishment.
- From Device R2, you can telnet to Device R1. Device R1 has no firewall filter configured, so this is the expected behavior.
- From Device R1, you cannot telnet to Device R2. Telnet uses TCP as the transport protocol, so this result might be surprising. The cause for the lack of telnet connectivity is the **from tcp-established** match condition. This match condition limits the type of TCP traffic that is accepted of Device R2. After this match condition is deactivated, the telnet session is successful.

Using telnet to Verify the Trusted Prefixes Condition in the TCP Firewall Filter

Purpose Make sure that telnet traffic works as expected.

Action Verify that the device can establish only telnet sessions with a host at an IP address that matches one of the trusted source addresses. For example, log in to the device with the **telnet** command from another host with one of the trusted address prefixes. Also, verify that telnet sessions with untrusted source addresses are blocked.

1. From Device R1, telnet to Device R2 from an untrusted source address.

```
user@R1> telnet 172.16.0.2 source 172.16.0.1
Trying 172.16.0.2...
^C
```

2. From Device R2, add 172.16/16 to the list of trusted prefixes.

```
[edit policy-options prefix-list trusted-addresses]
user@R2# set 172.16.0.0/16
user@R2# commit
```

3. From Device R1, try again to telnet to Device R2.

```
user@R1> telnet 172.16.0.2 source 172.16.0.1
Trying 172.16.0.2...
Connected to R2.example.net.
Escape character is '^]'.

R2 (ttyp4)

login:
```

Meaning Verify the following information:

- From Device R1, you cannot telnet to Device R2 with an unstrusted source address. After the 172.16/16 prefix is added to the list of trusted prefixes, the telnet request from source address 172.16.0.1 is accepted.
- OSPF session establishment is blocked. OSPF does not use TCP as its transport protocol. After the **from protocol tcp** match condition is deactivated, OSPF session establishment is not blocked.

Using OSPF to Verify the TCP Firewall Filter

Purpose Make sure that OSPF traffic works as expected.

Action Verify that the device cannot establish OSPF connectivity.

1. From Device R1, check the OSPF sessions.

```
user@R1> show ospf neighbor
Address          Interface              State     ID               Pri  Dead
10.0.0.2         fe-1/2/0.0             Init      192.168.0.2      128    34
```

2. From Device R2, check the OSPF sessions.

```
user@R2> show ospf neighbor
```

3. From Device R2, remove the **from protocol tcp** match condition.

   [edit firewall family inet filter protect-RE term tcp-connection-term]
   user@R2# **deactivate from protocol**
   user@R2# **commit**

4. From Device R1, recheck the OSPF sessions.

```
user@R1> show ospf neighbor
Address          Interface              State     ID               Pri  Dead
10.0.0.2         fe-1/2/0.0             Full      192.168.0.2      128    36
```

5. From Device R2, recheck the OSPF sessions.

```
user@R2> show ospf neighbor
Address          Interface              State     ID               Pri  Dead
10.0.0.1         fe-1/2/0.0             Full      192.168.0.1      128    39
```

Meaning Verify the following information:

- OSPF session establishment is blocked. OSPF does not use TCP as its transport protocol. After the **from protocol tcp** match condition is deactivated, OSPF session establishment is successful.

Verifying the ICMP Firewall Filter

Purpose Verify that ICMP packets are being policed and counted. Also make sure that ping requests are discarded when the requests originate from an untrusted source address.

Action

1. Undo the configuration changes made in previous verification steps.

   Reactivate the TCP firewall settings, and delete the 172.16/16 trusted source address.

   [edit firewall family inet filter protect-RE term tcp-connection-term]
   user@R2# **activate from protocol**
   user@R2# **activate from tcp-established**


```
[edit policy-options prefix-list trusted-addresses]
user@R2# delete 172.16.0.0/16

user@R2# commit
```

2. From Device R1, ping the loopback interface on Device R2.

```
user@R1> ping 192.168.0.2 rapid count 600 size 2000
PING 192.168.0.2 (192.168.0.2): 2000 data bytes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
--- 192.168.0.2 ping statistics ---
600 packets transmitted, 536 packets received, 10% packet loss
pinground-trip min/avg/max/stddev = 2.976/3.405/42.380/2.293 ms
```

3. From Device R2, check the firewall statistics.

```
user@R2> show firewall

Filter: protect-RE
Counters:
Name                                                Bytes              Packets
icmp-counter                                      1180804                 1135
Policers:
Name                                                Bytes              Packets
icmp-policer                                                            66
tcp-connection-policer                                                   0
```

4. From an untrusted source address on Device R1, send a ping request to Device R2's loopback interface.

```
user@R1> ping 172.16.0.2 source 172.16.0.1

PING 172.16.0.2 (172.16.0.2): 56 data bytes
^C
--- 172.16.0.2 ping statistics ---
14 packets transmitted, 0 packets received, 100% packet loss
```

Meaning Verify the following information:

- The ping output shows that 10% packet loss is occurring.
- The ICMP packet counter is incrementing, and the icmp-policer is incrementing.
- Device R2 does not send ICMP responses to the **ping 172.16.0.2 source 172.16.0.1** command.

Related Documentation

- Example: Configuring a Stateless Firewall Filter to Accept Traffic from Trusted Sources on page 691
- Two-Color Policer Configuration Overview on page 1027

Example: Protecting the Routing Engine with a Packets-Per-Second Rate Limiting Filter

This example shows how to configure a packets-per-second based rate-limiting filter to improve security. It will be applied to the loopback interface in order to help protect the Routing Engine from denial of service attacks.

BEST PRACTICE: This type of filter and policer combination is only one element in a multilayered approach that can be used to help protect the Routing Engine. Other layers of protection are needed in order to fully protect the Routing Engine. See Day One: Securing the Routing Engine on M, MX, and T Series for more information.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you use a stateless firewall filter to set packets-per-second (pps) rate limits for any traffic destined for the Routing Engine through the loopback interface (lo0.0).

To activate a policer from within a stateless firewall filter configuration:

1. Create a template for the policer by including the policer ***policer-name*** statement at the **[edit firewall]** hierarchy.
2. Reference the policer in a filter term that specifies the policer in the **policer** ***policer-name*** nonterminating action.

You can also apply a policer by including the **policer (input | output)** ***policer-name*** statement in a logical interface configuration.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall policer police_pps if-exceeding-pps pps-limit 1k
set firewall policer police_pps if-exceeding-pps packet-burst 150
set firewall policer police_pps then discard
set firewall family inet filter my_pps_filter term term1 then policer police_pps
set interfaces lo0 unit 0 family inet filter input my_pps_filter
set interfaces lo0 unit 0 family inet address 127.0.0.1/32
```

Configuring the Policer and the Stateless Firewall Filter

Step-by-Step Procedure

To configure the policer ***police_pps*** and stateless firewall filter ***my_pps_filter***:

1. Configure the policer template **police_pps**.

```
[edit firewall]
user@host# set policer police_pps if-exceeding-pps pps-limit 1k
user@host# set policer police_pps if-exceeding-pps packet-burst 150
user@host# set policer police_pps then discard
```

2. Create the stateless firewall filter **my_pps_filter**.

```
[edit]
user@host# edit firewall family inet filter my_pps_filter
```

3. Configure a filter term that uses policer **police_pps** to rate limit traffic by protocol family.

```
[edit firewall family inet filter my_pps_filter]
user@host# set term term1 then policer police_pps
```


Applying the Stateless Firewall Filter to the Loopback Logical Interface

Step-by-Step Procedure

To apply the filter **my_pps_filter** to the loopback interface:

1. Configure the logical loopback interface to which you will apply the stateless firewall filter.

```
[edit]
user@host# edit interfaces lo0 unit 0
```

2. Apply the stateless firewall filter to the loopback interface.

```
[edit interfaces lo0 unit 0]
user@host# set family inet filter input my_pps_filter
```

3. Configure the interface address for the loopback interface.

```
[edit interfaces lo0 unit 0]
user@host# set family inet address 127.0.0.1/32
```

Results

Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show firewall
family inet{
  filter my_pps_filter {
    term term1 {
      then policer police_pps;
    }
  }
}
policer police_pps {
  if-exceeding-pps {
    pps-limit 1k;
    packet-burst 150;
  }
  then discard;
}
```

Confirm the configuration of the interface by entering the **show interfaces lo0** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show interfaces lo0
unit 0 {
  family inet {
    filter {
      input my_pps_filter;
    }
    address 127.0.0.1/32;
```



```
  }
}
```

If you are done configuring the device, enter commit from configuration mode.

```
user@host# commit
```

Verification

- Verifying the Operation of the Filter on page 733

Verifying the Operation of the Filter

Purpose

To confirm that the configuration is working properly, enter the **show firewall filter my_pps_filter** operational mode command.

NOTE: The following output results from running a rapid ping from another host to the router under test. In order to show results in the output, a **pps-limit** setting of **50** and a **packet-burst** setting of **10** were used during the ping test.

Action

```
user@host> show firewall filter my_pps_filter
Filter: my_pps_filter
Policers:
Name                                                Bytes              Packets
police_pps-term1                                     8704                   17
```

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- PPS-Based Policer Overview on page 982
- *if-exceeding-pps (Policer)*

Example: Configuring a Filter to Exclude DHCPv6 and ICMPv6 Control Traffic for LAC Subscriber

This example shows how to configure a standard stateless firewall filter that excludes DHCPv6 and ICMPv6 control packets from being considered for idle-timeout detection for tunneled subscribers at the LAC.

- Requirements on page 733
- Overview on page 734
- Configuration on page 734

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

Subscriber access on a LAC can be limited by configuring an idle timeout period that specifies the maximum period of time a subscriber can remain idle after the subscriber session is established. The LAC monitors the subscriber's upstream and downstream data traffic to determine whether the subscriber is inactive. Based on the session accounting statistics. the subscriber is not considered idle as long as data traffic is detected in either direction. When no traffic is detected for the duration of the idle time out, the subscriber is logged out gracefully similarly to a RADIUS-initiated disconnect or a CLI-initiated logout.

However, after a tunnel is established for L2TP subscribers, all packets through the tunnel at the LAC are treated as data packets. Consequently, the accounting statistics for the session are inaccurate and the subscriber is not considered to be idle as long as DHCPv6 and ICMPv6 control packets are being sent.

Starting in Junos OS Release 17.2R1, you can define a firewall filter for the **inet6** family with terms to match on these control packets. Include the use the **exclude-accounting** terminating action in the filter terms to drop these control packets.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set access profile v6-exclude-idle session-options client-idle-timeout 10
set access profile v6-exclude-idle session-options client-idle-timeout-ingress-only
```

```
edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER
set interface-specific
set term EXCLUDE-ACCT-DHCP-INET6 from next-header udp
set term EXCLUDE-ACCT-DHCP-INET6 from source-port 546
set term EXCLUDE-ACCT-DHCP-INET6 from source-port 547
set term EXCLUDE-ACCT-DHCP-INET6 from destination-port 546
set term EXCLUDE-ACCT-DHCP-INET6 from destination-port 547
set term EXCLUDE-ACCT-DHCP-INET6 then count exclude-acct-dhcpv6
set term EXCLUDE-ACCT-DHCP-INET6 then exclude-accounting

set term EXCLUDE-ACCT-ICMP6 from next-header icmp6
set term EXCLUDE-ACCT-ICMP6 from icmp-type router-solicit
set term EXCLUDE-ACCT-ICMP6 from icmp-type neighbor-solicit
set term EXCLUDE-ACCT-ICMP6 from icmp-type neighbor-advertisement
set term EXCLUDE-ACCT-ICMP6 then count exclude-acct-icmpv6
set term EXCLUDE-ACCT-ICMP6 then exclude-accounting
```



```
set term default then accept

top
edit dynamic-profiles pppoe-dynamic-profile interfaces pp0 unit "$junos-interface-unit"
set family inet6 filter input EXCLUDE-ACCT-INET6-FILTER
set family inet6 filter output EXCLUDE-ACCT-INET6-FILTER
set actual-transit-statistics
```

Configure the Filter

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the filter:

1. Set the idle timeout for subscriber sessions..

   ```
   [edit access profile v6-exclude-idle]
   user@host# set session-options client-idle-timeout 10
   ```

2. Specify the idle timeout applies only to ingress traffic.

   ```
   [edit access profile v6-exclude-idle]
   user@host# set session-options client-idle-timeout-ingress-only
   ```

3. Define the firewall filter term that excludes the DHCPv6 control packets from accounting statistics.

   a. Specify a match on packets with the first Next Header field set to UDP (17).

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term EXCLUDE-ACCT-DHCP-INET6 from next-header udp
   ```

   b. Specify a match on packets with a source port of 546 or 547 (DHCPv6).

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term EXCLUDE-ACCT-DHCP-INET6 from source-port 546
   user@host# set term EXCLUDE-ACCT-DHCP-INET6 from source-port 547
   ```

   c. Specify a match on packets with a DHCP destination port of 546 or 547 (DHCPv6).

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term EXCLUDE-ACCT-DHCP-INET6 from destination-port 546
   user@host# set term EXCLUDE-ACCT-DHCP-INET6 from destination-port 547
   ```

   d. Count the matched DHCPv6 packets.

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term EXCLUDE-ACCT-DHCP-INET6 then count
     exclude-acct-dhcpv6
   ```

   e. Exclude the matched DHCPv6 packets from accounting statistics.

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term EXCLUDE-ACCT-DHCP-INET6 then exclude-accounting
   ```


4. Define the firewall filter term that excludes the ICMPv6 control packets from accounting statistics.

   a. Specify a match on packets with the first Next Header field set to ICMPv6 (58).

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term EXCLUDE-ACCT-ICMP6 from next-header icmp6
   ```

   b. Specify a match on packets with an ICMPv6 message type.

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term EXCLUDE-ACCT-ICMP6 from icmp-type router-solicit
   user@host# set term EXCLUDE-ACCT-ICMP6 from icmp-type neighbor-solicit
   user@host# set term EXCLUDE-ACCT-ICMP6 from icmp-type
     neighbor-advertisement
   ```

   c. Count the matched ICMPv6 packets.

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term EXCLUDE-ACCT-ICMP6 then count exclude-acct-icmpv6
   ```

   d. Exclude the matched ICMPv6 packets from accounting statistics.

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term EXCLUDE-ACCT-DHCP-INET6 then exclude-accounting
   ```

5. Define the default filter term to accept all other packets.

   ```
   [edit firewall family inet6 filter EXCLUDE-ACCT-INET6-FILTER]
   user@host# set term default then accept
   ```

6. Configure the dynamic profile to apply the filter to input and output interfaces for the **inet6** family.

   ```
   [edit dynamic-profiles pppoe-dynamic-profile interfaces pp0 unit
     "$junos-interface-unit"]
   user@host# set family inet6 filter input EXCLUDE-ACCT-INET6-FILTER
   user@host# set family inet6 filter output EXCLUDE-ACCT-INET6-FILTER
   ```

7. Enable subscriber management accurate accounting.

   ```
   [edit dynamic-profiles pppoe-dynamic-profile interfaces pp0 unit
     "$junos-interface-unit"]
   user@host# set actual-transit-statistics
   ```

Results

From configuration mode, confirm your configuration by entering the **show access**, **show firewall**, and **show dynamic-profiles** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show access
profile v6-exclude-idle {
  session-options {
    client-idle-timeout 10;
    client-idle-timeout-ingress-only;
  }
```

```
}
user@host# show firewall
family inet6 {
   filter EXCLUDE-ACCT-INET6-FILTER {
      interface-specific;
      term EXCLUDE-ACCT-DHCP-INET6 {
         from {
            next-header udp;
            source-port [ 546 547 ];
            destination-port [ 546 547 ];
         }
         then {
            count exclude-acct-dhcpv6;
            exclude-accounting
         }
      }
      term EXCLUDE-ACCT-ICMP6 {
         from {
            next-header icmp6;
            icmp-type [ router-solicit neighbor-solicit neighbor-advertisement ]
         }
         then {
            count exclude-acct-icmpv6;
            exclude-accounting;
         }
      }
      term default {
         then accept;
      }
   }
}
user@host# show dynamic-profiles
pppoe-dynamic-profile {
   interfaces {
      pp0 {
         unit "$junos-interface-unit" {
            actual-transit-statistics;
            family inet6 {
               filter {
                  input EXCLUDE-ACCT-INET6-FILTER;
                  output EXCLUDE-ACCT-INET6-FILTER;
               }
            }
         }
      }
   }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Related Documentation

- *Classic Filters Overview*
- *Dynamically Attaching Statically Created Filters for a Specific Interface Family Type*

- Understanding How to Use Standard Firewall Filters on page 559
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Match Conditions for IPv6 Traffic on page 629

CHAPTER 17

Applying Firewall Filters to Transit Traffic

Example: Configuring a Filter for Use as an Ingress Queuing Filter

This example shows how to configure a firewall filter for use as an ingress queuing filter. The ingress queuing filter assists in traffic shaping operations by allowing you to set the forwarding class and packet loss priority, or drop the packet prior to ingress queue selection. The firewall filter must be configured within one of the following protocol families: **bridge**, **cc**, **inet**, **inet6**, **mpls**, or **vpls** and have one or more of the following actions: **accept**, **discard**, **forwarding-class**, and **loss-priority**.

The ingress queuing filter can only be used on MX Series routers with MPCs. An error is generated at commit if the ingress queuing filter is applied to an interface on any other type of port concentrator.

Requirements

This example uses the following hardware and software components:

- An MX Series router with MPC

In order for ingress queuing filters to function, **ingress-and-egress** must be configured as the **traffic-manager** mode at the **[edit chassis fpc *slot* pic *slot* traffic-manager mode]** hierarchy level.

Overview

In this example, you create a firewall filter named **iqfilter1** in the **inet** protocol family that sets the loss priority and forwarding class of packets coming from the 192.168.2.0/24 network. You then apply the **iqfilter1** filter to the ge-0/0/0.0 logical interface as an ingress queuing filter.

To configure a firewall filter and apply it for use as an ingress queuing filter involves:

- Creating a firewall filter named **iqfilter1** in the **inet** protocol family with the following two actions: **forwarding class** and **loss priority**.
- Applying the firewall filter to the ge-0/0/0.0 interface as an ingress queuing filter.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter iqfilter1 term t1 from address 192.168.2.0/24
set firewall family inet filter iqfilter1 term t1 then loss-priority low
set firewall family inet filter iqfilter1 term t1 then forwarding-class expedited-forwarding
set interfaces ge-0/0/0 unit 0 family inet ingress-queuing-filter iqfilter1
```


Configuring the Firewall Filter and Applying It to an Interface as an Input Queuing Filter

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the firewall filter, **iqfilter1**, and apply it to logical interface ge-0/0/0 unit 0:

1. Create a firewall filter named **iqfilter1**.

```
[edit firewall family inet]
user@router# set filter iqfilter1 term t1 from address 192.168.2.0/24
user@router# set filter iqfilter1 term t1 then loss-priority low
user@router# set filter iqfilter1 term t1 then forwarding-class expedited-forwarding
```

2. Apply the firewall filter to the logical interface.

```
[edit]
user@router# set interfaces ge-0/0/0 unit 0 family inet ingress-queuing-filter
  iqfilter1
```

Results

From configuration mode, confirm your configuration by entering the **show firewall** and the **show interfaces ge-0/0/0.0** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@router# show firewall
family inet {
  filter iqfilter1 {
    term t1 {
      from {
        address {
          192.168.0.0/24;
        }
      }
      then {
        loss-priority low;
        forwarding-class expedited-forwarding;
      }
    }
  }
}
user@router# show interfaces ge-0/0/0.0
family inet {
  ingress-queuing-filter iqfilter1;
}
```

If you are done configuring the device, enter **commit** from configuration mode.

```
user@router# commit
```


Related Documentation

- Multifield Classifier for Ingress Queuing on MX Series Routers with MPC on page 575
- **ingress-queuing-filter on page 1208**

Example: Configuring a Filter to Match on IPv6 Flags

This example shows how to configure a filter to match on IPv6 TCP flags.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you configure a filter to match on IPv6 TCP flags. You can use this example to configure IPv6 TCP flags in M Series, MX Series, and T Series routing devices.

Configuration

Step-by-Step Procedure

To configure a filter to match on IPv6 TCP flags:

1. Include the family statement at the firewall hierarchy level, specifying **inet6** as the protocol family.

   ```
   [edit]
   user@host# edit firewall family inet6
   ```

2. Create the stateless firewall filter.

   ```
   [edit firewall family inet6]
   user@host# edit filter tcpfilt
   ```

3. Define the first term for the filter.

   ```
   [edit firewall family inet6 filter tcpfilt]
   user@host# edit term 1
   ```

4. Define the source address match conditions for the term.

   ```
   [edit firewall family inet6 filter tcpfilt term 1]
   user@host# set from next-header tcp tcp-flags syn
   ```

5. Define the actions for the term.

   ```
   [edit firewall family inet6 filter tcpfilt term 1]
   ```



```
user@host# set then count tcp_syn_pkt log accept
```

6. If you are done configuring the device, commit the configuration.

```
[edit firewall family inet6 filter tcpfilt term 1]
user@host top

[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the **show firewall filter tcpfilt** command.

Example: Configuring a Filter to Match on Port and Protocol Fields

This example shows how to configure a standard stateless firewall filter to match on destination port and protocol fields.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you configure a stateless firewall filter that accepts all IPv4 packets except for TCP and UDP packets. TCP and UDP packets are accepted if destined for the SSH port or the Telnet port. All other packets are rejected.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level:

```
set firewall family inet filter filter1 term term1 from protocol-except tcp
```

```
set firewall family inet filter filter1 term term1 from protocol-except udp
set firewall family inet filter filter1 term term1 then accept
set firewall family inet filter filter1 term term2 from address 192.168.0.0/16
set firewall family inet filter filter1 term term2 then reject
set firewall family inet filter filter1 term term3 from destination-port ssh
set firewall family inet filter filter1 term term3 from destination-port telnet
set firewall family inet filter filter1 term term3 then accept
set firewall family inet filter filter1 term term4 then reject
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
set interfaces ge-0/0/1 unit 0 family inet filter input filter1
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter **filter1**:

1. Create the IPv4 stateless firewall filter.

   ```
   [edit]
   user@host# edit firewall family inet filter filter1
   ```

2. Configure a term to accept all traffic except for TCP and UDP packets.

   ```
   [edit firewall family inet filter filter1]
   user@host# set term term1 from protocol-except tcp
   user@host# set term term1 from protocol-except udp
   user@host# set term term1 then accept
   ```

3. Configure a term to reject packets to or from the **192.168/16** prefix.

   ```
   [edit firewall family inet filter filter1]
   user@host# set term term2 from address 192.168.0.0/16
   user@host# set term term2 then reject
   ```

4. Configure a term to accept packets destined for either the SSH port or the Telnet port.

   ```
   [edit firewall family inet filter filter1]
   user@host# set term term3 from destination-port ssh
   user@host# set term term3 from destination-port telnet
   user@host# set term term3 then accept
   ```

5. Configure the last term to reject all packets.

   ```
   [edit firewall family inet filter filter1]
   user@host# set term term4 then reject
   ```


Apply the Stateless Firewall Filter to a Logical Interface

Step-by-Step Procedure

To apply the stateless firewall filter to a logical interface:

1. Configure the logical interface to which you will apply the stateless firewall filter.

```
[edit]
user@host# edit interfaces ge-0/0/1 unit 0 family inet
```

2. Configure the interface address for the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set address 10.1.2.3/30
```

3. Apply the stateless firewall filter to the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set filter input filter1
```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter filter1 {
    term term1 {
      from {
        protocol-except [tcp udp];
      }
      then {
        accept;
      }
    }
    term term2 {
      from {
        address 192.168/16;
      }
      then {
        reject;
      }
    }
    term term3 {
      from {
        destination-port [ssh telnet];
      }
      then {
```

```
                accept;
            }
        }
        term term4 {
            then {
                reject;
            }
        }
    }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
ge-0/0/1 {
    unit 0 {
        family inet {
            filter {
                input filter1;
            }
            address 10.1.2.3/30;
        }
    }
}
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the show firewall **filter filter1** operational mode command.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Filter to Match on IPv6 Flags on page 744
- Example: Configuring a Filter to Match on Two Unrelated Criteria on page 768

Example: Configuring a Filter to Count Accepted and Rejected Packets

This example shows how to configure a firewall filter to count packets.

- Requirements on page 749
- Overview on page 749

- Configuration on page 749
- Verification on page 752

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you use a stateless firewall filter to reject all addresses except 192.168.5.0/24.

Topology

In the first term, the match condition **address 192.168.5.0/24 except** causes this address to be considered a mismatch, and this address is passed to the next term in the filter. The match condition **address 0.0.0.0/0** matches all other packets, and these are counted, logged, and rejected.

In the second term, all packets that passed though the first term (that is, packets whose address matches **192.168.5.0/24**) are counted, logged, and accepted.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configure the Stateless Firewall Filter on page 750
- Apply the Stateless Firewall Filter to a Logical Interface on page 750
- Confirm and Commit Your Candidate Configuration on page 751

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter fire1 term 1 from address 192.168.5.0/24 except
set firewall family inet filter fire1 term 1 from address 0.0.0.0/0
set firewall family inet filter fire1 term 1 then count reject_pref1_1
set firewall family inet filter fire1 term 1 then log
set firewall family inet filter fire1 term 1 then reject
set firewall family inet filter fire1 term 2 then count reject_pref1_2
set firewall family inet filter fire1 term 2 then log
set firewall family inet filter fire1 term 2 then accept
set interfaces ge-0/0/1 unit 0 family inet filter input fire1
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter **fire1**:

1. Create the stateless firewall filter **fire1**.

   ```
   [edit]
   user@host# edit firewall family inet filter fire1
   ```

2. Configure the first term to reject all addresses except those to or from the **192.168.5.0/24** prefix and then count, log, and reject all other packets.

   ```
   [edit firewall family inet filter fire1]
   user@host# set term 1 from address 192.168.5.0/24 except
   user@host# set term 1 from address 0.0.0.0/0
   user@host# set term 1 then count reject_pref1_1
   user@host# set term 1 then log
   user@host# set term 1 then reject
   ```

3. Configure the next term to count, log, and accept packets in the **192.168.5.0/24** prefix.

   ```
   [edit firewall family inet filter fire1]
   user@host# set term 2 then count reject_pref1_2
   user@host# set term 2 then log
   user@host# set term 2 then accept
   ```

Apply the Stateless Firewall Filter to a Logical Interface

Step-by-Step Procedure

To apply the stateless firewall filter to a logical interface:

1. Configure the logical interface to which you will apply the stateless firewall filter.

   ```
   [edit]
   user@host# edit interfaces ge-0/0/1 unit 0 family inet
   ```

2. Configure the interface address for the logical interface.

   ```
   [edit interfaces ge-0/0/1 unit 0 family inet]
   user@host# set address 10.1.2.3/30
   ```

3. Apply the stateless firewall filter to the logical interface.

   ```
   [edit interfaces ge-0/0/1 unit 0 family inet]
   user@host# set filter input fire1
   ```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
   filter fire1 {
      term 1 {
         from {
            address {
               192.168.5.0/24 except;
               0.0.0.0/0;
            }
         }
         then {
            count reject_pref1_1;
            log;
            reject;
         }
      }
      term 2 {
         then {
            count reject_pref1_2;
            log;
            accept;
         }
      }
   }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
ge-0/0/1 {
   unit 0 {
      family inet {
         filter {
            input fire1;
         }
         address 10.1.2.3/30;
      }
   }
}
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the **show firewall filter fire1** operational mode command. You can also display the log and individual counters separately by using the following forms of the command:

- **show firewall counter reject_pref1_1**
- **show firewall counter reject_pref1_2**
- **show firewall log**

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Filter to Count IP Options Packets on page 755
- Example: Configuring a Filter to Count and Discard IP Options Packets on page 752

Example: Configuring a Filter to Count and Discard IP Options Packets

This example shows how to configure a standard stateless firewall to count packets.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Because the filter term matches on *any* IP option value, the filter term can use the **count** nonterminating action without the **discard** terminating action or (alternatively) without requiring an interface on a 10-Gigabit Ethernet Modular Port Concentrator (MPC), 60-Gigabit Ethernet MPC, 60-Gigabit Queuing Ethernet MPC, or 60-Gigabit Ethernet Enhanced Queuing MPC on an MX Series router.

Overview

In this example, you use a standard stateless firewall filter to count and discard packets that include any IP option value but accept all other packets.

The IP option header field is an optional field in IPv4 headers only. The **ip-options** and **ip-options-except** match conditions are supported for standard stateless firewall filters and service filters only.

NOTE: On M and T series routers, firewall filters cannot count **ip-options** packets on a per option type and per interface basis. A limited work around is to use the **show pfe statistics ip options** command to see **ip-options** statistics on a per Packet Forwarding Engine (PFE) basis. See *show pfe statistics ip* for sample output.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter block_ip_options term 10 from ip-options any
set firewall family inet filter block_ip_options term 10 then count option_any
set firewall family inet filter block_ip_options term 10 then discard
set firewall family inet filter block_ip_options term 999 then accept
set interfaces ge-0/0/1 unit 0 family inet filter input block_ip_options
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter:

1. Create the stateless firewall filter **block_ip_options**.

```
[edit]
user@host# edit firewall family inet filter block_ip_options
```

2. Configure the first term to count and discard packets that include any IP options header fields.

```
[edit firewall family inet filter block_ip_options]
user@host# set term 10 from ip-options any
user@host# set term 10 then count option_any
user@host# set term 10 then discard
```

3. Configure the other term to accept all other packets.

```
[edit firewall family inet filter block_ip_options]
user@host# set term 999 then accept
```

Apply the Stateless Firewall Filter to a Logical Interface

Step-by-Step Procedure

To apply the stateless firewall filter to a logical interface:

1. Configure the logical interface to which you will apply the stateless firewall filter.

```
[edit]
user@host# edit interfaces ge-0/0/1 unit 0 family inet
```

2. Configure the interface address for the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set address 10.1.2.3/30
```

3. Apply the stateless firewall filter to the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set filter input block_ip_options
```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter block_ip_options {
    term 10 {
      from {
        ip-options any;
      }
      then {
        count option_any;
        discard;
      }
    }
    term 999 {
      then accept;
    }
  }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
```



```
ge-0/0/1 {
   unit 0 {
      family inet {
         filter {
            input block_ip_options;
         }
         address 10.1.2.3/30;
      }
   }
}
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the **show firewall filter block_ip_options** operational mode command. To display the count of discarded packets separately, enter the **show firewall count option_any** form of the command.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Filter to Count Accepted and Rejected Packets on page 748
- Example: Configuring a Filter to Count IP Options Packets on page 755

Example: Configuring a Filter to Count IP Options Packets

This example shows how use a stateless firewall filter to count individual IP options packets:

Requirements

This example uses an interface on a 10-Gigabit Ethernet Modular Port Concentrator (MPC), 60-Gigabit Ethernet MPC, 60-Gigabit Queuing Ethernet MPC, or 60-Gigabit Ethernet Enhanced Queuing MPC on an MX Series router. This interface enables you to apply an IPv4 firewall filter (standard or service filter) that can use the **count**, **log**, and **syslog** nonterminating actions on packets that match a *specific* **ip-option** value without having to also use the **discard** terminating action.

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you use a stateless firewall filter to count IP options packets but not block any traffic. Also, the filter logs packets that have loose or strict source routing.

The IP option header field is an optional field in IPv4 headers only. The **ip-options** and **ip-options-except** match conditions are supported for standard stateless firewall filters and service filters only.

NOTE: On M and T series routers, firewall filters cannot count **ip-options** packets on a per option type and per interface basis. A limited work around is to use the **show pfe statistics ip options** command to see **ip-options** statistics on a per Packet Forwarding Engine (PFE) basis. See *show pfe statistics ip* for sample output.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configure the Stateless Firewall Filter on page 757
- Apply the Stateless Firewall Filter to a Logical Interface on page 758
- Confirm and Commit Your Candidate Configuration on page 758

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter ip_options_filter term match_strict_source from ip-options
  strict-source-route
set firewall family inet filter ip_options_filter term match_strict_source then count
  strict_source_route
set firewall family inet filter ip_options_filter term match_strict_source then log
set firewall family inet filter ip_options_filter term match_strict_source then accept
set firewall family inet filter ip_options_filter term match_loose_source from ip-options
  loose-source-route
set firewall family inet filter ip_options_filter term match_loose_source then count
  loose_source_route
set firewall family inet filter ip_options_filter term match_loose_source then log
set firewall family inet filter ip_options_filter term match_loose_source then accept
set firewall family inet filter ip_options_filter term match_record from ip-options
  record-route
set firewall family inet filter ip_options_filter term match_record then count record_route
set firewall family inet filter ip_options_filter term match_record then accept
set firewall family inet filter ip_options_filter term match_timestamp from ip-options
  timestamp
set firewall family inet filter ip_options_filter term match_timestamp then count timestamp
set firewall family inet filter ip_options_filter term match_timestamp then accept
```



```
set firewall family inet filter ip_options_filter term match_router_alert from ip-options
   router-alert
set firewall family inet filter ip_options_filter term match_router_alert then count
   router_alert
set firewall family inet filter ip_options_filter term match_router_alert then accept
set firewall family inet filter ip_options_filter term match_all then accept
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
set interfaces ge-0/0/1 unit 0 family inet filter input ip_options_filter
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter **ip_option_filter**:

1. Create the stateless firewall filter **ip_option_filter**.

   ```
   [edit]
   user@host# edit firewall family inet filter ip_options_filter
   ```

2. Configure the first term to count, log, and accept packets with the **strict_source_route** IP optional header field.

   ```
   [edit firewall family inet filter ip_option_filter]
   user@host# set term match_strict_source from ip-options strict_source_route
   user@host# set term match_strict_source then count strict_source_route
   user@host# set term match_strict_source then log
   user@host# set term match_strict_source then accept
   ```

3. Configure the next term to count, log, and accept packets with the **loose-source-route** IP optional header field.

   ```
   [edit firewall family inet filter ip_option_filter]
   user@host# set term match_loose_source from ip-options loose-source-route
   user@host# set term match_loose_source then count loose_source_route
   user@host# set term match_loose_source then log
   user@host# set term match_loose_source then accept
   ```

4. Configure the next term to count and accept packets with the **record-route** IP optional header field.

   ```
   [edit firewall family inet filter ip_option_filter]
   user@host# set term match_record from ip-options record-route
   user@host# set term match_record then count record_route
   user@host# set term match_record then accept
   ```

5. Configure the next term to count and accept packets with the **timestamp** IP optional header field.

   ```
   [edit firewall family inet filter ip_option_filter]
   user@host# set term match_timestamp from ip-options timestamp
   user@host# set term match_timestamp then count timestamp
   user@host# set term match_timestamp then accept
   ```

6. Configure the next term to count and accept packets with the **router-alert** IP optional header field.

```
[edit firewall family inet filter ip_option_filter]
user@host# set term match_router_alert from ip-options router-alert
user@host# set term match_router_alert then count router_alert
user@host# set term match_router_alert then accept
```

7. Create the last term to accept any packet without incrementing any counters.

```
[edit firewall family inet filter ip_option_filter]
user@host# set term match_all then accept
```

Apply the Stateless Firewall Filter to a Logical Interface

Step-by-Step Procedure

To apply the stateless firewall filter to a logical interface:

1. Configure the logical interface to which you will apply the stateless firewall filter.

```
[edit]
user@host# edit interfaces ge-0/0/1 unit 0 family inet
```

2. Configure the interface address for the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set address 10.1.2.3/30
```

3. Apply the stateless firewall filter to the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set filter input ip_options_filter
```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter ip_options_filter {
    term match_strict_source {
      from {
        ip-options strict-source-route;
      }
      then {
        count strict_source_route;
        log;
        accept;
```



```
            }
        }
        term match_loose_source {
            from {
                ip-options loose-source-route;
            }
            then {
                count loose_source_route;
                log;
                accept;
            }
        }
        term match_record {
            from {
                ip-options record-route;
            }
            then {
                count record_route;
                accept;
            }
        }
        term match_timestamp {
            from {
                ip-options timestamp;
            }
            then {
                count timestamp;
                accept;
            }
        }
        term match_router_alert {
            from {
                ip-options router-alert;
            }
            then {
                count router_alert;
                accept;
            }
        }
        term match_all {
            then accept;
        }
    }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
ge-0/0/1 {
    unit 0 {
        family inet {
            filter {
```

```
                input ip_option_filter;
              }
              address 10.1.2.3/30;
            }
          }
        }
```

3. If you are done configuring the device, commit your candidate configuration.

   ```
   [edit]
   user@host# commit
   ```

Verification

To confirm that the configuration is working properly, enter the **show firewall filter ip_option_filter** operational mode command. You can also display the log and individual counters separately by using the following forms of the command:

- **show firewall counter strict_source_route**
- **show firewall counter loose_source_route**
- **show firewall counter record_route**
- **show firewall counter timestamp**
- **show firewall counter router_alert**
- **show firewall log**

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Filter to Count Accepted and Rejected Packets on page 748
- Example: Configuring a Filter to Count and Discard IP Options Packets on page 752

Example: Configuring a Filter to Count and Sample Accepted Packets

This example shows how to configure a standard stateless firewall filter to count and sample accepted packets.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Before you begin, configure traffic sampling by including the **sampling** statement at the **[edit forwarding-options]** hierarchy level.

Overview

In this example, you use a standard stateless firewall filter to count and sample all packets received on a logical interface.

NOTE: When you enable reverse path forwarding (RPF) on an interface with an input filter for firewall log and count, the input firewall filter does not log the packets rejected by RPF, although the rejected packets are counted. To log the rejected packets, use an RPF check fail filter.

WARNING: On MX Series routers with MPC3 or MPC4, if firewall filters are configured to count Two-Way Active Measurement Protocol (TWAMP) packets then the count is doubled for all TWAMP packets. There may also be a small increase in round trip time (RTT) when the TWAMP server is hosted on MPC3 or MPC4. This warning does not apply for routers with MPC1 or MPC2 cards.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configure the Stateless Firewall Filter on page 761
- Apply the Stateless Firewall Filter to a Logical Interface on page 762
- Confirm and Commit Your Candidate Configuration on page 762

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter sam term all then count count_sam
set firewall family inet filter sam term all then sample
set interfaces at-2/0/0 unit 301 family inet address 10.1.2.3/30
set interfaces at-2/0/0 unit 301 family inet filter input sam
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter **sam**:

1. Create the stateless firewall filter **sam**.

```
[edit]
```

```
user@host# edit firewall family inet filter sam
```

2. Configure the term to count and sample all packets.

```
[edit firewall family inet filter sam]
user@host# set term all then count count_sam
user@host# set term all then sample
```

Apply the Stateless Firewall Filter to a Logical Interface

Step-by-Step Procedure

To apply the stateless firewall filter to a logical interface:

1. Configure the logical interface to which you will apply the stateless firewall filter.

```
[edit]
user@host# edit interfaces ge-0/0/1 unit 0 family inet
```

2. Configure the interface address for the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set address 10.1.2.3/30
```

3. Apply the stateless firewall filter to the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set filter input sam
```

NOTE: The Junos OS does not sample packets originating from the router or switch. If you configure a filter and apply it to the output side of an interface, then only the transit packets going through that interface are sampled. Packets that are sent from the Routing Engine to the Packet Forwarding Engine are not sampled.

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
    filter sam {
        term all {
            then {
                count count_sam;
```



```
                sample; # default action is accept
            }
        }
    }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
interfaces {
    at-2/0/0 {
        unit 301 {
            family inet {
                filter {
                    input sam;
                }
                address 10.1.2.3/30;
            }
        }
    }
}
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

Confirm that the configuration is working properly.

Displaying the Packet Counter

Purpose Verify that the firewall filter is evaluating packets.

Action

```
user@host> show firewall filter sam
Filter:
Counters:
Name                 Bytes                 Packets
sam
sam-1                98                    8028
```

Displaying the Firewall Filter Log Output

Purpose Display the packet header information for all packets evaluated by the firewall filter.

Action

```
user@host> show firewall log
Time      Filter     A Interface         Pro Source address   Destination address
23:09:09 -          A at-2/0/0.301       TCP 10.2.0.25        10.211.211.1:80
23:09:07 -          A at-2/0/0.301       TCP 10.2.0.25        10.211.211.1:56
23:09:07 -          A at-2/0/0.301       ICM 10.2.0.25        10.211.211.1:49552
23:02:27 -          A at-2/0/0.301       TCP 10.2.0.25        10.211.211.1:56
23:02:25 -          A at-2/0/0.301       TCP 10.2.0.25        10.211.211.1:80
23:01:22 -          A at-2/0/0.301       ICM 10.2.2.101       10.211.211.1:23251
23:01:21 -          A at-2/0/0.301       ICM 10.2.2.101       10.211.211.1:16557
23:01:20 -          A at-2/0/0.301       ICM 10.2.2.101       10.211.211.1:29471
23:01:19 -          A at-2/0/0.301       ICM 10.2.2.101       10.211.211.1:26873
```

Meaning This output file contains the following fields:

- **Time**—Time at which the packet was received (not shown in the default).
- **Filter**—Name of a filter that has been configured with the **filter** statement at the **[edit firewall]** hierarchy level. A hyphen (-) or the abbreviation **pfe** indicates that the packet was handled by the Packet Forwarding Engine. A space (no hyphen) indicates that the packet was handled by the Routing Engine.
- **A**—Filter action:
  - **A**—Accept (or next term)
  - **D**—Discard
  - **R**—Reject
- **Interface**—Interface on which the filter is configured.

NOTE: **We strongly recommend that you always explicitly configure an action in the then statement.**

- **Pro**—Packet's protocol name or number.
- **Source address**—Source IP address in the packet.
- **Destination address**—Destination IP address in the packet.

Displaying the Sampling Output

Purpose Verify that the sampling output contains appropriate data.

Action

```
  wtmp.0.gz           Size: 15017, Last changed: Dec 19 13:15:54 wtmp.1.gz
       Size: 493, Last changed: Nov 19 13:47:29
  wtmp.2.gz           Size: 57, Last changed: Oct 20 15:24:34
  |                   Pipe through a command
```

```
user@host> show log /var/tmp/sam
# Apr  7 15:48:50
Time                    Dest           Src Dest Src Proto TOS Pkt Intf  IP   TCP
                        addr          addr port port          len num frag flags
Apr 7 15:48:54 192.168.9.194 192.168.9.195   0    0    1   0x0  84  8   0x0   0x0
Apr 7 15:48:55 192.168.9.194 192.168.9.195   0    0    1   0x0  84  8   0x0   0x0
Apr 7 15:48:56 192.168.9.194 192.168.9.195   0    0    1   0x0  84  8   0x0   0x0
```

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Filter to Set the DSCP Bit to Zero on page 765

Example: Configuring a Filter to Set the DSCP Bit to Zero

This example shows how to configure a standard stateless firewall filter based on the Differentiated Services code point (DSCP).

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you use a stateless firewall filter to match packets on DSCP bit patterns. If the DSCP is **2**, the packet is classified to the **best-effort** forwarding class, and the DSCP is set to **0**. If the DSCP is **3**, the packet is classified to the **best-effort** forwarding class.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configure the Stateless Firewall Filter on page 766
- Apply the Stateless Firewall Filter to a Logical Interface on page 766
- Confirm and Commit Your Candidate Configuration on page 767

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall filter filter1 term 1 from dscp 2
set firewall filter filter1 term 1 then forwarding-class best-effort
set firewall filter filter1 term 1 then dscp 0
set firewall filter filter1 term 2 from dscp 3
set firewall filter filter1 term 2 then forwarding-class best-effort
set interfaces so-0/1/0 unit 0 family inet filter input filter1
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter **filter1**:

1. Create the stateless firewall filter.

```
[edit]
user@host# edit firewall filter filter1
```

2. Configure the first term to match a packet with a DSCP of **2**, change the DSCP to **0**, and classify the packet to the **best-effort** forwarding class.

```
[edit firewall filter filter1]
user@host# set term 1 from dscp 2
user@host# set term 1 then forwarding-class best-effort
user@host# set term 1 then dscp 0
```

3. Configure the other term to match a packet with a DSCP of **3** and classify the packet to the **best-effort** forwarding class.

```
[edit firewall filter filter1]
user@host# set term 2 from dscp 3
user@host# set term 2 then forwarding-class best-effort
```

Apply the Stateless Firewall Filter to a Logical Interface

Step-by-Step Procedure

To apply the stateless firewall filter to the logical interface corresponding to the VPN routing and forwarding (VRF) instance:

1. Configure the logical interface to which you will apply the stateless firewall filter.

```
[edit]
user@host# edit interfaces so-0/1/0 unit 0 family inet
```


2. Apply the stateless firewall filter to the logical interface.

```
[ input filter1]
user@host# set filter input filter1
```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
filter filter1 {
  term term1 {
    from {
      dscp 2;
    }
    then {
      forwarding-class best-effort;
      dscp 0;
    }
  }
  term term2 {
    from {
      dscp 3;
    }
    then {
      forwarding-class best-effort;
    }
  }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
so-0/1/0 {
  unit 0 {
    family inet {
      filter input filter1;
    }
  }
}
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the following operational mode commands:

- **show class-of-service**—Displays the entire class-of-service (CoS) configuration, including system-chosen defaults.
- **show class-of-service classifier type dscp**—Displays only the classifiers of the DSCP for IPv4 type.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Filter to Count and Sample Accepted Packets on page 760

Example: Configuring a Filter to Match on Two Unrelated Criteria

This example shows how to configure a standard stateless firewall filter to match on two unrelated criteria.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you use a standard stateless firewall filter to match IPv4 packets that are either OSPF packets or packets that come from an address in the prefix **10.108/16**, and send an **administratively-prohibited** ICMP message for all packets that do not match.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter ospf_or_131 term protocol_match from protocol ospf
set firewall family inet filter ospf_or_131 term address-match from source-address
   10.108.0.0/16
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
set interfaces ge-0/0/1 unit 0 family inet filter input ospf_or_131
```

Configuring the IPv4 Firewall Filter

Step-by-Step Procedure

To configure the IPv4 firewall filter:

1. Enable configuration of the IPv4 firewall filter.

   ```
   [edit]
   user@host# edit firewall family inet filter ospf_or_131
   ```

2. Configure the first term to accept OSPF packets.

   ```
   [edit firewall family inet filter ospf_or_131]
   user@host# set term protocol_match from protocol ospf
   ```

   Packets that match the condition are accepted by default. Because another term follows this term, packets that do not match this condition are evaluated by the next term.

3. Configure the second term to accept packets from any IPv4 address in a particular prefix.

   ```
   [edit firewall family inet filter ospf_or_131]
   user@host# set term address_match from source-address 10.108.0.0/16
   ```

   Packets that match this condition are accepted by default. Because this is the last term in the filter, packets that do not match this condition are discarded by default.

Results

Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
   filter ospf_or_131 {
      term protocol_match {
         from {
            protocol ospf;
         }
      }
      term address_match {
         from {
            source-address {
```

```
                10.108.0.0/16;
              }
            }
          }
        }
      }
```

Applying the IPv4 Firewall Filter to a Logical Interface

Step-by-Step Procedure

To apply the stateless firewall filter to a logical interface:

1. Enable configuration of a logical interface.

   ```
   [edit]
   user@host# edit interfaces ge-0/0/1 unit 0 family inet
   ```

2. Configure an IP address for the logical interface.

   ```
   [edit interfaces ge-0/0/1 unit 0 family inet]
   user@host# set address 10.1.2.3/30
   ```

3. Apply the IPv4 firewall filter to the logical interface.

   ```
   [edit interfaces ge-0/0/1 unit 0 family inet]
   user@host# set filter input ospf_or_131
   ```

Results

Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-0/0/1 {
  unit 0 {
    family inet {
      filter {
        input ospf_or_131;
      }
      address 10.1.2.3/30;
    }
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

To confirm that the configuration is working properly, enter the show firewall **filter ospf_or_131** operational mode command.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559

Example: Configuring a Filter to Accept DHCP Packets Based on Address

This example shows how to configure a standard stateless firewall filter to accept packets from a trusted source.

Requirements

This example is supported only on MX Series routers and EX Series switches.

Overview

In this example, you create a filter (**rpf_dhcp**) that accepts DHCP packets with a source address of **0.0.0.0** and a destination address of **255.255.255.255**.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter rpf_dhcp term dhcp_term from source-address 0.0.0.0/32
set firewall family inet filter rpf_dhcp term dhcp_term from destination-address
  255.255.255.255/32
set firewall family inet filter rpf_dhcp term dhcp_term then accept
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
set interfaces ge-0/0/1 unit 0 family inet filter input sam
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter:

1. Create the stateless firewall filter **rpf_dhcp**.

```
[edit]
```

```
user@host# edit firewall family inet filter rpf_dhcp
```

2. Configure the term to match packets with a source address of **0.0.0.0** and a destination address of **255.255.255.255**.

```
[edit firewall family inet filter rpf_dhcp]
user@host# set term dhcp_term from source-address 0.0.0.0/32
user@host# set term dhcp_term from destination-address 255.255.255.255/32
```

3. Configure the term to accept packets that match the specified conditions.

```
[edit firewall family inet filter rpf_dhcp]
set term dhcp_term then accept
```

Apply the Firewall Filter to the Loopback Interface

Step-by-Step Procedure

To apply the filter to the input at the loopback interface:

1. Apply the **rpf_dhcp** filter if packets are not arriving on an expected path.

```
[edit]
user@host# set interfaces lo0 unit 0 family inet rpf-check fail-filter rpf_dhcp
```

2. Configure an address for the loopback interface.

```
[edit]
user@host# set interfaces lo0 unit 0 family inet address 127.0.0.1/32
```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter rpf_dhcp {
    term dhcp_term {
      from {
        source-address {
          0.0.0.0/32;
        }
        destination-address {
          255.255.255.255/32;
        }
      }
      then accept;
    }
```

```
    }
  }
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
lo0 {
  unit 0 {
    family inet {
      filter {
        rpf-check {
          fail-filter rpf_dhcp;
          mode loose;
        }
      }
      address 127.0.0.1/32;
    }
  }
}
```

3. When you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the **show firewall** operational mode command.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Stateless Firewall Filter to Accept Traffic from Trusted Sources on page 691
- Example: Configuring a Filter to Block Telnet and SSH Access on page 696
- Example: Configuring a Filter to Block TFTP Access on page 701
- Example: Configuring a Filter to Accept OSPF Packets from a Prefix on page 773

Example: Configuring a Filter to Accept OSPF Packets from a Prefix

This example shows how to configure a standard stateless firewall filter to accept packets from a trusted source.

- Requirements on page 774
- Overview on page 774

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you create a filter that accepts only OSPF packets from an address in the prefix 10.108.0.0/16, discarding all other packets with an **administratively-prohibited** ICMP message

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter ospf_filter term term1 from source-address 10.108.0.0/16
set firewall family inet filter ospf_filter term term1 from protocol ospf
set firewall family inet filter ospf_filter term term1 then accept
set firewall family inet filter ospf_filter term default-term then reject
   administratively-prohibited
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
set interfaces ge-0/0/1 unit 0 family inet filter input ospf_filter
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter **ospf_filter**:

1. Create the filter.

   ```
   [edit]
   user@host# edit firewall family inet filter ospf_filter
   ```

2. Configure the term that accepts packets.

   ```
   [edit firewall family inet filter ospf_filter]
   user@host# set term term1 from source-address 10.108.0.0/16
   user@host# set term term1 from protocol ospf
   user@host# set term term1 then accept
   ```


3. Configure the term that rejects all other packets.

```
[edit firewall family inet filter ospf_filter]
user@host# set term default_term then reject administratively-prohibited
```

Apply the Firewall Filter to the Loopback Interface

Step-by-Step Procedure

To apply the firewall filter to the loopback interface:

1. Configure the interface.

```
[edit]
user@host# edit interfaces ge-0/0/1 unit 0 family inet
```

2. Configure the logical interface IP address.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set address 10.1.2.3/30
```

3. Apply the filter to the input.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set filter input ospf_filter
```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter ospf_filter {
    term term1 {
      from {
        source-address {
          10.108.0.0/16;
        }
        protocol ospf;
      }
      then {
        accept;
      }
    }
    term default_term {
      then {
        reject administratively-prohibited; # default reject action
      }
    }
```

```
    }
  }
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
lo0 {
  unit 0 {
    family inet {
      filter {
        input ospf_filter;
      }
      address 10.1.2.3/30;
    }
  }
}
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the **show firewall filter ospf_filter** operational mode command.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Stateless Firewall Filter to Accept Traffic from Trusted Sources on page 691
- Example: Configuring a Filter to Block Telnet and SSH Access on page 696
- Example: Configuring a Filter to Block TFTP Access on page 701
- Example: Configuring a Filter to Accept DHCP Packets Based on Address on page 771

Example: Configuring a Stateless Firewall Filter to Handle Fragments

This example shows how to create a stateless firewall filter that handles packet fragments.

Requirements

No special configuration beyond device initialization is required before configuring stateless firewall filters.

Overview

In this example, you create a stateless firewall filter called **fragment-RE** that accepts fragmented packets originating from 10.2.1.0/24 and destined for the BGP port. This example includes the following firewall filter terms:

- **not-from-prefix-term**-–Discards packets that are not from 10.2.1.0/24 to ensure that subsequent terms in the firewall filter are matched against packets from 10.2.1.0/24 only.
- **small-offset-term**—Discards small (1–5) offset packets to ensure that subsequent terms in the firewall filter can be matched against all the headers in the packet. In addition, the term adds a record to the system logging destinations for the firewall facility.
- **not-fragmented-term**—Accepts unfragmented TCP packets with a destination port that specifies the BGP protocol. A packet is considered unfragmented if the MF flag is not set and the fragment offset equals 0.
- **first-fragment-term**—Accepts the first fragment of a fragmented TCP packet with a destination port that specifies the BGP protocol.
- **fragment-term**—Accepts all fragments that were not discarded by **small-offset-term**. (packet fragments 6–8191). However, only those fragments that are part of a packet containing a first fragment accepted by **first-fragment-term** are reassembled by the destination device.

Packet fragments offset can be from 1 through 8191.

NOTE: You can move terms within the firewall filter by using the **insert** command. For more information, see “*insert*” in the *CLI User Guide*.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter fragment-RE term not-from-prefix-term from source-address
  0.0.0.0/0
set firewall family inet filter fragment-RE term not-from-prefix-term from source-address
  10.2.1.0/24 except
set firewall family inet filter fragment-RE term not-from-prefix-term then discard
set firewall family inet filter fragment-RE term small-offset-term from fragment-offset
  1-5
```

```
set firewall family inet filter fragment-RE term small-offset-term then syslog
set firewall family inet filter fragment-RE term small-offset-term then discard
set firewall family inet filter fragment-RE term not-fragmented-term from fragment-offset
   0
set firewall family inet filter fragment-RE term not-fragmented-term from fragment-flags
   "!more-fragments"
set firewall family inet filter fragment-RE term not-fragmented-term from protocol tcp
set firewall family inet filter fragment-RE term not-fragmented-term from destination-port
   bgp
set firewall family inet filter fragment-RE term not-fragmented-term then accept
set firewall family inet filter fragment-RE term first-fragment-term from first-fragment
set firewall family inet filter fragment-RE term first-fragment-term from protocol tcp
set firewall family inet filter fragment-RE term first-fragment-term from destination-port
   bgp
set firewall family inet filter fragment-RE term first-fragment-term then accept
set firewall family inet filter fragment-RE term fragment-term from fragment-offset
   6-8191
set firewall family inet filter fragment-RE term fragment-term then accept
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the stateless firewall filter:

1. Define the stateless firewall filter.

   ```
   [edit]
   user@host# edit firewall family inet filter fragment-RE
   ```

2. Configure the first term for the filter.

   ```
   [edit firewall family inet filter fragment-RE ]
   user@host# set term not-from-prefix-term from source-address 0.0.0.0/0
   user@host# set term not-from-prefix-term from source-address 10.2.1.0/24 except
   user@host# set term not-from-prefix-term then discard
   ```

3. Define the second term for the filter.

   ```
   [edit firewall family inet filter fragment-RE]
   user@host# edit term small-offset-term
   ```

4. Define the match conditions for the term.

   ```
   [edit firewall family inet filter fragment-RE term small-offset-term]
   user@host# set from fragment-offset 1-5
   ```

5. Define the action for the term.

   ```
   [edit firewall family inet filter fragment-RE term small-offset-term]
   user@host# set then syslog discard
   ```

6. Define the third term for the filter.

   [edit]
   user@host# **edit firewall family inet filter fragment-RE term not-fragmented-term**

7. Define the match conditions for the term.

   [edit firewall family inet filter fragment-RE term not-fragmented-term]
   user@host# **set from fragment-flags "!more-fragments" fragment-offset 0 protocol tcp destination-port bgp**

8. Define the action for the term.

   [edit firewall family inet filter fragment-RE term not-fragmented-term]
   user@host# **set then accept**

9. Define the fourth term for the filter.

   [edit]
   user@host# **edit firewall family inet filter fragment-RE term first-fragment-term**

10. Define the match conditions for the term.

    [edit firewall family inet filter fragment-RE term first-fragment-term]
    user@host# **set from first-fragment protocol tcp destination-port bgp**

11. Define the action for the term.

    [edit firewall family inet filter fragment-RE term first-fragment-term]
    user@host# **set then accept**

12. Define the last term for the filter.

    [edit]
    user@host# **edit firewall family inet filter fragment-RE term fragment-term**

13. Define the match conditions for the term.

    [edit firewall family inet filter fragment-RE term fragment-term]
    user@host# **set from fragment-offset 6–8191**

14. Define the action for the term.

    [edit firewall family inet filter fragment-RE term fragment-term]
    user@host# **set then accept**

Results Confirm your configuration by entering the **show firewall** command from configuration mode. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show firewall
family inet {
  filter fragment-RE {
    term not-from-prefix-term {
      from {
        source-address {
          0.0.0.0/0;
          10.2.1.0/24 except;
        }
      }
      then discard;
    }
    term small-offset-term {
      from {
        fragment-offset 1-5;
      }
      then {
        syslog;
        discard;
      }
    }
    term not-fragmented-term {
      from {
        fragment-offset 0;
        fragment-flags "!more-fragments";
        protocol tcp;
        destination-port bgp;
      }
      then accept;
    }
    term first-fragment-term {
      from {
        first-fragment;
        protocol tcp;
        destination-port bgp;
      }
      then accept;
    }
    term fragment-term {
      from {
        fragment-offset 6-8191;
      }
      then accept;
    }
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

To confirm that the configuration is working properly, perform these tasks:

- Displaying Stateless Firewall Filter Configurations on page 781
- Verifying a Firewall Filter that Handles Fragments on page 781

Displaying Stateless Firewall Filter Configurations

Purpose Verify the configuration of the firewall filter. You can analyze the flow of the filter terms by displaying the entire configuration.

Action From configuration mode, enter the **show firewall** command.

Meaning Verify that the output shows the intended configuration of the firewall filter. In addition, verify that the terms are listed in the order in which you want the packets to be tested. You can move terms within a firewall filter by using the **insert** CLI command.

Verifying a Firewall Filter that Handles Fragments

Purpose Verify that the actions of the firewall filter terms are taken.

Action Send packets to the device that match the terms.

Meaning Verify that packets from 10.2.1.0/24 with small fragment offsets are recorded in the device's system logging destinations for the firewall facility.

Related Documentation

- *show route summary*

Configuring a Firewall Filter to Prevent or Allow IPv4 Packet Fragmentation

This topic explains how to use the **dont-fragment (set | clear)** actions in an ingress firewall filter to modify the Don't Fragment flag in IPv4 packet headers. These actions are supported only on MPCs in MX Series routers.

You can use a firewall filter on an ingress interface to match IPv4 packets that have the Don't Fragment flag set to one or cleared to zero. Fragmentation is prevented when this flag is set in the packet header. Fragmentation is allowed when the flag is not set.

To prevent an IPv4 packet from being fragmented:

- Configure a filter term that modifies the Don't Fragment flag to one.

  [edit firewall family inet filter dfSet]
  user@host# **set term t1 then dont-fragment set**

To allow an IPv4 packet to be fragmented:

- Configure a filter term that modifies the Don't Fragment flag to zero.

  [edit firewall family inet filter dfClear]
  user@host# **set term t1 then dont-fragment clear**

In the following example, the dfSet firewall filter matches packets that are fragmented and changes the Don’t Fragment flag to prevent fragmentation. The dfClear firewall filter matches packets that are not fragmented and changes the Don’t Fragment flag to allow fragmentation.

```
[edit firewall family inet]
user@host# edit filter dfSet
user@host# set term t1 from fragment-flags is-fragment
user@host# set term t1 then dont-fragment set
user@host# up
user@host# edit filter dfClear
user@host# set term t1 from fragment-flags dont-fragment
user@host# set term t1 then dont-fragment clear
```

Related Documentation

- Firewall Filter Match Conditions for IPv4 Traffic on page 589
- Firewall Filter Nonterminating Actions on page 673
- Stateless Firewall Filter Components on page 561
- Stateless Firewall Filter Overview on page 558

Configuring a Firewall Filter to Discard Ingress IPv6 Packets with a Mobility Extension Header

This topic explains how to configure a firewall filter to discard IPv6 packets that contain a mobility extension header. This feature is supported only on MPCs in MX Series routers.

To configure the stateless firewall filter:

1. Create the stateless firewall filter.

   ```
   [edit]
   user@host# edit firewall family inet6 filter filter-name
   ```

   For example:

   ```
   [edit]
   user@host# edit firewall family inet6 filter drop-mobility
   ```

2. Configure a term to discard all traffic that contains a mobility extension header.

   ```
   [edit firewall family inet6 filter drop-mobility]
   user@host# set term term1 from extension-header mobility
   user@host# set term term1 then discard
   ```

3. Configure a term to accept all other traffic.

   ```
   [edit firewall family inet6 filter drop-mobility]
   user@host# set term term2 then accept
   ```

4. Apply the firewall filter to a logical interface.

   ```
   [edit interfaces ge-1/2/10 unit 0 family inet6]
   ```

user@host# **set filter input drop-mobility**

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Firewall Filter Match Conditions for IPv6 Traffic on page 629

Example: Configuring a Rate-Limiting Filter Based on Destination Class

This example shows how to configure a rate-limiting stateless firewall filter.

- Requirements on page 783
- Overview on page 783
- Configuration on page 783
- Verification on page 786

Requirements

No special configuration beyond device initialization is required before configuring this example.

Before you begin, configure the destination class **class1**.

Overview

In this example, you use a stateless firewall filter to set rate limits based on a destination class.

To activate a policer from within a stateless firewall filter configuration:

- Create a template for the policer by including the policer ***policer-name*** statement.
- Reference the policer in a filter term that specifies the policer in the **policer *policer-name*** nonterminating action.

You can also activate a policer by including the **policer (input | output) *policer-template-name*** statement at a logical interface.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

- Configure the Stateless Firewall Filter on page 784
- Apply the Stateless Firewall Filter to a Logical Interface on page 784
- Confirm and Commit Your Candidate Configuration on page 785

CLI Quick Configuration

To quickly configure this example, copy the following commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall filter rl_dclass1 policer police_class1 if-exceeding bandwidth-limit 25
```

```
set firewall filter rl_dclass1 policer police_class1 if-exceeding burst-size-limit 1000
set firewall filter rl_dclass1 policer police_class1 then discard
set firewall filter rl_dclass1 term term1 from destination-class class1
set firewall filter rl_dclass1 term term1 then policer police_class1
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
set interfaces ge-0/0/1 unit 0 family inet filter input rl_dclass1
```

Configure the Stateless Firewall Filter

Step-by-Step Procedure

To configure the stateless firewall filter **rl_dclass1** with policer **police_class1** for destination class **class1**:

1. Create the stateless firewall filter **rl_dclass1**.

```
[edit]
user@host# edit firewall filter rl_dclass1
```

2. Configure the policer template **police_class1**.

```
[edit firewall filter rl_dclass1]
user@host# set policer police_class1 if-exceeding bandwidth-limit 25
user@host# set policer police_class1 if-exceeding burst-size-limit 1000
user@host# set policer police_class1 then discard
```

3. Configure a filter term that uses policer **police_class1** to rate-limit traffic for destination class **class1**.

```
[edit firewall filter rl_dclass1]
user@host# set term term1 from destination-class class1
user@host# set term term1 then policer police_class1
```

Apply the Stateless Firewall Filter to a Logical Interface

Step-by-Step Procedure

To apply the filter **rl_dclass1** to a logical interface:

1. Configure the logical interface to which you will apply the stateless firewall filter.

```
[edit]
user@host# edit interfaces ge-0/0/1 unit 0 family inet
```

2. Configure the interface address for the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set address 10.1.2.3/30
```

3. Apply the stateless firewall filter to the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set filter input rl_dclass1
```


Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
filter rl_dclass1 {
  policer police_class1 {
    if-exceeding {
      bandwidth-limit 25;
      burst-size-limit 1000;
    }
    then {
      discard;
    }
  }
  term term1 {
    from {
      destination-class class1;
    }
    then {
      policer police_class1;
    }
  }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
ge-0/0/1 {
  unit 0 {
    family inet {
      filter {
        input rl_dclass1;
      }
      address 10.1.2.3/30;
    }
  }
}
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the **show class-of-service ge-0/0/1** operational mode command.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Filtering Packets Received on an Interface Set Overview on page 862
- Example: Filtering Packets Received on an Interface Set on page 707

CHAPTER 18

Configuring Firewall Filters in Logical Systems

Firewall Filters in Logical Systems Overview

This topic covers the following information:

Logical Systems

With the Junos OS, you can partition a single physical router or switch into multiple logical devices that perform independent routing tasks. Because logical systems perform a subset of the tasks once handled by the physical router or switch, logical systems offer an effective way to maximize the use of a single router or switch.

Firewall Filters in Logical Systems

You can configure a separate set of firewall filters for each logical system on a router or switch. To configure a filter in a logical system, you must define the filter in the **firewall** stanza at the **[edit logical-systems *logical-system-name*]** hierarchy level, and you must apply the filter to a logical interface that is also configured at the **[edit logical-systems *logical-system-name*]** hierarchy level.

Identifiers for Firewall Objects in Logical Systems

To identify firewall objects configured under logical systems, operational **show** commands and firewall-related SNMP MIB objects include a **__logical-system-name/** prefix in the object name. For example, firewall objects configured under the **ls1** logical system include **__ls1/** as the prefix.

Related Documentation

- Stateless Firewall Filter Types on page 560
- Guidelines for Configuring and Applying Firewall Filters in Logical Systems on page 788
- Unsupported Firewall Filter Statements for Logical Systems on page 817
- Unsupported Actions for Firewall Filters in Logical Systems on page 819
- Example: Configuring a Stateless Firewall Filter to Protect a Logical System Against ICMP Floods on page 814
- “*Introduction to Logical Systems*
- ”*Logical Systems Operations and Restrictions*

Guidelines for Configuring and Applying Firewall Filters in Logical Systems

This topic covers the following information:

Statement Hierarchy for Configuring Firewall Filters in Logical Systems

To configure a firewall filter in a logical system, include the **filter**, **service-filter**, or **simple-filter** statement at the **[edit logical-systems *logical-system-name* firewall family *family-name*]** hierarchy level.

```
[edit]
logical systems {
  logical-system-name {
    firewall {
      family family-name {
        filter filter-name {
          interface-specific;
          physical-interface-filter;
          term term-name {
            filter filter-name;
            from {
```



```
                match-conditions;
              }
              then {
                actions;
              }
            }
          }
          service-filter filter-name { # For 'family inet' or 'family inet6' only.
            term term-name {
              from {
                match-conditions;
              }
              then {
                actions;
              }
            }
          }
          simple-filter filter-name { # For 'family inet' only.
            term term-name {
              from {
                match-conditions;
              }
              then {
                actions;
              }
            }
          }
        }
      }
    }
  }
```

Filter Types in Logical Systems

There are no special restrictions on the types of stateless firewall filter types that you can configure in logical systems.

In a logical system, you can use the same types of stateless firewall filters that are available on a physical router or switch:

- Standard stateless firewall filters
- Service filters
- Simple filters

Firewall Filter Protocol Families in Logical Systems

There are no special restrictions on the protocol families supported with stateless firewall filters in logical systems.

In a logical system, you can filter the same protocol families as you can on a physical router or switch.

- Standard stateless firewall filters—In logical systems, you can filter the following traffic types: protocol-independent, IPv4, IPv6, MPLS, MPLS-tagged IPv4 or IPv6, VPLS, Layer 2 circuit cross-connection, and Layer 2 bridging.
- Service filters—In logical systems, you can filter IPv4 and IPv6 traffic.
- Simple filters—In logical systems, you can filter IPv4 traffic only.

Firewall Filter Match Conditions in Logical Systems

There are no special restrictions on the match conditions supported with stateless firewall filters in logical systems.

Firewall Filter Actions in Logical Systems

There are no special restrictions on the actions supported with stateless firewall filters in logical systems.

Statement Hierarchy for Applying Firewall Filters in Logical Systems

To apply a firewall filter in a logical system, include the **filter** ***filter-name***, **service-filter** ***service-filter-name***, or **simple-filter** ***simple-filter-name*** statement to a logical interface in the logical system.

The following configuration shows the hierarchy levels at which you can apply the statements:

```
[edit]
logical-systems logical-system-name {
  interfaces {
    interface-name {
      unit logical-unit-number {
        family family-name {
          filter {
            group group-name;
            input filter-name;
            input-list [ filter-names ];
            output filter-name;
            output-list [ filter-names ]
          }
          rpf-check { # For 'family inet' or 'family inet6' only.
            fail-filter filter-name;
            mode loose;
          }
          service { # For 'family inet' or 'family inet6' only.
            input {
              service-set service-set-name <service-filter service-filter-name>;
              post-service-filter service-filter-name;
            }
            output {
              service-set service-set-name <service-filter service-filter-name>;
            }
          }
```



```
                    simple-filter { # For 'family inet' only.
                        input simple-filter-name;
                    }
                }
            }
        }
    }
}
```

Related Documentation

- Firewall Filters in Logical Systems Overview on page 787
- References from a Firewall Filter in a Logical System to Subordinate Objects on page 791
- References from a Firewall Filter in a Logical System to Nonfirewall Objects on page 792
- References from a Nonfirewall Object in a Logical System to a Firewall Filter on page 794
- Example: Configuring a Stateless Firewall Filter to Protect a Logical System Against ICMP Floods on page 814
- Unsupported Firewall Filter Statements for Logical Systems on page 817
- Unsupported Actions for Firewall Filters in Logical Systems on page 819

References from a Firewall Filter in a Logical System to Subordinate Objects

This topic covers the following information:

- Resolution of References from a Firewall Filter to Subordinate Objects on page 791
- Valid Reference from a Firewall Filter to a Subordinate Object on page 791

Resolution of References from a Firewall Filter to Subordinate Objects

If a firewall filter defined in a logical system references a subordinate object (for example, a policer or prefix list), that subordinate object must be defined within the **firewall** stanza of the same logical system. For example, if a firewall filter configuration references a policer, the firewall filter and the policer must be configured under the same **[edit logical-systems *logical-system-name* firewall]** hierarchy level.

This rule applies even if the same policer is configured under the main firewall configuration or if the same policer is configured as part of a firewall in another logical system.

Valid Reference from a Firewall Filter to a Subordinate Object

In this example, the firewall filter **filter1** references the policer **pol1**. Both **filter1** and **pol1** are defined under the same firewall object. This configuration is valid. If **pol1** had been defined under another firewall object, the configuration would not be valid.

```
[edit]
logical systems {
    ls-A {
        firewall {
            policer pol1 {
                if-exceeding {
```

```
                bandwidth-limit 401k;
                burst-size-limit 50k;
            }
            then discard;
        }
        filter filter1 {
            term one {
                from {
                    source-address 12.1.0.0/16;
                }
                then {
                    reject host-unknown;
                }
            }
            term two {
                from {
                    source-address 12.2.0.0/16;
                }
                then policer pol1;
            }
        }
    }
  }
}
```

Related Documentation

- Firewall Filters in Logical Systems Overview on page 787
- Guidelines for Configuring and Applying Firewall Filters in Logical Systems on page 788
- References from a Firewall Filter in a Logical System to Nonfirewall Objects on page 792
- References from a Nonfirewall Object in a Logical System to a Firewall Filter on page 794

References from a Firewall Filter in a Logical System to Nonfirewall Objects

This topic covers the following information:

- Resolution of References from a Firewall Filter to Nonfirewall Objects on page 792
- Valid Reference to a Nonfirewall Object Outside of the Logical System on page 793

Resolution of References from a Firewall Filter to Nonfirewall Objects

In many cases, a firewall configuration references objects outside the firewall configuration. As a general rule, the referenced object must be defined under the same logical system as the referencing object. However, there are cases when the configuration of the referenced object is not supported at the **[edit logical-systems *logical-system-name*]** hierarchy level.

Valid Reference to a Nonfirewall Object Outside of the Logical System

This example configuration illustrates an exception to the general rule that the objects referenced by a firewall filter in a logical system must be defined under the same logical system as the referencing object.

In the following scenario, the service filter **inetsf1** is applied to IPv4 traffic associated with the service set **fred** at the logical interface **fe-0/3/2.0**, which is on an adaptive services interface.

- Service filter **inetsf1** is defined in **ls-B** and references prefix list **prefix1**.
- Service set **fred** is defined at the main services hierarchy level, and the policy framework software searches the **[edit services]** hierarchy for the definition of the **fred** service set.

Because service rules cannot be configured in logical systems. firewall filter configurations in the **[edit logical-systems logical-system *logical-system-name*]** hierarchy are allowed to reference *service sets* outside the logical system hierarchy.

```
[edit]
logical-systems {
  ls-B {
    interfaces {
      fe-0/3/2 {
        unit 0 {
          family inet {
            service {
              input {
                service-set fred service-filter inetsf1;
              }
            }
          }
        }
      }
    }
    policy-options {
      prefix-list prefix1 {
        1.1.0.0/16;
        1.2.0.0/16;
        1.3.0.0/16;
      }
    }
    firewall { # Under logical-system 'ls-B'.
      family inet {
        filter filter1 {
          term one {
            from {
              source-address {
                12.1.0.0/16;
              }
            }
            then {
              reject host-unknown;
            }
          }
          term two {
```

```
                from {
                    source-address {
                        12.2.0.0/16;
                    }
                }
                then policer pol1;
            }
        }
        service-filter inetsf1 {
            term term1 {
                from {
                    source-prefix-list {
                        prefix1;
                    }
                }
                then count prefix1;
            }
        }
    }
    policer pol1 {
        if-exceeding {
            bandwidth-limit 401k;
            burst-size-limit 50k;
        }
        then discard;
    }
  }
 }
} # End of logical systems configuration.
services { # Main services hierarchy level.
    service-set fred {
        max-flows 100;
        interface-service {
            service-interface sp-1/2/0.0;
        }
    }
}
```

Related Documentation

- Firewall Filters in Logical Systems Overview on page 787
- Guidelines for Configuring and Applying Firewall Filters in Logical Systems on page 788
- References from a Firewall Filter in a Logical System to Subordinate Objects on page 791
- References from a Nonfirewall Object in a Logical System to a Firewall Filter on page 794

References from a Nonfirewall Object in a Logical System to a Firewall Filter

This topic covers the following information:

- Resolution of References from a Nonfirewall Object to a Firewall Filter on page 795
- Invalid Reference to a Firewall Filter Outside of the Logical System on page 795

- Valid Reference to a Firewall Filter Within the Logical System on page 796
- Valid Reference to a Firewall Filter Outside of the Logical System on page 798

Resolution of References from a Nonfirewall Object to a Firewall Filter

If a nonfirewall filter object in a logical system references an object in a firewall filter configured in a logical system, the reference is resolved using the following logic:

- If the nonfirewall filter object is configured in a logical system that includes firewall filter configuration statements, the policy framework software searches the **[edit logical-systems *logical-system-name* firewall]** hierarchy level. Firewall filter configurations that belong to *other* logical systems or to the main **[edit firewall]** hierarchy level are not searched.
- If the nonfirewall filter object is configured in a logical system that does not include any firewall filter configuration statements, the policy framework software searches the firewall configurations defined at the **[edit firewall]** hierarchy level.

Invalid Reference to a Firewall Filter Outside of the Logical System

This example configuration illustrates an unresolvable reference from a nonfirewall object in a logical system to a firewall filter.

In the following scenario, the stateless firewall filters **filter1** and **fred** are applied to the logical interface **fe-0/3/2.0** in the logical system **ls-C**.

- Filter **filter1** is defined in **ls-C**.
- Filter **fred** is defined in the main firewall configuration.

Because **ls-C** contains firewall filter statements (for **filter1**), the policy framework software resolves references to and from firewall filters by searching the **[edit logical systems ls-C firewall]** hierarchy level. Consequently, the reference from **fe-0/3/2.0** in the logical system to **fred** in the main firewall configuration cannot be resolved.

```
[edit]
logical-systems {
  ls-C {
    interfaces {
      fe-0/3/2 {
        unit 0 {
          family inet {
            filter {
              input-list [ filter1 fred ];
            }
          }
        }
      }
    }
    firewall { # Under logical system 'ls-C'.
      family inet {
        filter filter1 {
          term one {
            from {
              source-address 12.1.0.0/16;
```

```
                    }
                    then {
                        reject host-unknown;
                    }
                }
                term two {
                    from {
                        source-address 12.2.0.0/16;
                    }
                    then policer pol1;
                }
            }
        }
        policer pol1 {
            if-exceeding {
                bandwidth-limit 401k;
                burst-size-limit 50k;
            }
            then discard;
        }
    }
}
} # End of logical systems
firewall { # Under the main firewall hierarchy level
    family inet {
        filter fred {
            term one {
                from {
                    source-address 11.1.0.0/16;
                }
                then {
                    log;
                    reject host-unknown;
                }
            }
        }
    }
} # End of main firewall configurations.
```

Valid Reference to a Firewall Filter Within the Logical System

This example configuration illustrates resolvable references from a nonfirewall object in a logical system to two firewall filter.

In the following scenario, the stateless firewall filters **filter1** and **fred** are applied to the logical interface **fe-0/3/2.0** in the logical system **ls-C**.

- Filter **filter1** is defined in **ls-C**.
- Filter **fred** is defined in **ls-C** and also in the main firewall configuration.

Because **ls-C** contains firewall filter statements, the policy framework software resolves references to and from firewall filters by searching the **[edit logical systems ls-C firewall]** hierarchy level. Consequently, the references from **fe-0/3/2.0** in the logical system to **filter1** and **fred** use the stateless firewall filters configured in **ls-C**.

```
[edit]
```

```
logical-systems {
  ls-C {
    interfaces {
      fe-0/3/2 {
        unit 0 {
          family inet {
            filter {
              input-list [ filter1 fred ];
            }
          }
        }
      }
    }
    firewall { # Under logical system 'ls-C'.
      family inet {
        filter filter1 {
          term one {
            from {
              source-address 12.1.0.0/16;
            }
            then {
              reject host-unknown;
            }
          }
          term two {
            from {
              source-address 12.2.0.0/16;
            }
            then policer pol1;
          }
        }
        filter fred { # This 'fred' is in 'ls-C'.
          term one {
            from {
              source-address 10.1.0.0/16;
            }
            then {
              log;
              reject host-unknown;
            }
          }
        }
      }
      policer pol1 {
        if-exceeding {
          bandwidth-limit 401k;
          burst-size-limit 50k;
        }
        then discard;
      }
    }
  }
} # End of logical systems configurations.
firewall { # Main firewall filter hierarchy level
  family inet {
    filter fred {
```

```
            term one {
              from {
                source-address 11.1.0.0/16;
              }
              then {
                log;
                reject host-unknown;
              }
            }
          }
        }
      } # End of main firewall configurations.
```

Valid Reference to a Firewall Filter Outside of the Logical System

This example configuration illustrates resolvable references from a nonfirewall object in a logical system to two firewall filter.

In the following scenario, the stateless firewall filters **filter1** and **fred** are applied to the logical interface **fe-0/3/2.0** in the logical system **ls-C**.

- Filter **filter1** is defined in the main firewall configuration.
- Filter **fred** is defined in the main firewall configuration.

Because **ls-C** does not contain any firewall filter statements, the policy framework software resolves references to and from firewall filters by searching the **[edit firewall]** hierarchy level. Consequently, the references from **fe-0/3/2.0** in the logical system to **filter1** and **fred** use the stateless firewall filters configured in the main firewall configuration.

```
[edit]
logical-systems {
  ls-C {
    interfaces {
      fe-0/3/2 {
        unit 0 {
          family inet {
            filter {
              input-list [ filter1 fred ];
            }
          }
        }
      }
    }
  }
} # End of logical systems configurations.
firewall { # Main firewall hierarchy level.
  family inet {
    filter filter1 {
      term one {
        from {
          source-address 12.1.0.0/16;
        }
        then {
          reject host-unknown;
        }
      }
```



```
                term two {
                    from {
                        source-address 12.2.0.0/16;
                    }
                    then policer pol1;
                }
            }
            filter fred {
                term one {
                    from {
                        source-address 11.1.0.0/16;
                    }
                    then {
                        log;
                        reject host-unknown;
                    }
                }
            }
        }
        policer pol1 {
            if-exceeding {
                bandwidth-limit 701k;
                burst-size-limit 70k;
            }
            then discard;
        }
    } # End of main firewall configurations.
```

Related Documentation

- Firewall Filters in Logical Systems Overview on page 787
- Guidelines for Configuring and Applying Firewall Filters in Logical Systems on page 788
- References from a Firewall Filter in a Logical System to Subordinate Objects on page 791
- References from a Firewall Filter in a Logical System to Nonfirewall Objects on page 792

Example: Configuring Filter-Based Forwarding

Filter-based forwarding (FBF), which is also called Policy Based Routing (PBR), provides a a simple but powerful way to route IP traffic to different interfaces on the basis of Layer-3 or Layer-4 parameters.

FBF works by using match conditions in a firewall filter to select certain traffic and then direct it to a given routing instance that points to the desired next hop. To ensure the next hop is resolvable, interface routes from the main routing table are shared via RIB group with the routing table(s) specified in the routing instance(s).

Match conditions can include the source or destination IP address, source or destination port, IP protocol, DSCP value, TCP flag, ICMP type, and packet length.

Requirements

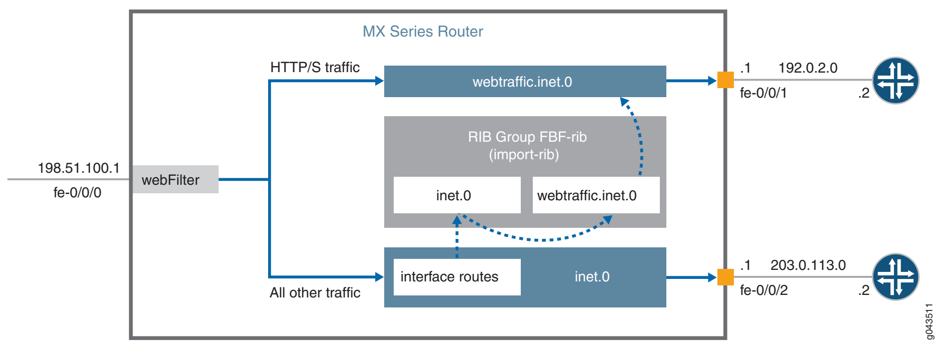
This example has the following hardware and software requirements:

- MX Series 3D Universal Edge Router as the routing device with the firewall filter configured.
- Junos OS Release 13.3 or later running on the routing device with the firewall filter configured.

Overview

This example shows the configuration settings you need to set up filter-based forwarding on a single device. Figure 50 on page 800 shows the ingress and egress interfaces on an MX Series router and illustrates the logical flow of events as packets traverse the device.

Figure 50: Filter-Based Forwarding to Specified Interfaces



A firewall filter called **webFilter** is attached to the ingress interface, **fe-0/0/0**. Packets arriving over the interface are evaluated against the match conditions specified in the filter, the logic of which directs HTTP and HTTPS traffic to a routing instance called **webtraffic**. This routing instance accomplishes three things: first, it establishes a routing table called **webtraffic.inet.0**; second, it lets you define a static route and next hop; and third, lets you configure the instance for forwarding traffic to the next hop (here, 192.0.2.2 on interface **fe-0/0/1**).

Term 2 in the firewall filter, **then accept**, specifies that all non-matching traffic take a different path. We define a static route with next hop of 203.0.113.2 to have this traffic egress the device via **fe-0/0/2**. The route is automatically installed in the master routing table, **inet.0**.

The last (logical) step in setting up FBF is to ensure that both routes are resolvable. The RIB group (**FBF-rib** in this example) makes it so interface-routes from **inet.0** can be shared with **webtraffic.inet.0**.

For examples that focus on a specific use case or multi-device topologies, see the Related Topics.

Configuration

CLI Quick Configuration

Both copy-paste and step-by-step instructions for creating filter-based forwarding on a single device are provided.

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Configure a device for filter-based forwarding

```
set interfaces fe-0/0/0 unit 0 family inet address 198.51.100.1/24
set interfaces fe-0/0/0 unit 0 family inet filter input webFilter
set interfaces fe-0/0/1 unit 0 family inet address 192.0.2.1/24
set interfaces fe-0/0/2 unit 0 family inet address 203.0.113.1/24
set firewall family inet filter webFilter term 1 from destination-port http
set firewall family inet filter webFilter term 1 from destination-port https
set firewall family inet filter webFilter term 1 then routing-instance webtraffic
set firewall family inet filter webFilter term 2 then accept
set routing-instances webtraffic routing-options static route 0.0.0.0/0 next-hop 192.0.2.2
set routing-instances webtraffic instance-type forwarding
set routing-options interface-routes rib-group inet int-routes
set routing-options static route 0.0.0.0/0 next-hop 203.0.113.2
set routing-options rib-groups FBF-rib import-rib inet.0
set routing-options rib-groups FBF-rib import-rib webtraffic.inet.0
set routing-options interface-routes rib-group inet FBF-rib
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the device:

1. Configure the inbound interface and attach the **webFilter** firewall filter to it.

   ```
   [edit interfaces fe-0/0/0 unit 0 family inet]
   user@device# set filter input webFilter
   user@device# set address 198.51.100.1/24
   ```

2. Configure the outbound interfaces, one for Web traffic and the other for all other traffic.

   ```
   [edit interfaces]
   user@device# set fe-0/0/1 unit 0 family inet address 192.0.2.1/24
   user@device# set fe-0/0/2 unit 0 family inet address 203.0.113.1/24
   ```

3. Configure the firewall filter to pass Web traffic to the **webtraffic** routing instance and all other traffic to **203.0.113.1**.

   ```
   [edit firewall family inet filter webFilter]
   user@device# set term 1 from destination-port http
   user@device# set term 1 from destination-port https
   user@device# set term 1 then routing-instance webtraffic
   user@device# set term 2 then accept
   ```

4. Optional: Monitor traffic handling of the firewall filter by adding a counter>

   ```
   [edit interfaces fe-0/0/0 unit 0 family inet]
   user@device# set firewall family inet filter webFilter term 1 then count
     webtraffic-count
   ```

5. Create the **webtraffic** routing instance and configure it to forward Web traffic to **fe-0/0/1**.

   ```
   [edit routing-instances webtraffic]
   user@device# set routing-options static route 0.0.0.0/0 next-hop 192.0.2.2
   user@device# set instance-type forwarding
   ```

6. Create a route for non-Web traffic (the route is automatically installed in the **inet.0** routing table).

   ```
   [edit routing-options]
   user@device# set static route 0.0.0.0/0 next-hop 203.0.113.2
   ```

7. Create a RIB group called **FBF-rib**, and configure it so **inet.0** shares interface routes with **webtraffic.inet.0**.

   ```
   [edit routing-options]
   user@device# set rib-groups FBF-rib import-rib inet.0
   user@device# set rib-groups FBF-rib import-rib webtraffic.inet.0
   user@device# set interface-routes rib-group inet FBF-rib
   ```

Results From configuration mode, confirm your configuration by entering the **show firewall**, **show routing-instances**, **show routing-options**, and **show interfaces**, commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

If you are done configuring the device, enter **commit** from configuration mode.

```
user@device# show interfaces fe-0/0/0
unit 0 {
  family inet {
    filter {
      input webFilter;
    }
    address 198.51.100.1/24;
  }
}
user@device# show interfaces fe-0/0/1
unit 0 {
  family inet {
    address 192.0.2.1/24;
  }
}
user@device# show interfaces fe-0/0/2
unit 0 {
  family inet {
    address 203.0.113.1/24;
```



```
  }
}
user@device# show firewall
family inet {
  filter webFilter {
    term 1 {
      from {
        destination-port [ http https ];
      }
      then {
        routing-instance webtraffic;
      }
    }
    term 2 {
      then accept;
    }
  }
}

user@device# show routing-options
interface-routes {
  rib-group inet FBF-rib;
}
static {
  route 0.0.0.0/0 next-hop 203.0.113.2;
}
rib-groups {
  FBF-rib {
    import-rib [ inet.0 webtraffic.inet.0 ];
  }
}

user@device# show routing-instances
webtraffic {
  instance-type forwarding;
  routing-options {
    static {
      route 0.0.0.0/0 next-hop 192.0.2.2;
    }
  }
}
```

Related Documentation

- Understanding Filter-Based Forwarding to a Specific Outgoing Interface or Destination IP Address on page 952
- Example: Configuring Filter-Based Forwarding to a Specific Destination IP Address on page 958
- Example: Configuring Filter-Based Forwarding to a Specific Outgoing Interface on page 953
- *Configuring Filter-Based Forwarding*
- *Understanding Filter-Based Forwarding*
- *Using Filter-Based Forwarding to Select Traffic to Be Secured*

- Example: Configuring Filter-Based Forwarding on the Source Address on page 942
- Understanding RIB Groups

Example: Configuring Filter-Based Forwarding on Logical Systems

This example shows how to configure filter-based forwarding within a logical system. The filter classifies packets to determine their forwarding path within the ingress routing device.

Requirements

In this example, no special configuration beyond device initialization is required.

Overview

Filter-based forwarding is supported for IP version 4 (IPv4) and IP version 6 (IPv6).

Use filter-based forwarding for service provider selection when customers have Internet connectivity provided by different ISPs yet share a common access layer. When a shared media (such as a cable modem) is used, a mechanism on the common access layer looks at Layer 2 or Layer 3 addresses and distinguishes between customers. You can use filter-based forwarding when the common access layer is implemented using a combination of Layer 2 switches and a single router.

With filter-based forwarding, all packets received on an interface are considered. Each packet passes through a filter that has match conditions. If the match conditions are met for a filter and you have created a routing instance, filter-based forwarding is applied to a packet. The packet is forwarded based on the next hop specified in the routing instance. For static routes, the next hop can be a specific LSP.

NOTE: **Source-class usage filter matching and unicast reverse-path forwarding checks are not supported on an interface configured with filter-based forwarding (FBF).**

To configure filter-based forwarding, perform the following tasks:

- Create a match filter on an ingress router or switch. To specify a match filter, include the **filter *filter-name*** statement at the **[edit firewall]** hierarchy level. A packet that passes through the filter is compared against a set of rules to classify it and to determine its membership in a set. Once classified, the packet is forwarded to a routing table specified in the accept action in the filter description language. The routing table then forwards the packet to the next hop that corresponds to the destination address entry in the table.

- Create routing instances that specify the routing table(s) to which a packet is forwarded, and the destination to which the packet is forwarded at the **[edit routing-instances]** or **[edit logical-systems *logical-system-name* routing-instances]** hierarchy level. For example:

```
[edit]
routing-instances {
  routing-table-name1 {
    instance-type forwarding;
    routing-options {
      static {
        route 0.0.0.0/0 nexthop 10.0.0.1;
      }
    }
  }
  routing-table-name2 {
    instance-type forwarding;
    routing-options {
      static {
        route 0.0.0.0/0 nexthop 10.0.0.2;
      }
    }
  }
}
```

- Create a routing table group that adds interface routes to the forwarding routing instances used in filter-based forwarding (FBF), as well as to the default routing instance **inet.0**. This part of the configuration resolves the routes installed in the routing instances to directly connected next hops on that interface. Create the routing table group at the **[edit routing-options]** or **[edit logical-systems *logical-system-name* routing-options]** hierarchy level.

NOTE: Specify **inet.0** as one of the routing instances that the interface routes are imported into. If the default instance **inet.0** is not specified, interface routes are not imported into the default routing instance.

This example shows a packet filter that directs customer traffic to a next-hop router in the domains, SP 1 or SP 2, based on the packet's source address.

If the packet has a source address assigned to an SP 1 customer, destination-based forwarding occurs using the sp1-route-table.inet.0 routing table. If the packet has a source address assigned to an SP 2 customer, destination-based forwarding occurs using the sp2-route-table.inet.0 routing table. If a packet does not match either of these conditions, the filter accepts the packet, and destination-based forwarding occurs using the standard inet.0 routing table.

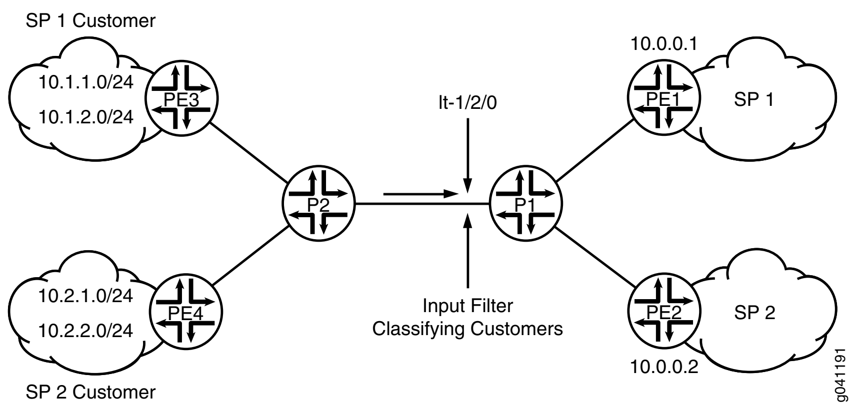
One way to make filter-based forwarding work within a logical system is to configure the firewall filter on the logical system that receives the packets. Another way is to configure the firewall filter on the main router and then reference the logical system in the firewall filter. This example uses the second approach. The specific routing instances are configured within the logical system. Because each routing instance has its own routing table, you have to reference the routing instances in the firewall filter, as well. The syntax looks as follows:

```
[edit firewall filter filter-name term term-name]
user@host# set then logical-system logical-system-name routing-instance
  routing-instance-name
```

Figure 51 on page 806 shows the topology used in this example.

On Logical System P1, an input filter classifies packets received from Logical System PE3 and Logical System PE4. The packets are routed based on the source addresses. Packets with source addresses in the 10.1.1.0/24 and 10.1.2.0/24 networks are routed to Logical System PE1. Packets with source addresses in the 10.2.1.0/24 and 10.2.2.0/24 networks are routed to Logical System PE2.

Figure 51: Logical Systems with Filter-Based Forwarding



To establish connectivity, OSPF is configured on all of the interfaces. For demonstration purposes, loopback interface addresses are configured on the routing devices to represent networks in the clouds.

The "CLI Quick Configuration" on page 806 section shows the entire configuration for all of the devices in the topology. The "Configuring the Routing Instances on the Logical System P1" on page 809 and "Configuring the Firewall Filter on the Main Router" on page 808 sections shows the step-by-step configuration of the ingress routing device, Logical System P1.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall filter classify-customers term sp1-customers from source-address 10.1.1.0/24
set firewall filter classify-customers term sp1-customers from source-address 10.1.2.0/24
set firewall filter classify-customers term sp1-customers then log
```

```
set firewall filter classify-customers term sp1-customers then logical-system P1
  routing-instance sp1-route-table
set firewall filter classify-customers term sp2-customers from source-address 10.2.1.0/24
set firewall filter classify-customers term sp2-customers from source-address 10.2.2.0/24
set firewall filter classify-customers term sp2-customers then log
set firewall filter classify-customers term sp2-customers then logical-system P1
  routing-instance sp2-route-table
set firewall filter classify-customers term default then accept
set logical-systems P1 interfaces lt-1/2/0 unit 10 encapsulation ethernet
set logical-systems P1 interfaces lt-1/2/0 unit 10 peer-unit 9
set logical-systems P1 interfaces lt-1/2/0 unit 10 family inet filter input classify-customers
set logical-systems P1 interfaces lt-1/2/0 unit 10 family inet address 172.16.0.10/30
set logical-systems P1 interfaces lt-1/2/0 unit 13 encapsulation ethernet
set logical-systems P1 interfaces lt-1/2/0 unit 13 peer-unit 14
set logical-systems P1 interfaces lt-1/2/0 unit 13 family inet address 172.16.0.13/30
set logical-systems P1 interfaces lt-1/2/0 unit 17 encapsulation ethernet
set logical-systems P1 interfaces lt-1/2/0 unit 17 peer-unit 18
set logical-systems P1 interfaces lt-1/2/0 unit 17 family inet address 172.16.0.17/30
set logical-systems P1 protocols ospf rib-group fbf-group
set logical-systems P1 protocols ospf area 0.0.0.0 interface all
set logical-systems P1 protocols ospf area 0.0.0.0 interface fxp0.0 disable
set logical-systems P1 routing-instances sp1-route-table instance-type forwarding
set logical-systems P1 routing-instances sp1-route-table routing-options static route
  0.0.0.0/0 next-hop 172.16.0.13
set logical-systems P1 routing-instances sp2-route-table instance-type forwarding
set logical-systems P1 routing-instances sp2-route-table routing-options static route
  0.0.0.0/0 next-hop 172.16.0.17
set logical-systems P1 routing-options rib-groups fbf-group import-rib inet.0
set logical-systems P1 routing-options rib-groups fbf-group import-rib
  sp1-route-table.inet.0
set logical-systems P1 routing-options rib-groups fbf-group import-rib
  sp2-route-table.inet.0
set logical-systems P2 interfaces lt-1/2/0 unit 2 encapsulation ethernet
set logical-systems P2 interfaces lt-1/2/0 unit 2 peer-unit 1
set logical-systems P2 interfaces lt-1/2/0 unit 2 family inet address 172.16.0.2/30
set logical-systems P2 interfaces lt-1/2/0 unit 6 encapsulation ethernet
set logical-systems P2 interfaces lt-1/2/0 unit 6 peer-unit 5
set logical-systems P2 interfaces lt-1/2/0 unit 6 family inet address 172.16.0.6/30
set logical-systems P2 interfaces lt-1/2/0 unit 9 encapsulation ethernet
set logical-systems P2 interfaces lt-1/2/0 unit 9 peer-unit 10
set logical-systems P2 interfaces lt-1/2/0 unit 9 family inet address 172.16.0.9/30
set logical-systems P2 protocols ospf area 0.0.0.0 interface all
set logical-systems P2 protocols ospf area 0.0.0.0 interface fxp0.0 disable
set logical-systems PE1 interfaces lt-1/2/0 unit 14 encapsulation ethernet
set logical-systems PE1 interfaces lt-1/2/0 unit 14 peer-unit 13
set logical-systems PE1 interfaces lt-1/2/0 unit 14 family inet address 172.16.0.14/30
set logical-systems PE1 interfaces lo0 unit 3 family inet address 172.16.1.1/32
set logical-systems PE1 protocols ospf area 0.0.0.0 interface all
set logical-systems PE1 protocols ospf area 0.0.0.0 interface fxp0.0 disable
set logical-systems PE2 interfaces lt-1/2/0 unit 18 encapsulation ethernet
set logical-systems PE2 interfaces lt-1/2/0 unit 18 peer-unit 17
set logical-systems PE2 interfaces lt-1/2/0 unit 18 family inet address 172.16.0.18/30
set logical-systems PE2 interfaces lo0 unit 4 family inet address 172.16.2.2/32
set logical-systems PE2 protocols ospf area 0.0.0.0 interface all
set logical-systems PE2 protocols ospf area 0.0.0.0 interface fxp0.0 disable
set logical-systems PE3 interfaces lt-1/2/0 unit 1 encapsulation ethernet
```

```
set logical-systems PE3 interfaces lt-1/2/0 unit 1 peer-unit 2
set logical-systems PE3 interfaces lt-1/2/0 unit 1 family inet address 172.16.0.1/30
set logical-systems PE3 interfaces lo0 unit 1 family inet address 10.1.1.1/32
set logical-systems PE3 interfaces lo0 unit 1 family inet address 10.1.2.1/32
set logical-systems PE3 protocols ospf area 0.0.0.0 interface all
set logical-systems PE3 protocols ospf area 0.0.0.0 interface fxp0.0 disable
set logical-systems PE4 interfaces lt-1/2/0 unit 5 encapsulation ethernet
set logical-systems PE4 interfaces lt-1/2/0 unit 5 peer-unit 6
set logical-systems PE4 interfaces lt-1/2/0 unit 5 family inet address 172.16.0.5/30
set logical-systems PE4 interfaces lo0 unit 2 family inet address 10.2.1.1/32
set logical-systems PE4 interfaces lo0 unit 2 family inet address 10.2.2.1/32
set logical-systems PE4 protocols ospf area 0.0.0.0 interface all
set logical-systems PE4 protocols ospf area 0.0.0.0 interface fxp0.0 disable
```

Configuring the Firewall Filter on the Main Router

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the firewall filter on the main router:

1. Configure the source addresses for SP1 customers.

   ```
   [edit firewall filter classify-customers term sp1-customers]
   user@host# set from source-address 10.1.1.0/24
   user@host# set from source-address 10.1.2.0/24
   ```

2. Configure the actions that are taken when packets are received with the specified source addresses.

   To track the action of the firewall filter, a log action is configured. The sp1-route-table.inet.0 routing table on Logical System P1 routes the packets.

   ```
   [edit firewall filter classify-customers term sp1-customers]
   user@host# set then log
   user@host# set then logical-system P1 routing-instance sp1-route-table
   ```

3. Configure the source addresses for SP2 customers.

   ```
   [edit firewall filter classify-customers term sp2-customers]
   user@host# set from source-address 10.2.1.0/24
   user@host# set from source-address 10.2.2.0/24
   ```

4. Configure the actions that are taken when packets are received with the specified source addresses.

   To track the action of the firewall filter, a log action is configured. The sp2-route-table.inet.0 routing table on Logical System P1 routes the packet.

   ```
   [edit firewall filter classify-customers term sp2-customers]
   user@host# set then log
   user@host# set then logical-system P1 routing-instance sp2-route-table
   ```


5. Configure the action to take when packets are received from any other source address.

   All of these packets are simply accepted and routed using the default IPv4 unicast routing table, inet.0.

```
[edit firewall filter classify-customers term default]
user@host# set then accept
```

Configuring the Routing Instances on the Logical System P1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the routing instances on a logical system:

1. Configure the interfaces on the logical system.

```
[edit logical-systems P1 interfaces lt-1/2/0]
user@host# set unit 10 encapsulation ethernet
user@host# set unit 10 peer-unit 9
user@host# set unit 10 family inet address 172.16.0.10/30

user@host# set unit 13 encapsulation ethernet
user@host# set unit 13 peer-unit 14
user@host# set unit 13 family inet address 172.16.0.13/30

user@host# set unit 17 encapsulation ethernet
user@host# set unit 17 peer-unit 18
user@host# set unit 17 family inet address 172.16.0.17/30
```

2. Assign the **classify-customers** firewall filter to router interface lt-1/2/0.10 as an input packet filter.

```
[edit logical-systems P1 interfaces lt-1/2/0]
user@host# set unit 10 family inet filter input classify-customers
```

3. Configure connectivity, using either a routing protocol or static routing.

   As a best practice, disable routing on the management interface.

```
[edit logical-systems P1 protocols ospf area 0.0.0.0]
user@host# set interface all
user@host# set interface fxp0.0 disable
```

4. Create the routing instances.

   These routing instances are referenced in the **classify-customers** firewall filter.

   The forwarding instance type provides support for filter-based forwarding, where interfaces are not associated with instances. All interfaces belong to the default instance, in this case Logical System P1.

   ```
   [edit logical-systems P1 routing-instances]
   user@host# set sp1-route-table instance-type forwarding

   user@host# set sp2-route-table instance-type forwarding
   ```

5. Resolve the routes installed in the routing instances to directly connected next hops.

   ```
   [edit logical-systems P1 routing-instances]
   user@host# set sp1-route-table routing-options static route 0.0.0.0/0 next-hop
     172.16.0.13

   user@host# set sp2-route-table routing-options static route 0.0.0.0/0 next-hop
     172.16.0.17
   ```

6. Group together the routing tables to form a routing table group.

   The first routing table, inet.0, is the primary routing table, and the additional routing tables are the secondary routing tables.

   The primary routing table determines the address family of the routing table group, in this case IPv4.

   ```
   [edit logical-systems P1 routing-options]
   user@host# set rib-groups fbf-group import-rib inet.0
   user@host# set rib-groups fbf-group import-rib sp1-route-table.inet.0
   user@host# set rib-groups fbf-group import-rib sp2-route-table.inet.0
   ```

7. Apply the routing table group to OSPF.

   This causes the OSPF routes to be installed into all the routing tables in the group.

   ```
   [edit logical-systems P1 protocols ospf]
   user@host# set rib-group fbf-group
   ```

8. If you are done configuring the device, commit the configuration.

   ```
   [edit]
   user@host# commit
   ```

Results

Confirm your configuration by issuing the **show firewall** and **show logical-systems P1** commands.

```
user@host# show firewall
```

```
filter classify-customers {
   term sp1-customers {
      from {
         source-address {
            10.1.1.0/24;
            10.1.2.0/24;
         }
      }
      then {
         log;
         logical-system P1 routing-instance sp1-route-table;
      }
   }
   term sp2-customers {
      from {
         source-address {
            10.2.1.0/24;
            10.2.2.0/24;
         }
      }
      then {
         log;
         logical-system P1 routing-instance sp2-route-table;
      }
   }
   term default {
      then accept;
   }
}
```

```
user@host# show logical-systems P1
interfaces {
   lt-1/2/0 {
      unit 10 {
         encapsulation ethernet;
         peer-unit 9;
         family inet {
            filter {
               input classify-customers;
            }
            address 172.16.0.10/30;
         }
      }
      unit 13 {
         encapsulation ethernet;
         peer-unit 14;
         family inet {
            address 172.16.0.13/30;
         }
      }
      unit 17 {
         encapsulation ethernet;
         peer-unit 18;
         family inet {
            address 172.16.0.17/30;
         }
```

```
        }
      }
    }
    protocols {
      ospf {
        rib-group fbf-group;
        area 0.0.0.0 {
          interface all;
          interface fxp0.0 {
            disable;
          }
        }
      }
    }
    routing-instances {
      sp1-route-table {
        instance-type forwarding;
        routing-options {
          static {
            route 0.0.0.0/0 next-hop 172.16.0.13;
          }
        }
      }
      sp2-route-table {
        instance-type forwarding;
        routing-options {
          static {
            route 0.0.0.0/0 next-hop 172.16.0.17;
          }
        }
      }
    }
    routing-options {
      rib-groups {
        fbf-group {
          import-rib [ inet.0 sp1-route-table.inet.0 sp2-route-table.inet.0 ];
        }
      }
    }
```

Verification

Confirm that the configuration is working properly.

Pinging with Specified Source Addresses

Purpose Send some ICMP packets across the network to test the firewall filter.

Action

1. Log in to Logical System PE3.

```
user@host> set cli logical-system PE3
Logical system: PE3
```

2. Run the **ping** command, pinging the lo0.3 interface on Logical System PE1.

The address configured on this interface is 172.16.1.1.

Specify the source address 10.1.2.1, which is the address configured on the lo0.1 interface on Logical System PE3.

```
user@host:PE3> ping 172.16.1.1 source 10.1.2.1
PING 172.16.1.1 (172.16.1.1): 56 data bytes
64 bytes from 172.16.1.1: icmp_seq=0 ttl=62 time=1.444 ms
64 bytes from 172.16.1.1: icmp_seq=1 ttl=62 time=2.094 ms
^C
--- 172.16.1.1 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.444/1.769/2.094/0.325 ms
```

3. Log in to Logical System PE4.

```
user@host:PE3> set cli logical-system PE4
Logical system: PE4
```

4. Run the **ping** command, pinging the lo0.4 interface on Logical System PE2.

   The address configured on this interface is 172.16.2.2.

   Specify the source address 10.2.1.1, which is the address configured on the lo0.2 interface on Logical System PE4.

```
user@host:PE4> ping 172.16.2.2 source 10.2.1.1
PING 172.16.2.2 (172.16.2.2): 56 data bytes
64 bytes from 172.16.2.2: icmp_seq=0 ttl=62 time=1.473 ms
64 bytes from 172.16.2.2: icmp_seq=1 ttl=62 time=1.407 ms
^C
--- 172.16.2.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.407/1.440/1.473/0.033 ms
```

Meaning Sending these pings activates the firewall filter actions.

Verifying the Firewall Filter

Purpose Make sure the firewall filter actions take effect.

Action

1. Log in to Logical System P1.

```
user@host> set cli logical-system P1
Logical system: P1
```

2. Run the **show firewall log** command on Logical System P1.

```
user@host:P1> show firewall log
Log :
Time      Filter    Action Interface     Protocol        Src Addr
Dest Addr
13:52:20  pfe       A      lt-1/2/0.10   ICMP            10.2.1.1
172.16.2.2
13:52:19  pfe       A      lt-1/2/0.10   ICMP            10.2.1.1
```

```
172.16.2.2
13:51:53  pfe       A      lt-1/2/0.10   ICMP             10.1.2.1
172.16.1.1
13:51:52  pfe       A      lt-1/2/0.10   ICMP             10.1.2.1
172.16.1.1
```

Related Documentation

- *Configuring Filter-Based Forwarding*
- *Copying and Redirecting Traffic with Port Mirroring and Filter-Based Forwarding*
- Example: Configuring Filter-Based Forwarding on the Source Address on page 942
- *Using Filter-Based Forwarding to Export Monitored Traffic to Multiple Destinations*
- Filter-Based Forwarding Overview on page 939

Example: Configuring a Stateless Firewall Filter to Protect a Logical System Against ICMP Floods

This example shows how to configure a stateless firewall filter that protects against ICMP denial-of-service attacks on a logical system.

Requirements

In this example, no special configuration beyond device initialization is required.

Overview

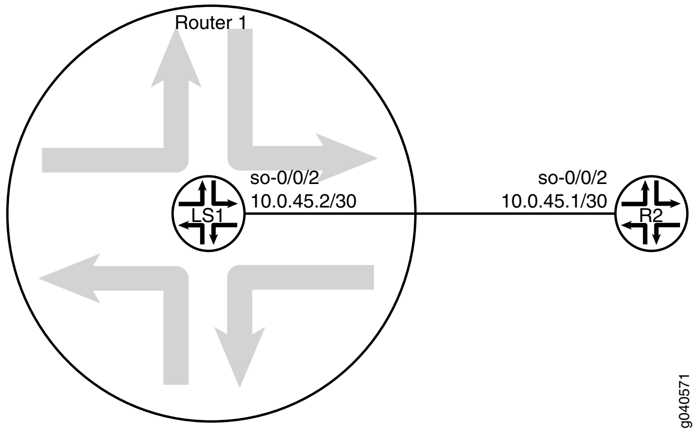
This example shows a stateless firewall filter called protect-RE that polices ICMP packets. The **icmp-policer** limits the traffic rate of the ICMP packets to 1,000,000 bps and the burst size to 15,000 bytes. Packets that exceed the traffic rate are discarded.

The policer is incorporated into the action of a filter term called **icmp-term**.

In this example, a ping is sent from a directly connected physical router to the interface configured on the logical system. The logical system accepts the ICMP packets if they are received at a rate of up to 1 Mbps (bandwidth-limit). The logical system drops all ICMP packets when this rate is exceeded. The **burst-size-limit** statement accepts traffic bursts up to 15 Kbps. If bursts exceed this limit, all packets are dropped. When the flow rate subsides, ICMP packets are again accepted.

Figure 52 on page 815 shows the topology used in this example.

Figure 52: Logical System with a Stateless Firewall



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

```
set logical-systems LS1 interfaces so-0/0/2 unit 0 family inet policer input icmp-policer
set logical-systems LS1 interfaces so-0/0/2 unit 0 family inet address 10.0.45.2/30
set logical-systems LS1 firewall family inet filter protect-RE term icmp-term from protocol
  icmp
set logical-systems LS1 firewall family inet filter protect-RE term icmp-term then policer
  icmp-policer
set logical-systems LS1 firewall family inet filter protect-RE term icmp-term then accept
set logical-systems LS1 firewall policer icmp-policer if-exceeding bandwidth-limit 1m
set logical-systems LS1 firewall policer icmp-policer if-exceeding burst-size-limit 15k
set logical-systems LS1 firewall policer icmp-policer then discard
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure an ICMP firewall filter on a logical system:

1. Configure the interface on the logical system.

```
[edit]
user@host# set logical-systems LS1 interfaces so-0/0/2 unit 0 family inet address
  10.0.45.2/30
```

2. Explicitly enable ICMP packets to be received on the interface.

```
[edit]
user@host# set logical-systems LS1 firewall family inet filter protect-RE term
  icmp-term from protocol icmp
user@host# set logical-systems LS1 firewall family inet filter protect-RE term
  icmp-term then accept
```


3. Create the policer.

   [edit]
   user@host# **set logical-systems LS1 firewall policer icmp-policer if-exceeding bandwidth-limit 1m**
   user@host# **set logical-systems LS1 firewall policer icmp-policer if-exceeding burst-size-limit 15k**
   user@host# **set logical-systems LS1 firewall policer icmp-policer then discard**

4. Apply the policer to a filter term.

   [edit]
   user@host# **set logical-systems LS1 firewall family inet filter protect-RE term icmp-term then policer icmp-policer**

5. Apply the policer to the logical system interface.

   [edit]
   user@host# **set logical-systems LS1 interfaces so-0/0/2 unit 0 family inet policer input icmp-policer**

6. If you are done configuring the device, commit the configuration.

   [edit]
   user@host# **commit**

Results

Confirm your configuration by issuing the **show logical-systems LS1** command.

```
user@host# show logical-systems LS1
interfaces {
    so-0/0/2 {
        unit 0 {
            family inet {
                policer {
                    input icmp-policer;
                }
                address 10.0.45.2/30;
            }
        }
    }
}
firewall {
    family inet {
        filter protect-RE {
            term icmp-term {
                from {
                    protocol icmp;
                }
                then {
                    policer icmp-policer;
                    accept;
```



```
            }
          }
        }
      }
      policer icmp-policer {
        if-exceeding {
          bandwidth-limit 1m;
          burst-size-limit 15k;
        }
        then discard;
      }
    }
```

Verification

Confirm that the configuration is working properly.

Verifying That Ping Works Unless the Limits Are Exceeded

Purpose Make sure that the logical system interface is protected against ICMP-based DoS attacks.

Action Log in to a system that has connectivity to the logical system and run the **ping** command.

```
user@R2> ping 10.0.45.2
PING 10.0.45.2 (10.0.45.2): 56 data bytes
64 bytes from 10.0.45.2: icmp_seq=0 ttl=64 time=1.316 ms
64 bytes from 10.0.45.2: icmp_seq=1 ttl=64 time=1.277 ms
64 bytes from 10.0.45.2: icmp_seq=2 ttl=64 time=1.269 ms

user@R2> ping 10.0.45.2 size 20000
PING 10.0.45.2 (10.0.45.2): 20000 data bytes
^C
--- 10.0.45.2 ping statistics ---
4 packets transmitted, 0 packets received, 100% packet loss
```

Meaning When you send a normal ping, the packet is accepted. When you send a ping packet that exceeds the filter limit, the packet is discarded.

Related Documentation

- *Example: Creating an Interface on a Logical System*

Unsupported Firewall Filter Statements for Logical Systems

Table 51 on page 818 shows statements that are supported at the **[edit firewall]** hierarchy level but not at the **[edit logical-systems *logical-system-name* firewall]** hierarchy level.

Table 51: Unsupported Firewall Statements for Logical Systems

Statement	Example	Description

accounting-profile	[edit]
	logical-systems {
	ls1 {
	firewall {
	family inet {
	filter myfilter {
	accounting-profile fw-profile;
	...
	term accept-all {
	then {
	count counter1;
	accept;
	}
	}
	}
	}
	}
	}
	}
		In this example, the **accounting-profile** statement is not allowed because the accounting profile **fw-profile** is configured under the **[edit accounting-options]** hierarchy.

hierarchical-policer	[edit]
	logical-systems {
	lr1 {
	firewall {
	hierarchical-policer {
	...
	}
	}
	}
	}
		In this example, the **hierarchical policer** statement requires a class-of-service configuration, which is not supported under logical systems.

load-balance-group	[edit]
	logical-systems {
	ls1 {
	firewall {
	load-balance-group lb-group {
	next-hop-group nh-group;
	}
	}
	}
	}
		This configuration is not allowed because the **next-hop-group nh-group** statement must be configured at the **[edit forwarding-options next-hop-group]** hierarchy level—outside the **[edit logical-systems *logical-system-name* firewall]** hierarchy.
		
		Currently, the **forwarding-options dhcp-relay** statement is the only forwarding option supported for logical systems.

Table 51: Unsupported Firewall Statements for Logical Systems *(continued)*

Statement	Example	Description

virtual-channel	`[edit]`
	`logical-systems {`
	`  ls1 {`
	`    firewall {`
	`      family inet {`
	`        filter foo {`
	`          term one {`
	`            from {`
	`              source-address 10.1.0.0/16;`
	`            }`
	`            then {`
	`              virtual-channel sammy;`
	`            }`
	`          }`
	`        }`
	`      }`
	`    }`
	`  }`
	`}`
		This configuration is not allowed because the virtual channel **sammy** refers to an object defined at the **[edit class-of-service]** hierarchy level, and class of service is not supported for logical systems.
		
		NOTE:
		
		The **virtual-channel** statement is supported for J Series devices only, provided the firewall filter is configured outside of a logical-system.

Related Documentation

- Firewall Filters in Logical Systems Overview on page 787
- Guidelines for Configuring and Applying Firewall Filters in Logical Systems on page 788
- Unsupported Actions for Firewall Filters in Logical Systems on page 819
- “*Introduction to Logical Systems*” in the *Logical Systems Feature Guide*
- “*Logical Systems Operations and Restrictions*” in the *Logical Systems Feature Guide*

Unsupported Actions for Firewall Filters in Logical Systems

Table 52 on page 819 describes the firewall filter actions that are supported at the **[edit firewall]** hierarchy level, but not supported at the **[edit logical-systems *logical-system-name* firewall]** hierarchy level.

Table 52: Unsupported Actions for Firewall Filters in Logical Systems

Firewall Filter Action	Example	Description
Terminating Actions Not Supported in a Logical System		

Table 52: Unsupported Actions for Firewall Filters in Logical Systems *(continued)*

Firewall Filter Action	Example	Description

logical-system	[edit]
	logical-systems {
	  ls1 {
	    firewall {
	      family inet {
	        filter foo {
	          term one {
	            from {
	              source-address 10.1.0.0/16;
	            }
	            then {
	              logical-system fred;
	            }
	          }
	        }
	      }
	    }
	  }
	}
		Because the **logical-system** action refers to **fred**—a logical system defined outside the local logical system—, this action is not supported.

Nonterminating Actions Not Supported in a Logical System		

ipsec-sa	[edit]
	logical-systems {
	  ls1 {
	    firewall {
	      family inet {
	        filter foo {
	          term one {
	            from {
	              source-address 10.1.0.0/16;
	            }
	            then {
	              ipsec-sa barney;
	            }
	          }
	        }
	      }
	    }
	  }
	}
		Because the **ipsec-sa** action modifier references **barney**—a security association defined outside the local logical system—this action is not supported.

Table 52: Unsupported Actions for Firewall Filters in Logical Systems *(continued)*

Firewall Filter Action	Example	Description

next-hop-group	`[edit]`
	`logical-systems {`
	`  ls1 {`
	`    firewall {`
	`      family inet {`
	`        filter foo {`
	`          term one {`
	`            from {`
	`              source-address 10.1.0.0/16;`
	`            }`
	`            then {`
	`              next-hop-group fred;`
	`            }`
	`          }`
	`        }`
	`      }`
	`    }`
	`  }`
	`}`
		Because the **next-hop-group** action refers to **fred**—an object defined at the **[edit forwarding-options next-hop-group]** hierarchy level—this action is not supported.

port-mirror	`[edit]`
	`logical-systems {`
	`  ls1 {`
	`    firewall {`
	`      family inet {`
	`        filter foo {`
	`          term one {`
	`            from {`
	`              source-address 10.1.0.0/16;`
	`            }`
	`            then {`
	`              port-mirror;`
	`            }`
	`          }`
	`        }`
	`      }`
	`    }`
	`  }`
	`}`
		Because the **port-mirror** action relies on a configuration defined at the **[edit forwarding-options port-mirroring]** hierarchy level, this action is not supported.

Table 52: Unsupported Actions for Firewall Filters in Logical Systems *(continued)*

Firewall Filter Action	Example	Description

sample	<pre>[edit]
	logical-systems {
	  ls1 {
	    firewall {
	      family inet {
	        filter foo {
	          term one {
	            from {
	              source-address 10.1.0.0/16;
	            }
	            then {
	              sample;
	            }
	          }
	        }
	      }
	    }
	  }
	}</pre>
		In this example, the **sample** action depends on the sampling configuration defined under the **[edit forwarding-options]** hierarchy. Therefore, the **sample** action is not supported.

syslog	<pre>[edit]
	logical-systems {
	  ls1 {
	    firewall {
	      family inet {
	        filter icmp-syslog {
	          term icmp-match {
	            from {
	              address {
	                192.168.207.222/32;
	              }
	              protocol icmp;
	            }
	            then {
	              count packets;
	              syslog;
	              accept;
	            }
	          }
	          term default {
	            then accept;
	          }
	        }
	      }
	    }
	  }
	}</pre>
		In this example, there must be at least one system log (**system syslog file *filename***) with the **firewall** facility enabled for the **icmp-syslog** filter's logs to be stored.
		
		Because this firewall configuration relies on a configuration outside the logical system, the **syslog** action modifier is not supported.

Related Documentation

- Firewall Filters in Logical Systems Overview on page 787
- Guidelines for Configuring and Applying Firewall Filters in Logical Systems on page 788
- Unsupported Firewall Filter Statements for Logical Systems on page 817
- *Introduction to Logical Systems*

- *Logical Systems Operations and Restrictions*

CHAPTER 19

Configuring Firewall Filter Accounting and Logging

Accounting for Firewall Filters Overview

Juniper Networks devices can collect various kinds of data about traffic passing through the device. You can set up one or more accounting profiles that specify some common characteristics of this data, including the following:

- Fields used in the accounting records.
- Number of files that the routing platform retains before discarding, and the number of bytes per file.
- Polling period that the system uses to record the data

There are several types of accounting profiles: interface, firewall filter, source class and destination class usage, and Routing Engine. If you apply the same profile name to both a firewall filter and an interface, it causes an error.

Related Documentation

- Example: Configuring Statistics Collection for a Firewall Filter on page 829

System Logging Overview

The Junos OS generates system log messages (also called *syslog messages*) to record *system events* that occur on the device. Events consist of routine operations, failure and error conditions, and critical conditions that might require urgent resolution. This system logging utility is similar to the UNIX **syslogd** utility.

Each Junos OS system log message belongs to a message category, called a *facility*, that reflects the hardware- or software-based source of the triggering event. A group of messages belonging to the same facility are either generated by the same software process or concern a similar hardware condition or user activity (such as authentication attempts). Each system log message is also preassigned a *severity*, which indicates how seriously the triggering event affects router (or switch) functions. Together, the facility and severity of an event are known as the message *priority*. The content of a syslog message identifies the Junos OS *process* that generates the message and briefly describes the operation or error that occurred.

By default, syslog messages that have a severity of **info** or more serious are written to the main system log file **messages** in the **/var/log** directory of the local Routing Engine. To configure global settings and facility-specific settings that override these default values, you can include statements at the **[edit system syslog]** hierarchy level.

For all syslog facilities or for a specified facility, you can configure the syslog message utility to redirect messages of a specified severity to a specified file instead of to the main system log file. You can also configure the syslog message utility to write syslog messages of a specified severity, for all syslog facilities or for a specified facility, to additional destinations. In addition to writing syslog messages to a log file, you can write syslog messages to the terminal sessions of any logged-in users, to the router (or switch) console, or to a remote host or the other Routing Engine.

At the global level—for all system logging messages, regardless of facility, severity, or destination—you can override the default values for file-archiving properties and the default timestamp format.

Related Documentation

- System Logging of Events Generated for the Firewall Facility on page 826
- Logging of Packet Headers Evaluated by a Firewall Filter Term on page 828
- Example: Configuring Logging for a Firewall Filter Term on page 834

System Logging of Events Generated for the Firewall Facility

System log messages generated for firewall filter actions belong to the **firewall** facility. Just as you can for any other Junos OS system logging facility, you can direct **firewall** facility syslog messages to one or more specific destinations: to a specified file, to the terminal session of one or more logged in users (or to all users), to the router (or switch) console, or to a remote host or the other Routing Engine on the router (or switch).

When you configure a syslog message destination for **firewall** facility syslog messages, you include a statement at the **[edit system syslog]** hierarchy level, and you specify the **firewall** facility name together with a severity level. Messages from the **firewall** that are rated at the specified level or more severe are logged to the destination.

System log messages with the **DFWD_** prefix are generated by the firewall process (**dfwd**), which manages compilation and downloading of Junos OS firewall filters. System log messages with the **PFE_FW_** prefix are messages about firewall filters, generated by the

Packet Forwarding Engine controller, which manages packet forwarding functions. For more information, see the System Log Explorer.

Table 53 on page 827 lists the system log destinations you can configure for the **firewall** facility.

Table 53: Syslog Message Destinations for the Firewall Facility

Destination	Description	Configuration Statements Under [edit system syslog]

File	Configuring this option keeps the **firewall** syslog messages out of the main system log file.
	
	To include priority and facility with messages written to the file, include the **explicit-priority** statement.
	
	To override the default standard message format, which is based on a UNIX system log format, include the **structured-data** statement. When the **structured-data** statement is included, other statements that specify the format for messages written to the file are ignored (the **explicit-priority** statement at the **[edit system syslog file *filename*]** hierarchy level and the **time-format** statement at the **[edit system syslog]** hierarchy level).
		file *filename* {
		  firewall *severity*;
		  allow-duplicates;
		  archive *archive-optios*;
		  explicit-priority;
		  structured-data;
		}
		allow-duplicates;
		archive *archive-options*;
		time-format (*option*);

Terminal session	Configuring this option causes a copy of the **firewall** syslog messages to be written to the specified terminal sessions. Specify one or more user names, or specify ***** for all logged in users.	**user (*username* | *) {** **firewall *severity*;** **}** **time-format (*option*);**
Router (or switch) console	Configuring this option causes a copy of the **firewall** syslog messages to be written to the router (or switch) console.	**console {** **firewall *severity*;** **}** **time-format (*option*);**

Remote host or the other Routing Engine	Configuring this option causes a copy of the **firewall** syslog messages to be written to the specified remote host or to the other Routing Engine.
	
	To override the default alternative facility for forwarding **firewall** syslog messages to a remote machine (**local3**), include the **facility-override firewall** statement.
	
	To include priority and facility with messages written to the file, include the **explicit-priority** statement.
		host (*hostname* | other-routing-engine) {
		  firewall *severity*;
		  allow-duplicates;
		  archive *archive-optios*;
		  facility-override firewall;
		  explicit-priority;
		}
		allow-duplicates; # All destinations
		archive *archive-options*;
		time-format (*option*);

By default, the timestamp recorded in a standard-format system log message specifies the month, date, hour, minute, and second when the message was logged, as in the example:

```
Sep 07 08:00:10
```

To include the year, the millisecond, or both in the timestamp for all system logging messages, regardless of the facility, include one of the following statement at the **[edit system syslog]** hierarchy level:

- **time-format year**;
- **time-format millisecond**;
- **time-format year millisecond**;

The following example illustrates the format for a timestamp that includes both the millisecond (**401**) and the year (**2010**):

```
Sep 07 08:00:10.401.2010
```

Related Documentation

- System Logging Overview on page 825
- Logging of Packet Headers Evaluated by a Firewall Filter Term on page 828
- Example: Configuring Logging for a Firewall Filter Term on page 834
- *Junos OS System Logging Facilities and Message Severity Levels*
- *Junos OS System Log Configuration Hierarchy*
- *Junos OS Default System Log Settings*
- *Logging Messages in Structured-Data Format*
- *Including the Year or Millisecond in Timestamps*
- *Changing the Alternative Facility Name for System Log Messages Directed to a Remote Destination*
- *Alternate Facilities for System Log Messages Directed to a Remote Destination*

Logging of Packet Headers Evaluated by a Firewall Filter Term

Built in to the stateless firewall filtering software is the capability to log packet-header information for the packets evaluated by a stateless firewall filter term. You can write the packet header information to the system log file on the local Routing Engine or to a firewall filter buffer in the Packet Forwarding Engine. Logging of packet headers evaluated by firewall filters is supported for standard stateless firewall filters for IPv4 or IPv6 traffic only. Service filters and simple filters do not support logging of packet headers.

Table 54 on page 829 lists the packet-header logs you can configure for a firewall filter action.

Table 54: Packet-Header Logs for Stateless Firewall Filter Terms

Log	Description	Configuration Statements

Syslog message destinations configured for the firewall facility	Configure this option by using the **syslog** nonterminating action.
	
	NOTE: Packet header information is interspersed with event messages.
	
	To list log files, enter the **show log** operational mode command without command options.
	
	To display log file contents for a specific file in the **/var/log** directory on the local Routing Engine, enter the **show log** ***filename*** operational mode command or the **file show /var/log/*filename*** operational mode command.
	
	To clear log file contents, enter the **clear log *filename* <all>** operational mode command. If you include the **all** option, the specified log file is truncated, all archived versions of the log file are deleted.
		firewall {
		family {
		filter *filter-name* {
		from {
		match-conditions;
		}
		then {
		...
		syslog;
		terminating-action;
		}
		}
		}
		}

Buffer in the Packet Forwarding Engine	Configure this option by using the **log** nonterminating action.
	
	NOTE: Restarting the router (or switch) causes the contents of this buffer to be cleared.
	
	To display the local log entries for firewall filters, enter the **show firewall log** operational mode command.
		firewall {
		family {
		filter *filter-name* {
		from {
		match-conditions;
		}
		then {
		...
		log;
		terminating-action;
		}
		}
		}
		}

Related Documentation

- System Logging Overview on page 825
- System Logging of Events Generated for the Firewall Facility on page 826
- Example: Configuring Logging for a Firewall Filter Term on page 834

Example: Configuring Statistics Collection for a Firewall Filter

This example shows how to configure and apply a firewall filter that collects data according to parameters specified in an associated accounting profile.

Requirements

Firewall filter accounting profiles are supported for all traffic types except **family any**.

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you create a firewall filter accounting profile and apply it to a firewall filter. The accounting profile specifies how frequently to collect packet and byte count statistics and the name of the file to which the statistics are written. The profile also specifies that statistics are to be collected for three firewall filter counters.

Topology

The firewall filter accounting profile **filter_acctg_profile** specifies that statistics are collected every 60 minutes, and the statistics are written to the file **/var/log/ff_accounting_file**. Statistics are collected for counters named **counter1**, **counter2**, and **counter3**.

The IPv4 firewall filter named **my_firewall_filter** increments a counter for each of three filter terms. The filter is applied to logical interface **ge-0/0/1.0**.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configure an Accounting Profile on page 831
- Configure a Firewall Filter That References the Accounting Profile on page 831
- Apply the Firewall Filter to an Interface on page 832
- Confirm Your Candidate Configuration on page 832
- Clear the Counters and Commit Your Candidate Configuration on page 834

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set accounting-options filter-profile filter_acctg_profile file ff_accounting_file
set accounting-options filter-profile filter_acctg_profile interval 60
set accounting-options filter-profile filter_acctg_profile counters counter1
set accounting-options filter-profile filter_acctg_profile counters counter2
set accounting-options filter-profile filter_acctg_profile counters counter3
set firewall family inet filter my_firewall_filter accounting-profile filter_acctg_profile
set firewall family inet filter my_firewall_filter term term1 from protocol ospf
set firewall family inet filter my_firewall_filter term term1 then count counter1
set firewall family inet filter my_firewall_filter term term1 then discard
set firewall family inet filter my_firewall_filter term term2 from source-address
  10.108.0.0/16
set firewall family inet filter my_firewall_filter term term2 then count counter2
set firewall family inet filter my_firewall_filter term term2 then discard
set firewall family inet filter my_firewall_filter term accept-all then count counter3
set firewall family inet filter my_firewall_filter term accept-all then accept
```

set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
set interfaces ge-0/0/1 unit 0 family inet filter input my_firewall_filter

Configure an Accounting Profile

Step-by-Step Procedure

To configure an accounting profile:

1. Create the accounting profile **filter_acctg_profile**.

   [edit]
   user@host# **edit accounting-options filter-profile filter_acctg_profile**

2. Configure the accounting profile to filter and collect packet and byte count statistics every 60 minutes and write them to the **/var/log/ff_accounting_file** file.

   [edit accounting-options filter-profile filter_acctg_profile]
   user@host# **set file ff_accounting_file**
   user@host# **set interval 60**

3. Configure the accounting profile to collect filter profile statistics (packet and byte counts) for three counters.

   [edit accounting-options filter-profile filter_acctg_profile]
   user@host# **set counters counter1**
   user@host# **set counters counter2**
   user@host# **set counters counter3**

Configure a Firewall Filter That References the Accounting Profile

Step-by-Step Procedure

To configure a firewall filter that references the accounting profile:

1. Create the firewall filter **my_firewall_filter**.

   [edit]
   user@host# **edit firewall family inet filter my_firewall_filter**

2. Apply the filter-accounting profile **filter_acctg_profile** to the firewall filter.

   [edit firewall family inet filter my_firewall_filter]
   user@host# **set** accounting-profile **filter_acctg_profile**

3. Configure the first filter term and counter.

   [edit firewall family inet filter my_firewall_filter]
   user@host# **set term term1 from protocol ospf**
   user@host# **set term term1 then count counter1**
   user@host# **set term term1 then discard**

4. Configure the second filter term and counter.

   [edit firewall family inet filter my_firewall_filter]
   user@host# **set term term2 from source-address 10.108.0.0/16**

```
user@host# set term term2 then count counter2
user@host# set term term2 then discard
```

5. Configure the third filter term and counter.

```
[edit firewall family inet filter my_firewall_filter]
user@host# set term accept-all then count counter3
user@host# set term accept-all then accept
```

Apply the Firewall Filter to an Interface

Step-by-Step Procedure

To apply the firewall filter to a logical interface:

1. Configure the logical interface to which you will apply the firewall filter.

```
[edit]
user@host# edit interfaces ge-0/0/1 unit 0 family inet
```

2. Configure the interface address for the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set address 10.1.2.3/30
```

3. Apply the firewall filter to the logical interface.

```
[edit interfaces ge-0/0/1 unit 0 family inet]
user@host# set filter input my_firewall_filter
```

Confirm Your Candidate Configuration

Step-by-Step Procedure

To confirm your candidate configuration:

1. Confirm the configuration of the accounting profile by entering the **show accounting-options** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show accounting-options
filter-profile filter_acctg_profile {
  file ff_accounting_file;
  interval 60;
  counters {
    counter1;
    counter2;
    counter3;
  }
}
```


2. Confirm the configuration of the firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
   filter my_firewall_filter {
      accounting-profile filter_acctg_profile;
      term term1 {
         from {
            protocol ospf;
         }
         then {
            count counter1;
            discard;
         }
      }
      term term2 {
         from {
            source-address {
               10.108.0.0/16;
            }
         }
         then {
            count counter2;
            discard;
         }
      }
      term accept-all {
         then {
            count counter3;
            accept;
         }
      }
   }
}
```

3. Confirm the configuration of the interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
ge-0/0/1 {
   unit 0 {
      family inet {
         filter {
            input my_firewall_filter;
         }
         address 10.1.2.3/30;
      }
   }
}
```


Clear the Counters and Commit Your Candidate Configuration

Step-by-Step Procedure

To clear the counters and commit your candidate configuration:

1. From operational command mode, use the **clear firewall all** command to clear the statistics for all firewall filters.

   To clear only the counters incremented in this example, include the name of the firewall filter.

   ```
   [edit]
   user@host> clear firewall filter my_firewall_filter
   ```

2. Commit your candidate configuration.

   ```
   [edit]
   user@host# commit
   ```

Verification

To verify that the filter is applied to the logical interface, run the **show interfaces** command with the **detail** or **extensive** output level.

To verify that the three counters are collected separately, run the **show firewall filter my_firewall_filter** command.

```
user@host> show firewall filter my_firewall_filter

Filter: my_firewall_filter
Counters:
Name                                                  Bytes                Packets
counter1                                                  0                      0
counter2                                                  0                      0
counter3                                                  0                      0
```

Related Documentation

- Accounting for Firewall Filters Overview on page 825

Example: Configuring Logging for a Firewall Filter Term

This example shows how to configure a firewall filter to log packet headers.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you use a firewall filter that logs and counts ICMP packets that have **192.168.207.222** as either their source or destination.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configure the Syslog Messages File for the Firewall Facility on page 835
- Configure the Firewall Filter on page 836
- Apply the Firewall Filter to a Logical Interface on page 836
- Confirm and Commit Your Candidate Configuration on page 837

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set system syslog file messages_firewall_any firewall any
set system syslog file messages_firewall_any archive no-world-readable
set firewall family inet filter icmp_syslog term icmp_match from address
  192.168.207.222/32
set firewall family inet filter icmp_syslog term icmp_match from protocol icmp
set firewall family inet filter icmp_syslog term icmp_match then count packets
set firewall family inet filter icmp_syslog term icmp_match then syslog
set firewall family inet filter icmp_syslog term icmp_match then accept
set firewall family inet filter icmp_syslog term default_term then accept
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
set interfaces ge-0/0/1 unit 0 family inet filter input icmp_syslog
```

Configure the Syslog Messages File for the Firewall Facility

Step-by-Step Procedure

To configure a syslog messages file for the **firewall** facility:

1. Configure a messages file for all syslog messages generated for the **firewall** facility.

   ```
   user@host# set system syslog file messages_firewall_any firewall any
   ```

2. Restrict permission to the archived **firewall** facility syslog files to the root user and users who have the Junos OS maintenance permission.

   ```
   user@host# set system syslog file messages_firewall_any archive no-world-readable
   ```


Configure the Firewall Filter

Step-by-Step Procedure

To configure the firewall filter **icmp_syslog** that logs and counts ICMP packets that have **192.168.207.222** as either their source or destination:

1. Create the firewall filter **icmp_syslog**.

   ```
   [edit]
   user@host# edit firewall family inet filter icmp_syslog
   ```

2. Configure matching on the ICMP protocol and an address.

   ```
   [edit firewall family inet filter icmp_syslog]
   user@host# set term icmp_match from address 192.168.207.222/32
   user@host# set term icmp_match from protocol icmp
   ```

3. Count, log,, and accept matching packets.

   ```
   [edit firewall family inet filter icmp_syslog]
   user@host# set term icmp_match then count packets
   user@host# set term icmp_match then syslog
   user@host# set term icmp_match then accept
   ```

4. Accept all other packets.

   ```
   [edit firewall family inet filter icmp_syslog]
   user@host# set term default_term then accept
   ```

Apply the Firewall Filter to a Logical Interface

Step-by-Step Procedure

To apply the firewall filter to a logical interface:

1. Configure the logical interface to which you will apply the firewall filter.

   ```
   [edit]
   user@host# edit interfaces ge-0/0/1 unit 0 family inet
   ```

2. Configure the interface address for the logical interface.

   ```
   [edit interfaces ge-0/0/1 unit 0 family inet]
   user@host# set address 10.1.2.3/30
   ```

3. Apply the firewall filter to the logical interface.

   ```
   [edit interfaces ge-0/0/1 unit 0 family inet]
   user@host# set filter input icmp_syslog
   ```


Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the syslog message file for the **firewall** facility by entering the **show system** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show system
syslog {
  file messages_firewall_any {
    firewall any;
    archive no-world-readable;
  }
}
```

2. Confirm the configuration of the firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter icmp_syslog {
    term icmp_match {
      from {
        address {
          192.168.207.222/32;
        }
        protocol icmp;
      }
      then {
        count packets;
        syslog;
        accept;
      }
    }
    term default_term {
      then accept;
    }
  }
}
```

3. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
ge-0/0/1 {
  unit 0 {
    family inet {
```

```
                filter {
                  input icmp_syslog;
                }
                address 10.1.2.3/30;
              }
            }
          }
```

4. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the **show log filter** command:

```
user@host> show log messages_firewall_any
Mar 20 08:03:11 hostname feb FW: so-0/1/0.0   A icmp 192.168.207.222

192.168.207.223      0      0 (1 packets)
```

This output file contains the following fields:

- **Date and Time**—Date and time at which the packet was received (not shown in the default).
- Filter action:
  - **A**—Accept (or next term)
  - **D**—Discard
  - **R**—Reject
- **Protocol**—Packet's protocol name or number.
- **Source address**—Source IP address in the packet.
- **Destination address**—Destination IP address in the packet.

NOTE: **If the protocol is ICMP, the ICMP type and code are displayed. For all other protocols, the source and destination ports are displayed.**

The last two fields (both zero) are the source and destination TCP/UDP ports, respectively, and are shown for TCP or UDP packets only. This log message indicates that only one packet for this match has been detected in about a 1-second interval. If packets arrive faster, the system log function compresses the information so that less output is generated, and displays an output similar to the following:

```
user@host> show log messages_firewall_any
Mar 20 08:08:45 hostname feb FW: so-0/1/0.0   A icmp 192.168.207.222
```



```
192.168.207.223     0     0 (515 packets)
```

Related Documentation

- System Logging Overview on page 825
- Logging of Packet Headers Evaluated by a Firewall Filter Term on page 828
- System log messages with the **DFWD_** prefix, described in the System Log Explorer
- System log messages with the **PFE_FW_*** prefix, described in the System Log Explorer

CHAPTER 20

Attaching Multiple Firewall Filters to a Single Interface

Understanding Multiple Firewall Filters in a Nested Configuration

This topic covers the following information:

The Challenge: Simplify Large-Scale Firewall Filter Administration

Typically, you apply a single firewall filter to an interface in the input or output direction or both. This approach might not be practical, however, when you have a router (or switch) configured with many, even hundreds of interfaces. In an environment of this scale, you want the flexibility of being able to modify filtering terms common to multiple interfaces without having to reconfigure the filter of every affected interface.

In general, the solution is to apply an effectively "chained" structure of multiple stateless firewall filters to a single interface. You partition your filtering terms into multiple firewall filters configured so that you can apply a unique filter to each router (or switch) interface but also apply common filters to multiple router (or switch) interfaces as required. The Junos OS policy framework provides two options for managing the application of multiple separate firewall filters to individual router (or switch) interfaces. One option is to apply multiple filters as a single input list or output list. The other option is to reference a stateless firewall filter from within the term of another stateless firewall filter.

A Solution: Configure Nested References to Firewall Filters

The most structured way to avoid configuring duplicate filtering terms common to multiple firewall filters is to configure multiple firewall filters so that each filter includes the shared filtering terms by *referencing* a separate filter that contains the common filtering terms. The Junos OS uses the filter terms—in the order in which they appear in the filter definition—to evaluate packets that transit the interface. If you need to modify filtering terms shared across multiple interfaces, you only need to modify one firewall filter.

NOTE: Similar to the alternative approach (applying a list of firewall filters), configuring a nested firewall filter combines multiple firewall filters into a new firewall filter definition.

Configuration of Nested Firewall Filters

Configuring a nested firewall filter for each router (or switch) interface involves separating shared packet-filtering rules from interface-specific packet-filtering rules as follows:

- For each set of packet-filtering rules common across multiple interfaces, configure a separate firewall filter that contains the shared filtering terms.
- For each router (or switch) interface, configure a separate firewall filter that contains:
  - All the filtering terms unique to that interface.
  - An additional filtering term that includes a **filter** reference to the firewall filter that contains the common filtering terms.

Application of Nested Firewall Filters to a Router or Switch Interface

Applying nested firewall filters is no different from applying an unnested firewall filter. For each interface, you can include an **input** or **output** statement (or both) within the **filter** stanza to specify the appropriate nested firewall filter.

Applying nested firewall filters to an interface, the shared filtering terms and the interface-specific firewall filters are applied through *a single nested firewall filter* that includes other filters through the **filter** statement within a separate filtering term.

Related Documentation

- Guidelines for Nesting References to Multiple Firewall Filters on page 842
- Example: Nesting References to Multiple Firewall Filters on page 854

Guidelines for Nesting References to Multiple Firewall Filters

This topic covers the following information:

Statement Hierarchy for Configuring Nested Firewall Filters

To reference a filter from within a filter, include the **filter *filter-name*** statement as a separate filter term:

```
firewall firewall-name {
    family family-name {
        filter filter-name {
            term term-name {
                filter filter-name;
            }
        }
    }
}
```

You can include the firewall configuration at one of the following hierarchy levels:

- **[edit]**
- **[edit logical-systems *logical-system-name*]**

Filter-Defining Terms and Filter-Referencing Terms

You cannot configure a firewall filter term that both references another firewall filter and defines a match condition or action. If a firewall filter term includes the **filter** statement, then it cannot also include the **from** or **then** statement.

For example, the firewall filter term **term term1** in the configuration is *not* valid:

```
[edit]
firewall {
    family inet {
        filter filter_1 {
            term term1 {
                filter filter_2;
                from {
                    source-address 172.16.1.1/32;
                }
                then {
                    accept;
                }
            }
        }
    }
}
```

In order for **term term1** to be a valid filter term, you must either remove the **filter filter_2** statement or remove both the **from** and **then** stanzas.

Types of Filters Supported in Nested Configurations

Nested configurations of firewall filters support firewall filters only. You cannot use service filters or simple filters in a nested firewall filter configuration.

Number of Filter References in a Single Filter

The total number of filters referenced from within a filter cannot exceed 256.

Depth of Filter Nesting

The Junos OS supports a single level of firewall filter nesting. If **filter_1** references **filter_2**, you cannot configure a filter that references a filter that references **filter_1**.

Related Documentation

- Understanding Multiple Firewall Filters in a Nested Configuration on page 841
- Example: Nesting References to Multiple Firewall Filters on page 854

Understanding Multiple Firewall Filters Applied as a List

This topic covers the following information:

The Challenge: Simplify Large-Scale Firewall Filter Administration

Typically, you apply a single firewall filter to an interface in the input or output direction or both. However, this approach might not be practical when you have a device configured with many interfaces. In large environments, you want the flexibility of being able to modify filtering terms common to multiple interfaces without having to reconfigure the filter of every affected interface.

In general, the solution is to apply an effectively “chained” structure of multiple firewall filters to a single interface. You partition your filtering terms into multiple firewall filters that each perform a filtering task. You can then choose which filtering tasks you want to perform for a given interface and apply the filtering tasks to that interface. In this way, you only manage the configuration for a filtering task in a single firewall filter.

The Junos OS policy framework provides two options for managing the application of multiple separate firewall filters to individual router interfaces. One option is to apply multiple filters as a single input list or output list. The other option is to reference a firewall filter from within the term of another firewall filter.

A Solution: Apply Lists of Firewall Filters

The most straightforward way to avoid configuring duplicate filtering terms common to multiple firewall filters is to configure multiple firewall filters and then apply a customized *list* of filters to each interface. The Junos OS uses the filters—in the order in which they appear in the list—to evaluate packets that transit the interface. If you need to modify filtering terms shared across multiple interfaces, you only need to modify one firewall filter that contains those terms.

NOTE: In contrast with the alternative approach (configuring nested firewall filters) applying firewall filter lists combines multiple firewall filters at each interface application point.

Configuration of Multiple Filters for Filter Lists

Configuring firewall filters to be applied in unique lists for each router interface involves separating shared packet-filtering rules from interface-specific packet-filtering rules as follows:

- **Unique filters**—For each set of packet-filtering rules unique to a specific interface, configure a separate firewall filter that contains only the filtering terms for that interface.
- **Shared filters**—For each set of packet-filtering rules common across two or more interfaces, consider configuring a separate firewall filter that contains the shared filtering terms.

TIP: When planning for a large number firewall filters to be applied using filter lists, administrators often organize the shared filters by filtering criteria, by the services to which customers subscribe, or by the purposes of the interfaces.

Application of Filter Lists to a Router Interface

Applying a list of firewall filters to an interface is a matter of selecting the filters that meet the packet-filtering requirements of that interface. For each interface, you can include an **input-list** or **output-list** statement (or both) within the **filter** stanza to specify the relevant filters in the order in which they are to be used:

- Include any filters that contain common filtering terms relevant to the interface.
- Include the filter that contain only the filtering terms unique to the interface.

Interface-Specific Names for Filter Lists

Because a filter list is configured under an interface, the resulting concatenated filter is *interface-specific*.

NOTE: When a filter list is configured under an interface, the resulting concatenated filter is interface-specific, regardless whether the firewall filters in the filter list are configured as interface-specific or not. Furthermore, the instantiation of interface-specfic firewall filters not only creates separate instances of any firewall filter counters, but also separate instances of any policer actions. Any policers applied through an action specified in the firewall filter configuration are applied separately to each interface in the interface group.

The system-generated name of an interface-specific filter consists of the full interface name followed by either '**-i**' for an input filter list or '**-o**' for an output filter list.

- **Input filter list name**—For example, if you use the **input-list** statement to apply a chain of filters to logical interface **ge-1/3/0.0**, the Junos OS uses the following name for the filter:

  ge-1/3/0.0-i

- **Output filter list name**—For example, if you use the **output-list** statement to apply a chain of filters to logical interface **fe-0/1/2.0**, the Junos OS uses the following name for the filter:

  fe-0/1/2.0-o

You can use the interface-specific name of a filter list when you enter a Junos OS operational mode command that specifies a firewall filter name.

How Filter Lists Evaluate Packets When the Matched Term Includes Terminating or Next Term Actions

The device evaluates a packet against the filters in a list sequentially, beginning with the first filter in the list until either a terminating action occurs or the packet is implicitly discarded.

Table 55 on page 847 describes how a firewall filter list evaluates a packet based on whether the matched term specifies a terminating action and the **next term** action. The **next term** action is neither a terminating action nor a nonterminating action but a *flow control* action.

Table 55: Firewall Filter List Behavior

Firewall Filter Actions Included in the Matched Term			
Terminating	next term	Term Description	Packet-Filtering Behavior
Yes	—	The matched term includes a terminating action (such as **discard**) but not the **next term** action	The device executes the terminating action. No subsequent terms in the filter and no subsequent filters in the list are used to evaluate the packet.
—	Yes	The matched term includes the **next term** action, but it does not include any terminating actions.	The device executes any nonterminating actions, then the device evaluates the packet against the next term in the filter or the next filter in the list.
—	—	The matched term includes neither the **next term** action nor any terminating actions.	The device executes any nonterminating actions, then the device implicitly accepts the packet. Because the **accept** action is a terminating action, no subsequent terms in the filter and no subsequent filters in the list are used to evaluate the packet.

For information about terminating actions, see "Firewall Filter Terminating Actions" on page 680.

NOTE: **You cannot configure the next term action with a terminating action in the same firewall filter term.**

How Filter Lists Evaluate Packets When the List Includes Protocol-Independent and IP Firewall Filters

On a single interface associated with a protocol-independent (**family any**) firewall filter and a protocol-specific (**family inet** or **family inet6**) firewall filter simultaneously, the protocol-independent firewall filter executes first.

The terminating action of the first filter determines whether the second filter also evaluates the packet:

- If the first filter terminates by executing the **accept** action, the second filter also evaluates the packet.
- If the first filter terminates without any terms matching the packet (an *implicit* **discard** action), the second filter also evaluates the packet.
- If the first filter terminates by executing an *explicit* **discard** action, the second filter does not evaluate the packet.

Related Documentation

- How Standard Firewall Filters Evaluate Packets on page 570
- Guidelines for Applying Multiple Firewall Filters as a List on page 848
- Example: Applying Lists of Multiple Firewall Filters on page 849

Guidelines for Applying Multiple Firewall Filters as a List

This topic covers the following information:

Statement Hierarchy for Applying Lists of Multiple Firewall Filters

To apply a single filter to the input or output direction of a router (or switch) logical interface, you include the **input** ***filter-name*** or **output** ***filter-name*** statement under the **filter** stanza for a protocol family.

To apply a list of multiple filters to the input or output direction of a router (or switch) logical interface, include the **input-list [** ***filter-names*** **]** or **output-list [** ***filter-names*** **]** statement under the **filter** stanza for a protocol family:

```
interfaces {
  interface-name {
    unit logical-unit-number {
      family family-name {
        filter {
          ...filter-options...
          input-list [ filter-names ];
          output-list [ filter-names ];
        }
      }
    }
  }
}
```

You can include the interface configuration at one of the following hierarchy levels:

- **[edit]**
- **[edit logical-systems** ***logical-system-name*****]**

Filter Input Lists and Output Lists for Router or Switch Interfaces

When applying a list of firewall filters as a list, the following limitations apply:

- You can specify up to 16 firewall filters for a filter input list.
- You can specify up to 16 firewall filters for a filter output list.

Types of Filters Supported in Lists

Lists of multiple firewall filters applied to a router (or switch) interface support standard stateless firewall filters only. You cannot apply lists containing service filters or simple filters to a router (or switch) interface.

Restrictions on Applying Filter Lists for MPLS or Layer 2 CCC Traffic

When applying firewall filters that evaluate MPLS traffic (**family mpls**) or Layer 2 circuit cross-connection traffic (**family ccc**), you can use the **input-list [*filter-names*]** and **output-list [*filter-names*]** statements for all interfaces except the following:

- Management and internal Ethernet (**fxp**) interfaces
- Loopback (**lo0**) interfaces
- USB modem (**umd**) interfaces

Related Documentation

- Understanding Multiple Firewall Filters Applied as a List on page 844
- Example: Applying Lists of Multiple Firewall Filters on page 849

Example: Applying Lists of Multiple Firewall Filters

This example shows how to apply lists of multiple firewall filters.

Requirements

Before you begin, be sure that you have:

- Installed your router or switch, and supported PIC, DPC, or MPC and performed the initial router or switch configuration.
- Configured basic Ethernet in the topology.
- Configured a logical interface to run the IP version 4 (IPv4) protocol (**family inet**) and configured the logical interface with an interface address. This example uses logical interface **ge-1/3/0.0** configured with the IP address 172.16.1.2/30.

  NOTE: **For completeness, the configuration section of this example includes setting an IP address for logical interface ge-1/3/0.0.**

- Verified that traffic is flowing in the topology and that ingress and egress IPv4 traffic is flowing through logical interface **ge-1/3/0.0**.
- Verified that you have access to the remote host that is connected to this router's or switch's logical interface **ge-1/3/0.0**.

Overview

In this example, you configure three IPv4 firewall filters and apply each filter directly to the same logical interface by using a list.

Topology

This example applies the following firewall filters as a *list of input filters* at logical interface **ge-1/3/0.0**. Each filter contains a single term that evaluates IPv4 packets and accepts packets based on the value of the **destination port** field in the TCP header:

- Filter **filter_FTP** matches on the FTP port number (**21**).
- Filter **filter_SSH** matches on the SSH port number (**22**).
- Filter **filter_Telnet** matches on the Telnet port number (**23**).

If an inbound packet does not match any of the filters in the input list, the packet is discarded.

NOTE: The Junos OS uses filters in a list in the order in which the filter names appear in the list. In this simple example, the order is irrelevant because all of the filters specify the same action.

Any of the filters can be applied to other interfaces, either alone (using the **input** or **output** statement) or in combination with other filters (using the **input-list** or **output-list** statement). The objective is to configure multiple “minimalist” firewall filters that you can reuse in interface-specific filter lists.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

- Configure Multiple IPv4 Firewall Filters on page 851
- Apply the Filters to a Logical Interface as an Input List and an Output List on page 852
- Confirm and Commit Your Candidate Configuration on page 852

CLI Quick Configuration

To quickly configure this example, copy the following commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter filter_FTP term 0 from protocol tcp
set firewall family inet filter filter_FTP term 0 from destination-port 21
set firewall family inet filter filter_FTP term 0 then count pkts_FTP
set firewall family inet filter filter_FTP term 0 then accept
set firewall family inet filter filter_SSH term 0 from protocol tcp
set firewall family inet filter filter_SSH term 0 from destination-port 22
set firewall family inet filter filter_SSH term 0 then count pkts_SSH
set firewall family inet filter filter_SSH term 0 then accept
set firewall family inet filter filter_Telnet term 0 from protocol tcp
```

```
set firewall family inet filter filter_Telnet term 0 from destination-port 23
set firewall family inet filter filter_Telnet term 0 then count pkts_Telnet
set firewall family inet filter filter_Telnet term 0 then accept
set firewall family inet filter filter_discard term 1 then count pkts_discarded
set firewall family inet filter filter_discard term 1 then discard
set interfaces ge-1/3/0 unit 0 family inet address 172.16.1.2/30
set interfaces ge-1/3/0 unit 0 family inet filter input-list filter_FTP
set interfaces ge-1/3/0 unit 0 family inet filter input-list filter_SSH
set interfaces ge-1/3/0 unit 0 family inet filter input-list filter_Telnet
set interfaces ge-1/3/0 unit 0 family inet filter input-list filter_discard
```

Configure Multiple IPv4 Firewall Filters

Step-by-Step Procedure

To configure the IPv4 firewall filters:

1. Navigate the CLI to the hierarchy level at which you configure IPv4 firewall filters.

```
[edit]
user@host# edit firewall family inet
```

2. Configure the first firewall filter to count and accept packets for port 21.

```
[edit firewall family inet]
user@host# set filter filter_FTP term 0 from protocol tcp
user@host# set filter filter_FTP term 0 from destination-port 21
user@host# set filter filter_FTP term 0 then count pkts_FTP
user@host# set filter filter_FTP term 0 then accept
```

3. Configure the second firewall filter to count and accept packets for port 22.

```
[edit firewall family inet]
user@host# set filter filter_SSH term 0 from protocol tcp
user@host# set filter filter_SSH term 0 from destination-port 22
user@host# set filter filter_SSH term 0 then count pkt_SSH
user@host# set filter filter_SSH term 0 then accept
```

4. Configure the third firewall filter to count and accept packets from port 23.

```
[edit firewall family inet]
user@host# set filter filter_Telnet term 0 from protocol tcp
user@host# set filter filter_Telnet term 0 from destination-port 23
user@host# set filter filter_Telnet term 0 then count pkts_Telnet
user@host# set filter filter_Telnet term 0 then accept
```

5. Configure the last firewall filter to count the discarded packets.

```
[edit firewall family inet]
user@host# set filter filter_discard term 1 then count pkts_discarded
user@host# set filter filter_discard term 1 then discard
```

Apply the Filters to a Logical Interface as an Input List and an Output List

Step-by-Step Procedure

To apply the six IPv4 firewall filters as a list of input filters and a list of output filters:

1. Navigate the CLI to the hierarchy level at which you apply IPv4 firewall filters to logical interface **ge-1/3/0.0**.

```
[edit]
user@host# edit interfaces ge-1/3/0 unit 0 family inet
```

2. Configure the IPv4 protocol family for the logical interface.

```
[edit interfaces ge-1/3/0 unit 0 family inet]
user@host# set address 172.16.1.2/30
```

3. Apply the filters as a list of input filters.

```
[edit interfaces ge-1/3/0 unit 0 family inet]
user@host# set filter input-list [ filter_FTP filter_SSH filter_Telnet filter_discard ]
```

Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the firewall filters by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
    filter filter_FTP {
        term 0 {
            from {
                protocol tcp;
                destination-port 21;
            }
            then {
                count pkts_FTP;
                accept;
            }
        }
    }
    filter filter_SSH {
        term 0 {
            from {
                protocol tcp;
                destination-port 22;
            }
            then {
                count pkts_SSH;
                accept;
```



```
            }
        }
    }
    filter filter_Telnet {
        term 0 {
            from {
                protocol tcp;
                destination-port 23;
            }
            then {
                count pkts_Telnet;
                accept;
            }
        }
    }
    filter filter_discard {
        term 1 {
            then {
                count pkts_discarded;
                discard;
            }
        }
    }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/0 {
    unit 0 {
        family inet {
            filter {
                input-list [ filter_FTP filter_SSH filter_Telnet filter_discard ];
            }
            address 172.16.1.2/30;
        }
    }
}
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

Confirm that the configuration is working properly.

- Verifying That Inbound Packets Are Accepted Only If Destined for the FTP, SSH or Telnet Port on page 854

Verifying That Inbound Packets Are Accepted Only If Destined for the FTP, SSH or Telnet Port

Purpose Verify that all three filters are active for the logical interface.

Action To verify that input packets are accepted according to the three filters:

1. From the remote host that is connected to this router's (or switch's) logical interface **ge-1/3/0.0**, send a packet with destination port number 21 in the header. The packet should be accepted.

2. From the remote host that is connected to this router's (or switch's) logical interface **ge-1/3/0.0**, send a packet with destination port number 22 in the header. The packet should be accepted.

3. From the remote host that is connected to this router's (or switch's) logical interface **ge-1/3/0.0**, send a packet with destination port number 23 in the header. The packet should be accepted.

4. From the remote host that is connected to this router's (or switch's) logical interface **ge-1/3/0.0**, send a packet with a destination port number *other than* 21, 22, or 23. The packet should be discarded.

5. To display counter information for the list of filters applied to the input at **ge-1/3/0.0-i** enter the **show firewall filter ge-1/3/0.0-i** operational mode command. The command output displays the number of bytes and packets that match filter terms associated with the following counters:

   - **pkts_FTP-ge-1/3/0.0-i**
   - **pkts_SSH-ge-1/3/0.0-i**
   - **pkts_Telnet-ge-1/3/0.0-i**
   - **pkts_discard-ge-1/3/0.0-i**

Related Documentation

- Understanding Multiple Firewall Filters Applied as a List on page 844
- Guidelines for Applying Multiple Firewall Filters as a List on page 848

Example: Nesting References to Multiple Firewall Filters

This example shows how to configure nested references to multiple firewall filters.

- Requirements on page 855
- Overview on page 855

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you configure a firewall filter for a match condition and action combination that can be shared among multiple firewall filters. You then configure two firewall filters that reference the first firewall filter. Later, if the common filtering criteria needs to be changed, you would modify only the one shared firewall filter configuration.

Topology

The **common_filter** firewall filter discards packets that have a UDP source or destination port field number of **69**. Both of the two additional firewall filters, **filter1** and **filter2**, reference the **common_filter**.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

CLI Quick Configuration

To quickly configure this example, copy the following commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter common_filter term common_term from protocol udp
set firewall family inet filter common_filter term common_term from port tftp
set firewall family inet filter common_filter term common_term then discard
set firewall family inet filter filter1 term term1 filter common-filter
set firewall family inet filter filter1 term term2 from address 192.168.0.0/16
set firewall family inet filter filter1 term term2 then reject
set firewall family inet filter filter2 term term1 filter common-filter
set firewall family inet filter filter2 term term2 from protocol udp
set firewall family inet filter filter2 term term2 from port bootps
set firewall family inet filter filter2 term term2 then accept
set interfaces ge-0/0/0 unit 0 family inet address 10.1.0.1/24
set interfaces ge-0/0/0 unit 0 family inet filter input filter1
set interfaces ge-0/0/3 unit 0 family inet address 10.1.3.1/24
set interfaces ge-0/0/0 unit 0 family inet filter input filter2
```

Configure the Nested Firewall Filters

Step-by-Step Procedure

To configure two nested firewall filters that share a common filter:

1. Navigate the CLI to the hierarchy level at which you configure IPv4 firewall filters.

   ```
   [edit]
   user@host# edit firewall family inet
   ```

2. Configure the common filter that will be referenced by multiple other filters.

   ```
   [edit firewall family inet]
   user@host# set filter common_filter term common_term from protocol udp
   user@host# set filter common_filter term common_term from port tftp
   user@host# set filter common_filter term common_term then discard
   ```

3. Configure a filter that references the common filter.

   ```
   [edit firewall family inet]
   user@host# set filter filter1 term term1 filter common-filter
   user@host# set filter filter1 term term2 from address 192.168.0.0/16
   user@host# set filter filter1 term term2 then reject
   ```

4. Configure a second filter that references the common filter.

   ```
   [edit firewall family inet]
   user@host# set filter filter2 term term1 filter common-filter
   user@host# set filter filter2 term term2 from protocol udp
   user@host# set filter filter2 term term2 from port bootps
   user@host# set filter filter2 term term2 then accept
   ```

Apply Both Nested Firewall Filters to Interfaces

Step-by-Step Procedure

To apply both nested firewall filters to logical interfaces:

1. Apply the first nested filter to a logical interface input.

   ```
   [edit]
   user@host# set interfaces ge-0/0/0 unit 0 family inet address 10.1.0.1/24
   user@host# set interfaces ge-0/0/0 unit 0 family inet filter input filter1
   ```

2. Apply the second nested filter to a logical interface input.

   ```
   [edit]
   user@host# set interfaces ge-0/0/3 unit 0 family inet address 10.1.3.1/24
   user@host# set interfaces ge-0/0/0 unit 0 family inet filter input filter2
   ```


Confirm and Commit Your Candidate Configuration

Step-by-Step Procedure

To confirm and then commit your candidate configuration:

1. Confirm the configuration of the firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
    filter common_filter {
        term common_term {
            from {
                protocol udp;
                port tftp;
            }
            then {
                discard;
            }
        }
    }
    filter filter1 {
        term term1 {
            filter common-filter;
        }
        term term2 {
            from {
                address 192.168/16;
            }
            then {
                reject;
            }
        }
    }
    filter filter2 {
        term term1 {
            filter common-filter;
        }
        term term2 {
            from {
                protocol udp;
                port bootps;
            }
            then {
                accept;
            }
        }
    }
}
```

2. Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
ge-0/0/0 {
   unit 0 {
      family inet {
         filter {
            input filter1;
         }
         address 10.1.0.1/24;
      }
   }
}
ge-0/0/3 {
   unit 0 {
      family inet {
         filter {
            input filter2;
         }
         address 10.1.3.1/24;
      }
   }
}
```

3. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Verification

To confirm that the configuration is working properly, enter the **show firewall filter filter1** and **show firewall filter filter2** operational mode commands.

Related Documentation

- Understanding Multiple Firewall Filters in a Nested Configuration on page 841
- Guidelines for Nesting References to Multiple Firewall Filters on page 842

CHAPTER 21

Attaching a Single Firewall Filter to Multiple Interfaces

Interface-Specific Firewall Filter Instances Overview

This topic covers the following information:

Instantiation of Interface-Specific Firewall Filters

On T Series, M120, M320, and MX Series routers, you can enable the Junos OS to automatically create an interface-specific instance of a firewall filter for each interface to which you apply the filter. If you enable interface-specific instantiation of a firewall filter and then apply that filter to multiple interfaces, any **count** actions or **policer** actions configured in the filter terms act on the traffic stream entering or exiting each individual interface, regardless of the sum of traffic on the multiple interfaces.

You can enable this option per firewall filter by including the **interface-specific** statement in the filter configuration.

NOTE: **On T Series, M120, M320, and MX Series routers, interfaces are distributed among multiple packet-forwarding components.**

Interface-specific firewall filtering is not supported on M Series routers other than the M120 and M320 routers. If you apply a firewall filter to multiple interfaces on an M Series router other than the M120 or M320 routers, the filter acts on the sum of traffic entering or exiting those interfaces.

Interface-specific firewall filtering is supported for standard stateless firewall filters and for service filters. Interface-specific instances are not supported for simple filters.

NOTE: **A firewall filter cannot be both interface-specific and interface-shared.**

Interface-Specific Names for Firewall Filter Instances

When the Junos OS creates a separate instance of a firewall filter for a logical interface, the instance is associate with an interface-specific name. The system-generated name of a firewall filter instance consists of the name of the configured filter followed by a hyphen ('**-**'), the full interface name, and either '**-i**' for an input filter instance or '**-o**' for an output filter instance.

- **Input filter instance name**—For example, if you apply the interface-specific firewall filter **filter_s_tcp** to the input at logical interface **at-1/1/1.0**, the Junos OS instantiates an interface-specific filter instance with the following system-generated name:

  filter_s_tcp-at-1/1/1.0-i

- **Output filter instance name**—For example, if you apply the interface-specific firewall filter **filter_s_tcp** to the output at logical interface **so-2/2/2.2**, the Junos OS instantiates an interface-specific filter instance with the following system-generated name:

  count_s_tcp-so-2/2/2.2-o

You can use the interface-specific name of a filter instance when you enter a Junos OS operational mode command that specifies a stateless firewall filter name.

TIP: **When you configure a firewall filter with interface-specific instances enabled, we recommend you limit the filter name to *52 bytes* in length. This is because firewall filter names are restricted to *64 bytes* in length. If a system-generated filter instance name exceeds this maximum length, the policy framework software might reject the instance name.**

Interface-Specific Firewall Filter Counters

Instantiation of interface-specific firewall filters causes the Packet Forwarding Engine to maintain any counters for the firewall filter separately for each interface. You specify interface-specific counters per firewall filter term by specifying the **count *counter-name*** non-terminating action.

The system-generated name of an interface-specific firewall filter counter consists of the name of the configured counter followed by a hyphen ('**-**'), the full interface name, and either '**-i**' for an input filter instance or '**-o**' for an output filter instance.

- **Interface-specific input filter counter name**—For example, suppose you configure the filter counter **count_tcp** for an interface-specific firewall filter. If the filter is applied to the input at logical interface **at-1/1/1.0**, the Junos OS creates the following system-generated counter name:

  count_tcp-at-1/1/1.0-i

- **Interface-specific output filter counter name**—For example, suppose you configure the filter counter **count_udp** for an interface-specific firewall filter. If the filter is applied to the output at logical interface **so-2/2/2.2**, the Junos OS creates the following system-generated counter name:

  count_udp-so-2/2/2.2-o

Interface-Specific Firewall Filter Policers

Instantiation of interface-specific firewall filters not only creates separate instances of any firewall filter counters but also creates separate instances of any policer actions. Any policers applied through an action specified in the firewall filter configuration are applied separately to each interface in the interface group. You specify interface-specific policers per firewall filter term by specifying the **policer *policer-name*** non-terminating action.

Related Documentation

- Example: Configuring Interface-Specific Firewall Filter Counters on page 862

Filtering Packets Received on a Set of Interface Groups Overview

You can configure a firewall filter term that matches packets tagged for a specified *interface group* or set of interface groups. An interface group consists of one or more logical interfaces with the same group number. Packets received on an interface in an interface group are tagged as being part of that group.

NOTE: EX9200 switches do not support *interface groups*. Use the interface-set configuration command as a workaround.

For standard stateless firewall filters, you can filter packets received on an interface group for IPv4, IPv6, virtual private LAN service (VPLS), Layer 2 circuit cross-connection (CCC), and Layer 2 bridging traffic. For service filters, you can filter packets received on an interface group for either IPv4 or IPv6 traffic.

NOTE: You can also configure a firewall filter term that matches on packets tagged for a specified *interface set*. For more information, see "Filtering Packets Received on an Interface Set Overview" on page 862.

Related Documentation

- Example: Configuring a Stateless Firewall Filter on an Interface Group on page 867

Filtering Packets Received on an Interface Set Overview

You can configure a standard stateless firewall filter term that matches packets tagged for a specified *interface set*. An interface set groups two or more physical or logical interfaces into a single interface-set name. You can filter packets received on an interface set for protocol-independent, IPv4, IPv6, MPLS, VPLS, or bridging traffic.

NOTE: You can also configure a standard stateless firewall filter term or a service filter term that matches on packets tagged for a specified *interface group*. For more information, see "Filtering Packets Received on a Set of Interface Groups Overview" on page 861.

Related Documentation

- Example: Configuring a Rate-Limiting Filter Based on Destination Class on page 783
- Example: Filtering Packets Received on an Interface Set on page 707

Example: Configuring Interface-Specific Firewall Filter Counters

This example shows how to configure and apply an interface-specific standard stateless firewall filter.

Requirements

Interface-specific stateless firewall filters are supported on T Series, M120, M320, and MX Series routers only.

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you create an interface-specific stateless firewall filter that counts and accepts packets with source or destination addresses in a specified prefix and the IP protocol type field set to a specific value.

Topology

You configure the interface-specific stateless firewall filter **filter_s_tcp** to count and accept packets with IP source or destination addresses in the **10.0.0.0/12** prefix and the IP protocol type field set to **tcp** (or the numeric value **6**).

The name of the firewall filter counter is **count_s_tcp**.

You apply the firewall filter to multiple logical interfaces:

- **at-1/1/1.0** input
- **so-2/2/2.2** output

Applying the filter to these two interfaces results in two instances of the filter: **filter_s_tcp-at-1/1/1.0-i** and **filter_s_tcp-so-2/2/2.2-o**, respectively.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configure the Interface-Specific Firewall Filter on page 863
- Apply the Interface-Specific Firewall Filter to Multiple Interfaces on page 864
- Confirm Your Candidate Configuration on page 864
- Clear the Counters and Commit Your Candidate Configuration on page 865

CLI Quick Configuration

To quickly configure this example, copy the following commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet filter filter_s_tcp interface-specific
set firewall family inet filter filter_s_tcp term 1 from address 10.0.0.0/12
set firewall family inet filter filter_s_tcp term 1 from protocol tcp
set firewall family inet filter filter_s_tcp term 1 then count count_s_tcp
set firewall family inet filter filter_s_tcp term 1 then accept
set interfaces at-1/1/1 unit 0 family inet filter input filter_s_tcp
set interfaces so-2/2/2 unit 2 family inet filter filter_s_tcp
```

Configure the Interface-Specific Firewall Filter

Step-by-Step Procedure

To configure the interface-specific firewall filter:

1. Create the IPv4 firewall filter **filter_s_tcp**.

```
[edit]
user@host# edit firewall family inet filter filter_s_tcp
```

2. Enable interface-specific instances of the filter.

```
[edit firewall family inet filter filter_s_tcp]
user@host# set interface-specific
```

3. Configure the match conditions for the term.

```
[edit firewall family inet filter filter_s_tcp]
user@host# set term 1 from address 10.0.0.0/12
user@host# set term 1 from protocol tcp
```

4. Configure the actions for the term.

```
[edit firewall family inet filter filter_s_tcp]
user@host# set term 1 then count count_s_tcp
user@host# set term 1 then accept
```

Apply the Interface-Specific Firewall Filter to Multiple Interfaces

Step-by-Step Procedure

To apply the filter **filter_s_tcp** to logical interfaces **at-1/1/1.0** and **so-2/2/2.2**:

1. Apply the interface-specific filter to packets received on logical interface **at-1/1/1.0**.

```
[edit]
user@host# set interfaces at-1/1/1 unit 0 family inet filter input filter_s_tcp
```

2. Apply the interface-specific filter to packets transmitted from logical interface **so-2/2/2.2**.

```
[edit]
user@host# set interfaces so-2/2/2 unit 2 family inet filter filter_s_tcp
```

Confirm Your Candidate Configuration

Step-by-Step Procedure

To confirm your candidate configuration:

1. Confirm the configuration of the stateless firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter filter_s_tcp {
    interface-specific;
    term 1 {
      from {
        address {
          10.0.0.0/12;
        }
        protocol tcp;
      }
      then {
        count count_s_tcp;
        accept;
      }
    }
  }
}
```

2. Confirm the configuration of the interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
at-1/1/1 {
   unit 0
      family inet {
         filter {
            input filter_s_tcp;
         }
      }
   ]
}
so-2/2/2 {
   unit 2
      family inet {
         filter {
            output filter_s_tcp;
         }
      }
   }
}
```

Clear the Counters and Commit Your Candidate Configuration

Step-by-Step Procedure

To clear the counters and commit your candidate configuration:

1. From operational command mode, use the **clear firewall all** command to clear the statistics for all firewall filters.

   To clear only the counters used in this example, include the interface-specific filter instance names:

   ```
   [edit]
   user@host> clear firewall filter filter_s_tcp-at-1/1/1.0-i
   user@host> clear firewall filter filter_s_tcp-so-2/2/2.2-o
   ```

2. Commit your candidate configuration.

   ```
   [edit]
   user@host# commit
   ```

Verification

Confirm that the configuration is working properly.

Verifying That the Filter Is Applied to Each of the Multiple Interfaces

Purpose Verify that the filter is applied to each of the multiple interfaces.

Action Run the **show interfaces** command with the **detail** or **extensive** output level.

1. Verify that the filter is applied to the input for **at-1/1/1.0**:

```
user@host> show interfaces at-1/1/1 detail
Physical interface: at-1/1/1, Enabled, Physical link is Up
  Interface index: 300, SNMP ifIndex: 194, Generation: 183

...
  Logical interface at-1/1/1.0 (Index 64) (SNMP ifIndex 204) (Generation 5)
    Flags: Point-To-Point SNMP-Traps 0x4000 Encapsulation: ATM-SNAP

...
    Protocol inet, MTU: 4470, Generation: 13, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Input Filters: filter_s_tcp-at-1/1/1.0-i,,,,,
```

2. Verify that the filter is applied to the output for **so-2/2/2.2**:

```
user@host> show interfaces so-2/2/2 detail
Physical interface: so-2/2/2, Enabled, Physical link is Up
  Interface index: 129, SNMP ifIndex: 502, Generation: 132

...
  Logical interface so-2/2/2.2 (Index 70) (SNMP ifIndex 536) (Generation 135)

    Flags: Point-To-Point SNMP-Traps 0x4000 Encapsulation: PPP

...
    Protocol inet, MTU: 4470, Generation: 146, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Output Filters: filter_s_tcp-so-2/2/2.2-o,,,,,
```

Verifying That the Counters Are Collected Separately by Interface

Purpose Make sure that the **count_s_tcp** counters are collected separately for the two logical interfaces.

Action Run the **show firewall** command.

```
user@host> show firewall filter filter_s_tcp
Filter: filter_s_tcp
Counters:
Name                                Bytes              Packets
count_s_tcp-at-1/1/1.0-i              420                    5
count_s_tcp-so-2/2/2.2-o             8888                  101
```

Related Documentation

- Interface-Specific Firewall Filter Instances Overview on page 859

Example: Configuring a Stateless Firewall Filter on an Interface Group

Firewall filters are essential for securing a network and simplifying network management. In Junos OS, you can configure a stateless firewall filters to control the transit of data packets through the system and to manipulate packets as necessary. Applying a stateless firewall filter to an interface group helps to filter packets transiting through each interface in the interface group. This example shows how to configure a standard stateless firewall filter to match packets tagged for a particular interface group.

Requirements

This example uses the following hardware and software components:

- Any two Juniper Networks routers or switches that are physically or logically connected to each other through interfaces belonging to a routing instance
- Junos OS Release 7.4 or later

Overview

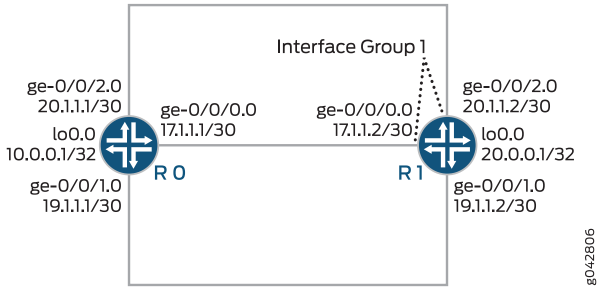
You can apply a stateless firewall filter to an interface group to apply it across all the interfaces in the interface group. This helps you to manage the packet filtering on various interfaces simultaneously.

In this example, you configure two router or switch interfaces to belong to the interface group. You also configure a stateless firewall filter with three terms. In term **term1**, the filter matches packets that have been tagged as received on that interface group and contain an ICMP protocol tag. The filter counts, logs, and rejects packets that match the conditions. In term **term2**, the filter matches packets that contain the ICMP protocol tag. The filter counts, logs, and accepts all packets that match the condition. In term **term3**, the filter counts all the transit packets.

By applying the firewall filter to the routing instance, you can simultaneously apply the filtering mechanism on all the interfaces in the interface group. For this to happen, all the interfaces in the interface group must belong to a single routing instance.

NOTE: **When you apply a firewall filter to a loopback interface, the interface filters all the packets destined to the Routing Engine.**

Figure 53: Configuring a Stateless Firewall Filter on an Interface Group



CLI Quick Configuration shows the configuration for all of the devices in Figure 53 on page 868. The section Step-by-Step Procedure describes the steps on Device R1.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

Device R0

```
set interfaces ge-0/0/0 unit 0 family inet address 172.16.17.1/30
set interfaces ge-0/0/1 unit 0 family inet address 172.16.19.1/30
set interfaces ge-0/0/2 unit 0 family inet address 20.1.1.1/30
set interfaces lo0 unit 0 family inet address 10.0.0.1/32
```

Device R1

```
set firewall family inet filter filter_if_group term term1 from interface-group 1
set firewall family inet filter filter_if_group term term1 from protocol icmp
set firewall family inet filter filter_if_group term term1 then count if_group_counter1
set firewall family inet filter filter_if_group term term1 then log
set firewall family inet filter filter_if_group term term1 then reject
set firewall family inet filter filter_if_group term term2 from protocol icmp
set firewall family inet filter filter_if_group term term2 then count if_group_counter2
set firewall family inet filter filter_if_group term term2 then log
set firewall family inet filter filter_if_group term term2 then accept
set firewall family inet filter filter_if_group term term3 then count default
set interfaces ge-0/0/0 unit 0 family inet filter group 1
set interfaces ge-0/0/0 unit 0 family inet address 172.16.17.2/30
set interfaces ge-0/0/1 unit 0 family inet address 172.16.19.2/30
set interfaces ge-0/0/2 unit 0 family inet filter group 1
set interfaces ge-0/0/2 unit 0 family inet address 20.1.1.2/30
set interfaces lo0 unit 0 family inet address 20.0.0.1/32
set forwarding-options family inet filter input filter_if_group
```

Configure and Apply the Stateless Firewall Filter on an Interface Group

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the stateless firewall filter **filter_if_group** on an interface group:

1. Create the stateless firewall filter **filter_if_group**.

   ```
   [edit firewall]
   user@R1# edit family inet filter filter_if_group
   ```

2. Configure the interfaces and assign two interfaces to interface group **1**.

   ```
   [edit interfaces]
   user@R1# set ge-0/0/0 unit 0 family inet filter group 1
   user@R1# set ge-0/0/0 unit 0 family inet address 172.16.17.2/30

   user@R1# set ge 0/0/1 unit 0 family inet address 172.16.19.2/30

   user@R1# set ge-0/0/2 unit 0 family inet filter group 1
   user@R1# set ge-0/0/2 unit 0 family inet address 20.1.1.2/30

   user@R1# set lo0 unit 0 family inet address 20.0.0.1/32
   ```

3. Configure term **term1** to match packets received on interface group **1** and with the ICMP protocol.

   ```
   [edit firewall]
   user@R1# set family inet filter filter_if_group term term1 from interface-group 1
   user@R1# set family inet filter filter_if_group term term1 from protocol icmp
   ```

4. Configure term **term1** to count, log, and reject all the matching packets.

   ```
   [edit firewall]
   user@R1# set family inet filter filter_if_group term term1 then count if_group_counter1
   user@R1# set family inet filter filter_if_group term term1 then log
   user@R1# set family inet filter filter_if_group term term1 then reject
   ```

5. Configure term **term2** to match packets with the ICMP protocol.

   ```
   [edit firewall]
   user@R1# set family inet filter filter_if_group term term2 from protocol icmp
   ```

6. Configure term **term2** to count, log, and accept all the matching packets.

   ```
   [edit firewall]
   user@R1# set family inet filter filter_if_group term term2 then count if_group_counter2
   user@R1# set family inet filter filter_if_group term term2 then log
   user@R1# set family inet filter filter_if_group term term2 then accept
   ```

7. Configure term **term3** to count all the transit packets.

```
[edit firewall]
user@R1# set family inet filter filter_if_group term term3 then count default
```

8. Apply the firewall filter to the router's (or switch's) interface group by applying it to the routing instance.

```
[edit]
user@R1# set forwarding-options family inet filter input filter_if_group
```

9. If you are done configuring the device, commit your candidate configuration.

```
[edit]
user@host# commit
```

Results

From configuration mode, confirm your configuration by issuing the **show interfaces**, **show firewall**, and **show forwarding-options** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@R1# show interfaces
ge-0/0/0 {
   unit 0 {
      family inet {
         filter {
            group 1;
         }
         address 172.16.17.2/30;
      }
   }
}
ge-0/0/1 {
   unit 0 {
      family inet {
         address 172.16.19.2/30;
      }
   }
}
ge-0/0/2 {
   unit 0 {
      family inet {
         filter {
            group 1;
         }
         address 20.1.1.2/30;
      }
   }
}
lo0 {
   unit 0 {
```



```
        family inet {
            address 20.0.0.1/32;
        }
    }
}

[edit]
user@R1# show firewall
family inet {
    filter filter_if_group {
        term term1 {
            from {
                interface-group 1;
                protocol icmp;
            }
            then {
                count if_group_counter1;
                log;
                reject;
            }
        }
        term term2 {
            from {
                protocol icmp;
            }
            then {
                count if_group_counter2;
                log;
                accept;
            }
        }
        term term3 {
            then count default;
        }
    }
}

[edit]
user@R1# show forwarding-options
family inet {
    filter {
        input filter_if_group;
    }
}
```

Verification

Confirm that the configuration is working properly.

- Verifying the Configuration of the Interfaces on page 871
- Verifying Stateless Firewall Filter Configuration on page 872

Verifying the Configuration of the Interfaces

Purpose Verify that the interfaces are properly configured.

Action To display the state of the interfaces, use the **show interfaces terse** operational mode command.

Device R0

```
user@R0> show interfaces terse
Interface               Admin Link Proto    Local                 Remote
ge-0/0/0                up    up
ge-0/0/0.0              up    up   inet     172.16.17.1/30
                                   multiservice
ge-0/0/1                up    up
ge-0/0/1.0              up    up   inet     172.16.19.1/30
                                   multiservice
ge-0/0/2                up    up
ge-0/0/2.0              up    up   inet     20.1.1.1/30
                                   multiservice
lo0                     up    up
lo0.0                   up    up   inet     10.0.0.1            --> 0/0
```

Device R1

```
user@R1> show interfaces terse
Interface               Admin Link Proto    Local                 Remote
ge-0/0/0                up    up
ge-0/0/0.0              up    up   inet     172.16.17.2/30
                                   multiservice
...
ge-0/0/1                up    up
ge-0/0/1.0              up    up   inet     172.16.19.2/30
                                   multiservice
ge-0/0/2                up    up
ge-0/0/2.0              up    up   inet     20.1.1.2/30
                                   multiservice
...
```

Meaning All the interfaces on Devices R0 and R1 are physically connected and up. The interface group 1 on Device R1 consists of two interfaces, namely ge-0/0/0.0 and ge-0/0/2.0.

Verifying Stateless Firewall Filter Configuration

Purpose Verify that the firewall filter match conditions are configured properly.

Action

- To display the firewall filter counters, enter the show firewall **filter filter_if_group** operational mode command.

```
user@R1> show firewall filter filter_if_group

Filter: filter_if_group
Counters:
Name                                                Bytes              Packets
default                                            192975                 3396
if_group_counter1                                    2520                   30
if_group_counter2                                    2604                   41
```


- To display the local log of packet headers for packets evaluated by the firewall filter, enter the show firewall log operational mode command.

```
user@R1> show firewall log
Log :
Time      Filter    Action Interface     Protocol        Src Addr
            Dest Addr
22:27:33  pfe        A      lo0.0         ICMP            20.1.1.2
            20.1.1.1
22:27:33  pfe        R      ge-0/0/2.0    ICMP            20.1.1.1
            20.1.1.2
22:27:32  pfe        A      lo0.0         ICMP            20.1.1.2
            20.1.1.1
22:27:32  pfe        R      ge-0/0/2.0    ICMP            20.1.1.1
            20.1.1.2
22:27:31  pfe        A      lo0.0         ICMP            20.1.1.2
            20.1.1.1
22:27:31  pfe        R      ge-0/0/2.0    ICMP            20.1.1.1
            20.1.1.2
22:27:30  pfe        A      lo0.0         ICMP            20.1.1.2
            20.1.1.1
22:27:30  pfe        R      ge-0/0/2.0    ICMP            20.1.1.1
            20.1.1.2
22:27:29  pfe        A      lo0.0         ICMP            20.1.1.2
            20.1.1.1
22:27:29  pfe        A      lo0.0         ICMP            20.1.1.2
            20.1.1.1
22:27:29  pfe        R      ge-0/0/2.0    ICMP            20.1.1.1
            20.1.1.2
22:27:21  pfe        A      ge-0/0/1.0    ICMP            172.16.19.1
               172.16.19.2
22:27:20  pfe        A      ge-0/0/1.0    ICMP            172.16.19.1
               172.16.19.2
22:27:19  pfe        A      ge-0/0/1.0    ICMP            172.16.19.1
               172.16.19.2
22:27:18  pfe        A      ge-0/0/1.0    ICMP            172.16.19.1
               172.16.19.2
22:27:04  pfe        A      lo0.0         ICMP            172.16.17.2
               172.16.17.1
22:27:04  pfe        R      ge-0/0/0.0    ICMP            172.16.17.1
               172.16.17.2
22:27:04  pfe        A      lo0.0         ICMP            172.16.17.2
               172.16.17.1
22:27:04  pfe        R      ge-0/0/0.0    ICMP            172.16.17.1
               172.16.17.2
22:27:02  pfe        A      lo0.0         ICMP            172.16.17.2
               172.16.17.1
22:27:02  pfe        R      ge-0/0/0.0    ICMP            172.16.17.1
               172.16.17.2
22:27:01  pfe        A      lo0.0         ICMP            172.16.17.2
               172.16.17.1
22:27:01  pfe        R      ge-0/0/0.0    ICMP            172.16.17.1
               172.16.17.2
22:27:00  pfe        A      lo0.0         ICMP            172.16.17.2
               172.16.17.1
22:27:00  pfe        R      ge-0/0/0.0    ICMP            172.16.17.1
               172.16.17.2
22:24:48  filter_if_group A fxp0.0       ICMP            10.92.16.2
            10.92.26.176
```


- To make sure that the firewall filters are active on interface group 1 on Device R1, use the **ping *<address>*** operational mode command on the CLI of Device R0.

```
user@R0> ping 172.16.17.2
PING 172.16.17.2 (172.16.17.2): 56 data bytes
36 bytes from 172.16.17.2: Communication prohibited by filter
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 f46b   0 0000  40  01 6239 172.16.17.1  172.16.17.2

36 bytes from 172.16.17.2: Communication prohibited by filter
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 f479   0 0000  40  01 622b 172.16.17.1  172.16.17.2

36 bytes from 172.16.17.2: Communication prohibited by filter
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 f487   0 0000  40  01 621d 172.16.17.1  172.16.17.2

user@R0> ping 20.1.1.2
PING 20.1.1.2 (20.1.1.2): 56 data bytes
36 bytes from 20.1.1.2: Communication prohibited by filter
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 f5bd   0 0000  40  01 5ae7 20.1.1.1  20.1.1.2

36 bytes from 20.1.1.2: Communication prohibited by filter
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 f5cd   0 0000  40  01 5ad7 20.1.1.1  20.1.1.2

36 bytes from 20.1.1.2: Communication prohibited by filter
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 f5d9   0 0000  40  01 5acb 20.1.1.1  20.1.1.2

36 bytes from 20.1.1.2: Communication prohibited by filter
Vr HL TOS  Len   ID Flg  off TTL Pro  cks      Src      Dst
 4  5  00 0054 f5f6   0 0000  40  01 5aae 20.1.1.1  20.1.1.2
```

- To make sure that the firewall filter is not applied on an interface that is not in interface group 1, use the **ping *<address>*** operational mode command on the CLI of Device R0.

```
user@R0> ping 172.16.19.2
PING 172.16.19.2 (172.16.19.2): 56 data bytes
64 bytes from 172.16.19.2: icmp_seq=0 ttl=64 time=8.689 ms
64 bytes from 172.16.19.2: icmp_seq=1 ttl=64 time=4.076 ms
64 bytes from 172.16.19.2: icmp_seq=2 ttl=64 time=8.501 ms
64 bytes from 172.16.19.2: icmp_seq=3 ttl=64 time=3.954 ms
...
```

Meaning

The stateless firewall filter is applied to all interfaces in interface group 1. The term **term1** match condition in the stateless firewall filter counts, logs, and rejects packets that are received on or sent from the interfaces in interface group 1 and with a source ICMP protocol. The term **term2** match condition matches packets tagged with the ICMP protocol and counts, logs, and accepts those packets. The term **term3** match condition counts all the transit packets.

Related Documentation

- Filtering Packets Received on a Set of Interface Groups Overview on page 861

CHAPTER 22

Configuring Filter-Based Tunneling Across IP Networks

Understanding Filter-Based Tunneling Across IPv4 Networks

This topic covers the following information:

Understanding Filter-Based Tunneling Across IPv4 Networks

Generic routing encapsulation (GRE) in its simplest form is the encapsulation of any network layer protocol over any other network layer protocol to connect disjointed networks that lack a native routing path between them. You can configure an IPv4 network to transport IPv4, IPv6, or MPLS transit traffic by using GRE tunneling protocol mechanisms initiated by two standard firewall filter actions. This feature is also supported in logical systems.

When you configure GRE tunneling with firewall filters, you do not need to create tunnel interfaces on Tunnel Services physical interface cards (PICs) or on MPC3E Modular Port Concentrators (MPCs). Instead, Packet Forwarding Engines provide tunnel services to Ethernet logical interfaces or aggregated Ethernet interfaces hosted on Modular Interface Cards (MICs) or MPCs in MX Series 3D Universal Edge Routers.

NOTE: GRE is a connectionless and stateless Layer 3 encapsulation protocol, and it offers no mechanisms for reliability, flow control, or sequencing. Traffic flows through the tunnel provided that the tunnel destination is routable.

Two MX Series routers installed as provider edge (PE) routers provide connectivity to customer edge (CE) routers on two disjoint networks. MIC or MPC interfaces on the PE routers perform GRE IPv4 encapsulation and de-encapsulation of payloads.

NOTE: Filter-based GRE tunneling is supported on PTX Series routers only when network services is set to **enhanced-mode**. For more information, see **enhanced-mode**.

Ingress Firewall Filter on the Ingress PE Router

On the ingress PE router, you configure a tunnel definition that specifies a unidirectional GRE tunnel. On a MIC or MPC ingress logical interface, you attach an encapsulating firewall filter. The firewall filter action references a tunnel definition and initiates the encapsulation of matched packets. The encapsulation process attaches an IPv4 header and a GRE header to the payload packet and then forwards the resulting GRE packet to the filter-specified tunnel.

Ingress Firewall Filter on the Egress PE Router

On the egress PE router, you attach a de-encapsulating firewall filter to the input of all MIC or MPC logical interfaces that are advertised addresses for the router. The firewall filter initiates the de-encapsulation of GRE protocol packets. De-encapsulation removes the inner GRE header and then forwards the original payload packet to its original destination on the destination customer network. If the action specifies an optional routing instance, route lookup is performed using that secondary table instead of the primary table.

Characteristics of Filter-Based Tunneling Across IPv4 Networks

Filter-based tunnels across IPv4 networks are unidirectional. They transport transit packets only, and they do not require tunnel interfaces.

Unidirectional Tunneling

Filter-based tunneling across IPv4 networks is unidirectional. You construct a filter-based GRE tunnel by attaching standard firewall filters at the *input* of each tunnel endpoint (at both the ingress PE router and the egress PE router). At the input to the ingress PE router, you apply an encapsulating firewall filter. At the input to the egress PE router, you apply a de-encapsulating firewall filter.

If you want to configure bidirectional GRE tunneling, you can use the same pair of PE routers, but you must configure a second tunnel in the reverse direction.

Transit Traffic Payloads

A filter-based GRE IPv4 tunnel can transport unicast or multicast transit traffic payloads only. Filter-initiated encapsulation and de-encapsulation operations execute on Packet Forwarding Engines for Ethernet logical interfaces and aggregated Ethernet interfaces hosted on MICs or MPCs in MX Series routers. This design enables more efficient use of Packet Forwarding Engine bandwidth as compared to GRE tunneling using tunnel interfaces. One of the trade-offs for this optimization, however, is the inability to transport router control traffic.

Packet Forwarding Engines operate in the Junos OS *forwarding plane* to process packets by forwarding them between input and output interfaces using a locally stored forwarding table (a local copy of the information from the Routing Engine). Routing Engines, on the other hand, operate in the Junos OS *control plane* to handle system management, user access to the router, and processes for routing protocols, router interface control, and some chassis component control. The Junos OS architecture separates the functions of these planes to enable flexibility of platform support and scalability of platform performance. Ingress control packets are directed to the control plane where the GRE encapsulation and de-encapsulation processes of the Packet Forwarding Engine are not available.

Although you can apply firewall filters to loopback addresses, GRE encapsulating and de-encapsulating firewall filter actions are not supported on router loopback interfaces.

Compact Configuration for Multiple GRE Tunnels

Firewall filters support a wide variety of match criteria and, by extension, the ability to terminate multiple GRE tunnels that match criteria specified in a single firewall filter definition. By creating multiple tunnels, each with its own set of match conditions, you can create tunnels that do not interfere with customer GRE packets or with one another and that re-inject packets to separate routing tables after de-encapsulation.

Tunneling with Firewall Filters and Tunneling with Tunnel Interfaces

Unlike tunneling with firewall filters, tunneling with tunnel interfaces supports router control traffic (in addition to transit traffic) and encryption. On the other hand, tunneling with firewall filters carries advantages in performance and scaling.

Tunnel Security

Filter-based tunneling across IPv4 networks is not encrypted. If you require secure tunneling, you must use IP Security (IPsec) encryption, which is not supported on MIC or MPC interfaces. However, Multiservices DPC (MS-DPC) interfaces on MX240, MX480, and MX960 routers support IPsec tools for configuring manual or dynamic security associations (SAs) for encryption of data traffic as well as traffic destined to or originating at the Routing Engine.

For information about Junos OS support for the IPsec security suite for the IPv4 and IPv6 network layers, see *Security Services Administration Guide for Routing Devices*, and *Enabling Service Packages*.

IPsec encryption is also supported on Adaptive Services PIC interfaces and Multiservices PIC interfaces on supported M Series Multiservice Edge Routers and T Series Core Routers.

Forwarding Performance

Filter-based tunneling across IPv4 networks enables more efficient use of Packet Forwarding Engine bandwidth as compared to GRE tunneling using tunnel interfaces. Encapsulation, de-encapsulation, and route lookup are packet header-processing activities that, for firewall filter-based tunneling, are performed on the Junos Trio chipset-based

Packet Forwarding Engine. Consequently, the encapsulator never needs to send payload packets to a separate tunnel interface (which might reside on a PIC in a different slot than the interface that receives payload packets).

Forwarding Scalability

Forwarding GRE traffic with tunnel interfaces requires traffic to be sent to a slot that hosts the tunnel interfaces. When you use tunnel interfaces to forward GRE traffic, this requirement limits the amount of traffic that can be forwarded per GRE tunnel destination address.

As an example, suppose you want to send 100 Gbps of GRE traffic from Router A to Router B and you have only 10 Gbps interfaces. To ensure that your configuration does not encapsulate all the traffic on the same board going to the same 10 Gbps interface, you must distribute the traffic across multiple encapsulation points.

Related Documentation

- Interfaces That Support Filter-Based Tunneling Across IPv4 Networks on page 881
- Components of Filter-Based Tunneling Across IPv4 Networks on page 883
- Firewall Filter Terminating Actions on page 680
- **tunnel-end-point on page 1269**
- Example: Transporting IPv6 Traffic Across IPv4 Using Filter-Based Tunneling on page 888

Firewall Filter-Based L2TP Tunneling in IPv4 Networks Overview

The Layer 2 Tunneling Protocol (L2TP) is a client-server protocol that allows the Point-to-Point Protocol (PPP) to be tunneled across a network. L2TP encapsulates Layer 2 packets, such as PPP, for transmission across a network. An L2TP access concentrator (LAC), configured on an access device, receives packets from a remote client and forwards them to an L2TP network server (LNS) on a remote network. L2TPv3 defines the base control protocol and encapsulation for tunneling multiple Layer 2 connections between two IPv6 nodes. The significant differences between L2TPv2 and L2TPv3 include the following:

- Separation of all PPP-related AVPs and references, which enables the inclusion of a portion of the L2TP data header that was specific to the needs of PPP.
- Transition from a 16-bit Session ID and Tunnel ID to a 32-bit Session ID and Control Connection ID respectively.
- Extension of the tunnel authentication mechanism to cover the entire control message rather than just a portion of certain messages.
- L2TPv3 is supported for IPv6 only.
- For firewall filters, only data plane L2TPv3 encapsulation/ decapsulation is supported.

L2TP is comprised of two types of messages, control messages and data messages (sometimes referred to as control packets and data packets respectively). Control messages are used in the establishment, maintenance, and clearing of control connections and sessions. These messages utilize a reliable control channel within L2TP to guarantee delivery. Data messages are used to encapsulate the L2 traffic being carried over the L2TP session.

You can configure an IPv4 network to transport IPv4, IPv6, or MPLS transit traffic by using GRE tunneling protocol mechanisms initiated by two standard firewall filter actions. This feature is also supported in logical systems. When you configure L2TP tunneling with firewall filters, you do not need to create tunnel interfaces on Tunnel Services physical interface cards (PICs) or on MPC3E Modular Port Concentrators (MPCs). Instead, Packet Forwarding Engines provide tunnel services to Ethernet logical interfaces or aggregated Ethernet interfaces hosted on Modular Interface Cards (MICs) or MPCs in MX Series 3D Universal Edge Routers.

Two MX Series routers installed as provider edge (PE) routers provide connectivity to customer edge (CE) routers on two disjoint networks. MIC or MPC interfaces on the PE routers perform L2TP IPv4 encapsulation and de-encapsulation of payloads. After decapsulation, packets are sent to the local interface of a routing table specified in the action, or to the default routing table, based on the protocol field of the L2TP header. However, an L2TP packet can optionally be sent across the fabric with a token equal to an output interface index to perform Layer 2 cross- connect. You can specify the output interface specifier to be used for the L2TP packet to be sent by including the **decapsulate l2tp output-interface *interface-name* cookie *l2tpv3-cookie*** statement at the **[edit firewall family *family-name* filter *filter-name* term *term-name* then]** hierarchy level.

During decapsulation, the inner header must be Ethernet for L2TP tunnels. Forwarding class by default is applied before the firewall and it is not preserved for the decapsulated packet (by using the **forwarding-class *class-name*** statement at the **[edit firewall family family-name]** hierarchy level, which is a nonterminating filter action). However, you can specify the forwarding class that the packet must be classified against by including the filter action for a decapsulated packet by using the **decapsulate l2tp forwarding-class *class-name*** statement at the **[edit firewall family *family-name* filter filter-name term *term-name* then]** hierarchy level.

The following field definitions are defined for use in all L2TP Session Header encapsulations.

- The Session ID field is a 32-bit field containing a non-zero identifier for a session. L2TP sessions are named by identifiers that have local significance only. The same logical session will be given different Session IDs by each end of the control connection for the life of the session. When the L2TP control connection is used for session establishment, Session IDs are selected and exchanged as Local Session ID AVPs during the creation of a session. The Session ID alone provides the necessary context for all further packet processing, including the presence, size, and value of the Cookie, the type of L2-Specific Sublayer, and the type of payload being tunneled.
- The optional Cookie field contains a variable-length value (maximum 64 bits) used to check the association of a received data message with the session identified by the Session ID. The Cookie field must be set to the configured or signaled random value for this session. The Cookie provides an additional level of guarantee that a data message has been directed to the proper session by the Session ID. A well-chosen Cookie might prevent inadvertent misdirection of random packets with recently reused Session IDs or for Session IDs subject to packet corruption. The Cookie might also

provide protection against some specific malicious packet insertion attacks. When the L2TP control connection is used for session establishment, random Cookie values are selected and exchanged as Assigned Cookie AVPs during session creation.

A session is a logical connection created between the LAC and the LNS when an end-to-end PPP connection is established between a remote system and the LNS. There is a one-to-one relationship between established L2TP sessions and their associated PPP connections. A tunnel is an aggregation of one or more L2TP sessions.

Starting with Junos OS Release 15.1, decapsulation of IP packets that are sent through an L2TP tunnel with standard firewall filter match conditions and actions specified is performed using a Layer 3 lookup. In Junos OS release 14.2 and earlier, decapsulation of traffic over an L2TP tunnel with firewall filter actions configured is performed using Layer 2 interface properties.

This topic covers the following information:

Unidirectional Tunneling

Filter-based L2TP tunnels across IPv4 networks are unidirectional. They transport transit packets only, and they do not require tunnel interfaces. Although you can apply firewall filters to loopback addresses, GRE encapsulating and de-encapsulating firewall filter actions are not supported on router loopback interfaces. Filter-initiated encapsulation and de-encapsulation operations of L2TP packets execute on Packet Forwarding Engines for Ethernet logical interfaces and aggregated Ethernet interfaces hosted on MICs or MPCs in MX Series routers. This design enables more efficient use of Packet Forwarding Engine bandwidth as compared to GRE tunneling using tunnel interfaces. One of the trade-offs for this optimization, however, is the inability to transport router control traffic.

Tunnel Security

Filter-based tunneling across IPv4 networks is not encrypted. If you require secure tunneling, you must use IP Security (IPsec) encryption, which is not supported on MIC or MPC interfaces. However, Multiservices DPC (MS-DPC) interfaces on MX240, MX480, and MX960 routers support IPsec tools for configuring manual or dynamic security associations (SAs) for encryption of data traffic as well as traffic destined to or originating at the Routing Engine.

Forwarding Performance

Filter-based tunneling across IPv4 networks enables more efficient use of Packet Forwarding Engine bandwidth as compared to L2TP tunneling using tunnel interfaces. Encapsulation, de-encapsulation, and route lookup are packet header-processing activities that, for firewall filter-based tunneling, are performed on the Junos Trio chipset-based Packet Forwarding Engine. Consequently, the encapsulator never needs to send payload packets to a separate tunnel interface (which might reside on a PIC in a different slot than the interface that receives payload packets).

Forwarding Scalability

Forwarding L2TP traffic with tunnel interfaces requires traffic to be sent to a slot that hosts the tunnel interfaces. When you use tunnel interfaces to forward GRE traffic, this requirement limits the amount of traffic that can be forwarded per GRE tunnel destination address. As an example, suppose you want to send 100 Gbps of L2TP traffic from Router A to Router B and you have only 10 Gbps interfaces. To ensure that your configuration does not encapsulate all the traffic on the same board going to the same 10 Gbps interface, you must distribute the traffic across multiple encapsulation points.

Release History Table

Release	Description
15.1	Starting with Junos OS Release 15.1, decapsulation of IP packets that are sent through an L2TP tunnel with standard firewall filter match conditions and actions specified is performed using a Layer 3 lookup.
14.2	In Junos OS release 14.2 and earlier, decapsulation of traffic over an L2TP tunnel with firewall filter actions configured is performed using Layer 2 interface properties.

Related Documentation

- Interfaces That Support Filter-Based Tunneling Across IPv4 Networks on page 881
- Components of Filter-Based Tunneling Across IPv4 Networks on page 883
- Firewall Filter Terminating Actions on page 680
- **tunnel-end-point on page 1269**
- Example: Transporting IPv6 Traffic Across IPv4 Using Filter-Based Tunneling on page 888

Interfaces That Support Filter-Based Tunneling Across IPv4 Networks

You can attach IPv4 encapsulation and de-encapsulation firewall filters to the input of Ethernet logical interfaces or aggregated Ethernet interfaces hosted on Modular Interface Cards (MICs) or Modular Port Concentrators (MPCs) in MX Series routers.

NOTE: Filter-based generic routing encapsulation (GRE) tunneling is supported on PTX Series routers only when network services is set to enhanced-mode. For more information, see enhanced-mode.

- Interfaces on MX240, MX480, MX960, MX2010, and MX2020 Routers on page 882
- Interfaces on MX5, MX10, MX40, and MX80 Routers on page 882
- CLI Commit Check for Filter-Based Tunneling Across IPv4 Networks on page 882

Interfaces on MX240, MX480, MX960, MX2010, and MX2020 Routers

On MX240, MX480, MX960, MX2010, and MX2020 routers, firewall filter actions for IPv4 tunneling are supported on Ethernet logical interfaces or aggregated Ethernet interfaces configured on the following types of ports:

- Ports on MICs that insert into slots in MPCs, which have two Packet Forwarding Engines.
- Ports on a 16-port 10-Gigabit Ethernet MPC (MPC-3D-16XGE-SFPP), a specialized fixed-configuration MPC that has four Packet Forwarding Engines and contains no slots for MICs.

For these physical interfaces, Trio chipset-based Packet Forwarding Engine processes operate in *fabric mode* to provide forwarding and storage functions and lookup and processing functions between Ethernet interfaces and the routing fabric of the chassis.

For information about MPCs, see *MX Series MPC Overview* and *MPCs Supported by MX Series Routers*. For information about MICs, see *MX Series MIC Overview* and *MICs Supported by MX Series Routers*.

Interfaces on MX5, MX10, MX40, and MX80 Routers

On the MX Series midrange family of routers (MX5, MX10, MX40, and MX80 routers), firewall filter actions for IPv4 tunneling are supported on Ethernet logical interfaces and aggregated Ethernet interfaces configured on ports on a built-in MIC or on MICs that install into dedicated slots in the router chassis.

- The MX80 router—available as a modular (MX80) or fixed (MX80-48T) chassis—has a built-in 4-port 10-Gigabit Ethernet MIC. The modular chassis has two dedicated slots for MICs. The fixed chassis has 48 built-in tri-rate (10/100/1000Base-T) RJ-45 ports in place of two front-pluggable MIC slots.
- On the MX40 router, only the first two of the four built-in 10-Gigabit Ethernet MIC ports are enabled. As with the modular MX80, the two front-pluggable MIC slots are enabled and support dual-wide MICs that span the two slots.
- The MX5 and MX10 routers are pre-populated with a front-pluggable 20-port Gigabit Ethernet MIC with SFP, and none of the four built-in 10-Gigabit Ethernet MIC ports is enabled. The MX10 supports MICs in both front-pluggable slots, but the MX5 supports MICs in the second slot only.

For more information, see *MX5, MX10, MX40, and MX80 Modular Interface Card Description*.

The MX Series midrange routers have no switching fabric, and the single Packet Forwarding Engine resides on the base board of the chassis and operates in *standalone mode*. In standalone mode, the Packet Forwarding Engine provides—in addition to forwarding and storage functions and lookup and processing functions—hierarchical queuing, congestion management, and granular statistical functions.

CLI Commit Check for Filter-Based Tunneling Across IPv4 Networks

If you commit a configuration that attaches an encapsulating or de-encapsulating firewall filter to an interface that does not support filter-based tunneling across IPv4 networks, a system event writes a syslog warning message that the interface does not support the filter.

Related Documentation

- Understanding Filter-Based Tunneling Across IPv4 Networks on page 875
- Components of Filter-Based Tunneling Across IPv4 Networks on page 883
- Firewall Filter Terminating Actions on page 680
- **tunnel-end-point on page 1269**
- Example: Transporting IPv6 Traffic Across IPv4 Using Filter-Based Tunneling on page 888

Components of Filter-Based Tunneling Across IPv4 Networks

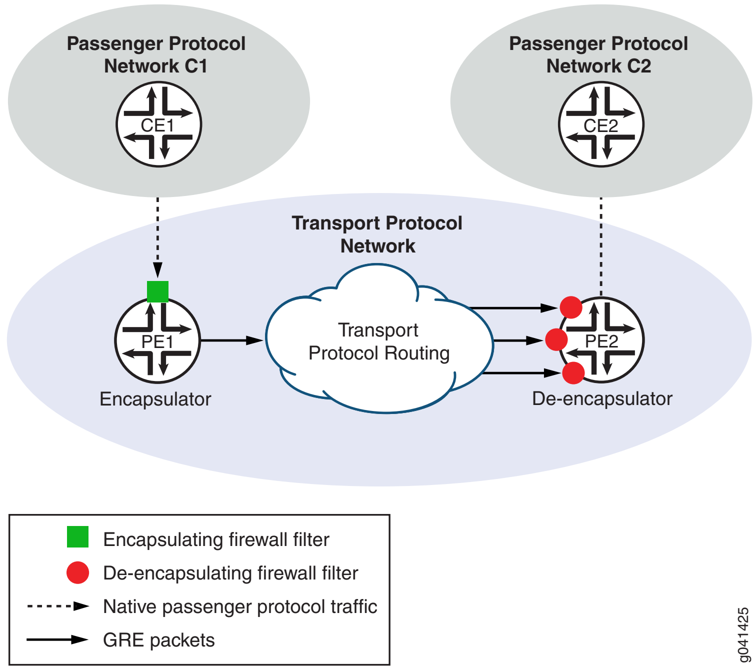
This topic covers the following information:

Topology of Filter-Based Tunneling Across IPv4 Networks

NOTE: Filter-based generic routing encapsulation (GRE) tunneling is supported on PTX Series routers only when network services is set to **enhanced-mode**. For more information, see **enhanced-mode**.

Figure 54 on page 884 shows the path of passenger protocol packets from customer network C1 as they are transported across a service provider IPv4 network to customer network C2.

Figure 54: Unidirectional Filter-Based Tunnel Across an IPv4 Network



In this example topology, C1 and C2 are disjoint networks that lack a native routing path between them. The IPv4 transport network is configured with a unidirectional generic routing encapsulation (GRE) tunnel from PE1 to PE2 using firewall filters and without requiring tunnel interfaces. The GRE tunnel from PE1 to PE2 provides a logical path from C1 to C2 across the IPv4 transport network.

Routing of GRE Packets Across the Tunnel

Traffic flows through the tunnel provided that PE2 is routable from PE1. Routing paths from PE1 to PE2 can be provided by static routes manually added to routing tables or by static or dynamic route-sharing protocols.

Routing of Passenger Protocol Packets from PE2 to C2

By default, PE2 forwards packets based on interface routes (direct routes) imported from the primary routing table. As an option, the de-encapsulating filter can specify that the Packet Forwarding Engine uses an alternate routing table to forward payload packets to the destination customer network. Specify the alternate routing table in a routing instance installed with routes into C2, then use a routing information base (RIB) group definition to share the primary routes with the alternate routes. A RIB group specifies the sharing of routing information (including routes learned from peers, local routes resulting from the application of protocol policies to the learned routes, and the routes advertised to peers) of multiple routing tables.

Terminology at the Network Layer Protocols Level

In filter-based tunneling across an IPv4 network, the network-layer protocols are described in the following terms:

passenger protocol—The type of protocol (IPv4, IPv6, or MPLS) used by the networks that are connected by a GRE tunnel. Packets that are encapsulated and routed across the transport network are *payload packets*.

encapsulation protocol—The type of network layer protocol (GRE) used to encapsulate passenger protocol packets so that the resulting GRE packets can be carried over the transport protocol network as the packet payload.

transport protocol—The type of protocol (IPv4) used by the network that routes passenger protocol packets through a GRE tunnel. The transport protocol is also called the *delivery protocol*.

Terminology at the Ingress PE Router

In filter-based tunneling across an IPv4 network, an egress PE router is described in the following terms:

encapsulator—A PE router that receives packets from a passenger protocol source network, adds an encapsulation protocol (GRE) header and a transport protocol (IPv4) header to this payload, and forwards the resulting GRE packet to the GRE tunnel. This ingress node is also known as the *tunnel source*.

encapsulating interface—On the encapsulator, an Ethernet logical interface or an aggregated Ethernet interface configured on a customer-facing interface hosted on a MIC or an MPC. The encapsulating interface receives passenger protocol packets from a CE router. For more information, see "Interfaces That Support Filter-Based Tunneling Across IPv4 Networks" on page 881.

encapsulation filter—On the encapsulator, a firewall filter that you apply to the input of the encapsulating interface. The encapsulating filter action causes the Packet Forwarding Engine to use information in the specified tunnel template to encapsulate matched packets and forward the resulting GRE packets.

tunnel source interface—On the encapsulator, one or more core-facing egress interfaces to the tunnel.

tunnel template—On the encapsulator, a named CLI construct that defines the characteristics of a tunnel:

- Transport protocol family (IPv4).
- IP address or address range of tunnel-facing *egress* interfaces on the encapsulator.
- IP address or address range of tunnel-facing *ingress* interfaces on the de-encapsulator (the egress PE router).
- Encapsulation protocol (GRE).

Terminology at the Egress PE Router

In filter-based tunneling across IPv4 networks, an egress PE router is described in the following terms:

de-encapsulator—A PE router that receives GRE packets routed through a filter-based GRE tunnel, removes the transport protocol header and GRE header, and forwards the resulting payload protocol packets to the destination network CE router. The de-encapsulator node is also known as a *de-encapsulating tunnel endpoint* or the *tunnel destination*.

de-encapsulating interfaces—On the de-encapsulator, any Ethernet logical interface or aggregated Ethernet interface configured on any core-facing ingress interface that can receive GRE packets from a GRE tunnel. The underlying physical interface must be hosted on a MIC or an MPC. For more information, see "Interfaces That Support Filter-Based Tunneling Across IPv4 Networks" on page 881.

de-encapsulation filter—On the de-encapsulator, a firewall filter that causes the Packet Forwarding Engine to de-encapsulate matched GRE packets and then forward the original passenger protocol packets to destination network CE routers.

GRE packets transported through a single GRE tunnel can arrive at the de-encapsulator node on any of multiple ingress interfaces, depending on how routing is configured. Therefore, you must apply the de-encapsulation firewall filter to the input of every core-facing interface that is an advertised address for the de-encapsulator.

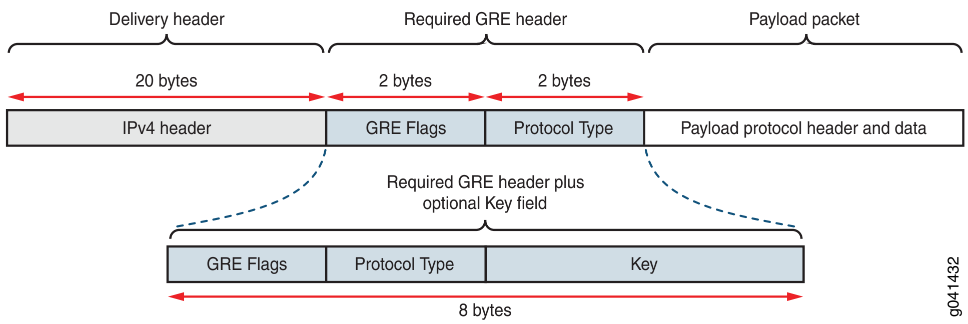
GRE Protocol Format for Filter-Based Tunneling Across IPv4 Networks

In filter-based tunneling across IPv4 networks, the encapsulating interface is an *RFC 1701-compliant transmitter* and the de-encapsulating interfaces are *RFC 1701-compliant receivers*. The packet encapsulation structure implemented in this feature uses a GRE header format that complies with informational RFC 1701, *Generic Routing Encapsulation (GRE)*, October 1994, and with standards track RFC 2784, *Generic Routing Encapsulation (GRE)*, March 2000.

Packet Encapsulation Structure

Filter-based tunneling encapsulates the original passenger protocol packet in an outer shell. For filter-based tunneling across IPv4 networks, the shell adds 24 bytes or 28 bytes of overhead, including 20 bytes of IPv4 header. Figure 55 on page 887 shows the structure of a passenger protocol packet (the GRE payload) with a GRE header and IPv4 header attached.

Figure 55: Encapsulation Structure for Filter-Based Tunneling Across an IPv4 Network

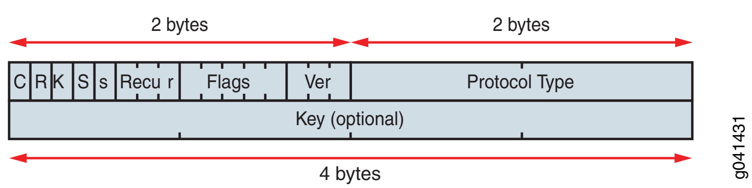


As specified in RFC 1701, five GRE flag bits indicate whether a particular GRE header includes any optional fields (Checksum, Offset, Key, Sequence Number, and Routing). Of the five optional fields, filter-based GRE IPv4 tunneling uses the Key field only.

GRE Header Format

Figure 56 on page 887 shows the format of the variable-size GRE header used for filter-based tunneling across IPv4 networks, with bit 0 the most significant bit and bit 15 the least significant bit.

Figure 56: GRE Header Format for Filter-Based Tunneling Across IPv4 Networks



The first two octets encode GRE flags, as described in Table 56 on page 887.

The 2-octet Protocol Type field contains the value 0x0800 to specify the EtherType value for the IPv4 protocol.

The 4-octet Key field is included only if the Key Present bit is set to 1. The Key field carries the key value of the tunnel defined on the encapsulator. If the GRE tunnel definition specifies a key, the Packet Forwarding Engine for the encapsulating endpoint sets the Key Present bit and adds the Key to the GRE header.

Table 56: GRE Flag Values for Filter-Based Tunneling Across IPv4 Networks

Bit Offset and Field Name		Transmitted Value for Filter-Based GRE Tunneling	
0	**C** = Checksum Present	**0**	Checksum field is not used.
1	**R** = Routing Present	**0**	Offset and Routing fields are not used.
2	**K** = Key Present	**0** or **1**	Transmitted as **0** for a keyless tunnel or **1** for a keyed tunnel.

Table 56: GRE Flag Values for Filter-Based Tunneling Across IPv4 Networks *(continued)*

Bit Offset and Field Name		Transmitted Value for Filter-Based GRE Tunneling	
3	**S** = Sequence Number Present	**0**	Sequence Number field is not used.
4	**s** = Strict Source Route	**0**	Not all routing information is Strict Source Routes.
5 - 7	**Recur** = Recursion Control information	**000**	No additional encapsulations are permitted.
8 - 12	**Flags** = Flag bits	**00000**	Reserved.
13 - 15	**Ver** = Version number	**000**	Reserved.

When the Packet Forwarding Engine performs encapsulation for a keyed GRE IPv4 tunnel, the process constructs the first two octets of the GRE header as 0x0000. When the Packet Forwarding Engine performs encapsulation for a non-keyed GRE IPv4 tunnel, the process constructs the first two octets of the GRE header as 0x2000.

Related Documentation

- Understanding Filter-Based Tunneling Across IPv4 Networks on page 875
- Interfaces That Support Filter-Based Tunneling Across IPv4 Networks on page 881
- Firewall Filter Terminating Actions on page 680
- **tunnel-end-point on page 1269**
- Example: Transporting IPv6 Traffic Across IPv4 Using Filter-Based Tunneling on page 888

Example: Transporting IPv6 Traffic Across IPv4 Using Filter-Based Tunneling

This example shows how to configure a unidirectional generic routing encapsulation (GRE) tunnel to transport IPv6 unicast transit traffic across an IPv4 transport network. To provide network connectivity to the two disjoint IPv6 networks, two MX Series 3D Universal Edge Routers are configured with interfaces that can originate and understand both IPv4 and IPv6 packets. The configuration does not require the creation of tunnel interfaces on Tunnel Services physical interface cards (PICs) or on MPC3E Modular Port Concentrators (MPCs). Instead, you attach firewall filters to Ethernet logical interfaces hosted on Modular Interface Cards (MICs) or MPCs in the two MX Series routers.

NOTE: **Filter-based GRE tunneling is supported on PTX Series routers only when network services is set to enhanced-mode. For more information, see enhanced-mode.**

Requirements

This example uses the following Juniper Networks hardware and Junos OS software:

- Transport network—An IPv4 network running Junos OS Release 12.3R2 or later.
- PE routers—Two MX80 routers installed as provider edge (PE) routers that connect the IPv4 network to two disjoint IPv6 networks that require a logical path from one network to the other.
- Encapsulating interface—On the encapsulator (the ingress PE router), one Ethernet logical interface configured on the built-in 10-Gigabit Ethernet MIC.
- De-encapsulating interfaces—On the de-encapsulator (the egress PE router), Ethernet logical interfaces configured on three ports of the built-in 10-Gigabit Ethernet MIC.

Before you begin configuring this example:

1. On each PE router, use the **show chassis fpc pic-status** operational mode command to determine which router line cards support filter-based GRE IPv4 tunneling and then use the **interfaces** configuration statement to configure encapsulating and de-encapsulating interfaces.
   - At PE1, the encapsulator, configure *one encapsulating interface* on a supported line card.
   - At PE2, the de-encapsulator, configure *three de-encapsulating interfaces* on a supported line card.
2. Check that IPv4 routing protocols are enabled across the network to support routing paths from the encapsulator to the de-encapsulator.

   Configure routing information by manually adding static routes to route tables or by configuring static or dynamic route-sharing protocols. For more information, see *Transport and Internet Protocols Feature Guide for Routing Devices*.
3. At PE1, *ping* the PE2 IPv4 loopback address to verify that the de-encapsulator is reachable from the encapsulator.
4. At PE2, *ping* the CE2 router IPv6 loopback address to verify that the destination customer edge router is reachable from the de-encapsulator..

   IPv6 routing paths from PE2 to CE2 can be provided by static routes manually added to routing tables or by static or dynamic route-sharing protocols.
   - By default, PE2 forwards packets based on interface routes (direct routes) imported from the primary routing table.
   - As an option, the de-encapsulating filter can specify that the Packet Forwarding Engine uses an alternate routing table to forward payload packets to the destination customer network. In an optional configuration task in this example, you specify an alternate routing table by installing static routes from PE2 to C1 in the routing instance **blue**. You configure the routing information base (RIB) group **blue_group**

to specify that the route information of **inet6.0** is shared with **blue.inet6.0**, then you associate the PE2 interfaces with routes stored in both the default routes and the routing instance.

Overview

In this example you configure a unidirectional filter-based GRE IPv4 tunnel from Router PE1 to Router PE2, providing a logical path from IPv6 network C1 to IPv6 network C2.

NOTE: To enable *bidirectional* filter-based GRE tunneling, you must configure a second tunnel in the reverse direction.

As an optional task in this example, you can create a RIB group, which specifies the sharing of routing information (including routes learned from peers, local routes resulting from the application of protocol policies to the learned routes, and the routes advertised to peers) of multiple routing tables.

Topology

Figure 57 on page 890 shows the path of IPv6 traffic transported from network C1 to network C2, across an IPv4 transport network using a filter-based tunnel from PE1 to PE2 and without requiring tunnel interfaces.

Figure 57: Filter-Based Tunnel from PE1 to PE2 in an IPv4 Network

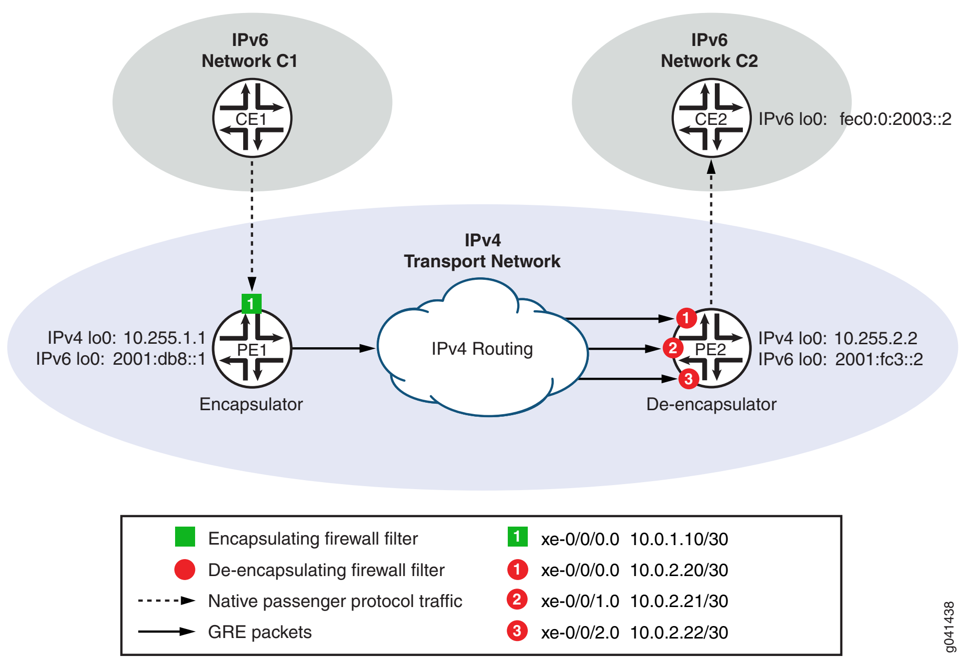


Table 57 on page 891 summarizes the configuration of Router PE1 as the encapsulator. Table 58 on page 891 summarizes the configuration of Router PE2 as the de-encapsulator.

Table 57: Encapsulator Components on PE1

Component	CLI Names		Description
Encapsulator	Device name: IPv4 loopback: IPv6 loopback:	**PE1** **10.255.1.1** **2001:db8::1**	MX80 router installed as an ingress PE router. PE1 connects the IPv4 network the customer edge router CE1 in the IPv6 source network C1.
Encapsulating interface	Interface name: IPv4 address: IPv6 address:	**xe-0/0/0.0** **10.0.1.10/30** **::10.34.1.10/120**	Customer-facing logical interface hosted on a 10-Gigabit Ethernet MIC. CE1 sends this interface IPv6 traffic that originates at end-user hosts and is destined for applications or hosts on the IPv6 destination network C2.
Encapsulation filter	Filter name:	**gre_encap_1**	IPv6 firewall filter whose action causes the Packet Forwarding Engine to encapsulate matched packets using the specified tunnel characteristics. Encapsulation consists of adding a GRE header, adding an IPv4 packet header, and then forwarding the resulting GRE packet through the GRE IPv4 tunnel.
Tunnel source interface	Interface name: IPv4 address:	**xe-0/0/2.0** **10.0.1.12**	Core-facing egress interface to the tunnel.
GRE tunnel template	Tunnel name:	**tunnel_1**	Defines the GRE IPv4 tunnel from Router PE1 (**10.255.1.1**) to Router PE2(**10.255.2.2**), using the tunneling protocol supported on IPv4 (**gre**).

Table 58: De-Encapsulator Components on PE2

Component	CLI Names		Description
De-encapsulator	Device name: IPv4 loopback: IPv6 loopback:	**PE2** **10.255.2.2** **2001:fc3::2**	MX80 router installed as an egress PE router to receive GRE packets forwarded from ingress router PE1 across a GRE IPv4 tunnel.
De-encapsulating interfaces	Interface name: IPv4 address: Interface name: IPv4 address: Interface name: IPv4 address:	**xe-0/0/0.0** **10.0.2.24/30** **xe-0/0/1.0** **10.0.2.21/30** **xe-0/0/2.0** **10.0.2.22/30**	Core-facing ingress logical interfaces hosted on 10-Gigabit Ethernet MICs. The interfaces receive GRE packets routed through the GRE IPv4 tunnel from PE1.
De-encapsulation filter	Filter name:	**gre_decap_1**	IPv4 firewall filter that applies the **decapsulate** action to GRE packets. The filter action causes the Packet Forwarding Engine to de-encapsulate matched packets. De-encapsulation consists of removing the outer GRE header and then forwarding the inner IPv6 payload packet to its original destination on the destination IPv6 network by performing destination lookup on the default routing table.
Tunnel egress interface	Interface name: IPv4 address: IPv6 address:	**xe-0/0/3.0** **10.0.2.23/30** **::20.34.2.23/120**	Customer-facing interface though which the router forwards de-encapsulated IPv6 packets to the destination IPv6 network C2.

Configuration

To transport IPv6 packets from CE1 to CE2 across an IPv4 transport network using a filter-based tunnel from PE1 to PE2 and without configuring tunnel interfaces, perform these tasks:

- Configuring PE1 to Encapsulate IPv6 Packets on page 893
- Configuring PE2 to De-Encapsulate GRE Packets on page 895
- Optional: Configuring PE2 with an Alternate Routing Table on page 898

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Configuring PE1 to Encapsulate IPv6 Packets

```
set interfaces lo0 unit 0 family inet address 10.255.1.1
set interfaces lo0 unit 0 family inet6 address 2001:db8::1
set interfaces xe-0/0/0 unit 0 family inet address 10.0.1.10/30
set interfaces xe-0/0/0 unit 0 family inet6 address 2001::10.34.1.10/120
set interfaces xe-0/0/0 unit 0 family inet6 filter input gre_encap_1
set interfaces xe-0/0/2 unit 0 family inet address 10.0.1.12/30
set firewall family inet6 filter gre_encap_1 term t1 then count c_gre_encap_1
set firewall family inet6 filter gre_encap_1 term t1 then encapsulate tunnel_1
set firewall tunnel-end-point tunnel_1 ipv4 source-address 10.255.1.1
set firewall tunnel-end-point tunnel_1 ipv4 destination-address 10.255.2.2
set firewall tunnel-end-point tunnel_1 gre
```

Configuring PE2 to De-Encapsulate GRE Packets

```
set interfaces lo0 unit 0 family inet address 10.255.2.2
set interfaces lo0 unit 0 family inet6 address 2001:fc3::2
set interfaces xe-0/0/0 unit 0 family inet address 10.0.2.20/30
set interfaces xe-0/0/1 unit 0 family inet address 10.0.2.21/30
set interfaces xe-0/0/2 unit 0 family inet address 10.0.2.22/30
set interfaces xe-0/0/3 unit 0 family inet address 10.0.2.23/30
set interfaces xe-0/0/3 unit 0 family inet6 address ::20.34.2.23/120
set forwarding-options family inet filter input gre_decap_1
set firewall family inet filter gre_decap_1 term t1 from source-address
10.255.1.1/32
set firewall family inet filter gre_decap_1 term t1 from destination-address
10.255.2.2/32
set firewall family inet filter gre_decap_1 term t1 then count c_gre_decap_1
set firewall family inet filter gre_decap_1 term t1 then decapsulate gre
```

Optional: Configuring PE2 with an Alternate Routing Table

```
set routing-instances blue instance-type forwarding
set routing-instances blue routing-options rib blue.inet6.0 static route 0::/0
next-hop fec0:0:2003::2
set routing-options passive
set routing-options rib inet6.0
set routing-options rib-groups blue_group import-rib inet6.0
set routing-options rib-groups blue_group import-rib blue.inet6.0
set routing-options interface-routes rib-group inet6 blue_group
```

```
set firewall family inet filter gre_decap_1 term t1 then decapsulate gre
routing-instance blue
```

Configuring PE1 to Encapsulate IPv6 Packets

Step-by-Step Procedure

To configure Router PE1 to encapsulate IPv6 packets arriving from CE1:

1. Configure the router loopback addresses.

   [edit]
   user@PE1# **set interfaces lo0 unit 0 family inet address 10.255.1.1**
   user@PE1# **set interfaces lo0 unit 0 family inet6 address 2001:db8::1**

2. Configure the encapsulating interface IPv4 and IPv6 addresses and attach the encapsulating filter to the IPv6 input.

   [edit]
   user@PE1# **set interfaces xe-0/0/0 unit 0 family inet address 10.0.1.10/30**
   user@PE1# **set interfaces xe-0/0/0 unit 0 family inet6 address ::10.34.1.10/120**
   user@PE1# **set interfaces xe-0/0/0 unit 0 family inet6 filter input gre_encap_1**

3. Configure the core-facing egress interface to the tunnel.

   [edit]
   user@PE2# **set interfaces xe-0/0/2 unit 0 family inet address 10.0.1.12/30**

4. Define an IPv6 firewall filter that causes the Packet Forwarding Engine to encapsulate all packets.

   [edit]
   user@PE1# **set firewall family inet6 filter gre_encap_1 term t1 then count c_gre_encap_1**
   user@PE1# **set firewall family inet6 filter gre_encap_1 term t1 then encapsulate tunnel_1**

   NOTE: The encapsulate firewall filter action is a *terminating* filter action. A filter-terminating action halts all evaluation of a firewall filter for a specific packet. The router performs the specified action, and no additional terms are examined.

5. Define a GRE IPv4 tunnel template named tunnel_1 that specifies the host IP addresses of the one tunnel source interface and three tunnel destination interfaces.

   [edit]
   user@PE1# **set firewall tunnel-end-point tunnel_1 ipv4 source-address 10.255.1.1**
   user@PE1# **set firewall tunnel-end-point tunnel_1 ipv4 destination-address 10.255.2.2**
   user@PE1# **set firewall tunnel-end-point tunnel_1 gre**

NOTE: You can tunnel multiple but distinct flows from 10.0.1.10 (the tunnel source interface on PE1) to 10.0.2.20 – 10.0.2.22 (the de-encapsulating interfaces on PE2) if you use the GRE option key *number* to uniquely identify each tunnel.

6. If you are done configuring the device, commit the configuration.

```
[edit ]
user@PE1# commit
```

Results From configuration mode, confirm your configuration by entering the **show firewall** and **show interfaces** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Router PE1 Confirm the firewall filter and tunnel template on the encapsulator.

```
user@PE2# show firewall
family inet6 {
  filter gre_encap_1 {
    term t1 {
      then {
        count c_gre_encap_1;
        encapsulate tunnel_1;
      }
    }
  }
}
tunnel-end-point tunnel_1 {
  ipv4 {
    source-address 10.255.1.1;
    destination-address 10.255.2.2;
  }
  gre;
}
```

Router PE1 Confirm the interfaces on the encapsulator.

```
user@PE1# show interfaces
lo0 {
  unit 0 {
    family inet {
      address 10.255.1.1;
    }
    family inet6 {
      address 2001:db8::1;
    }
  }
}
xe-0/0/0 {
```



```
    unit 0 {
        family inet {
            address 10.0.1.10/30;
        }
        family inet6 {
            address ::10.34.1.10/120;
            filter input gre_encap_1;
        }
    }
}
xe-0/0/2 {
    unit 0 {
        family inet {
            address 10.0.1.12/30;
        }
    }
}
```

Configuring PE2 to De-Encapsulate GRE Packets

Step-by-Step Procedure

To configure Router PE2 to de-encapsulate GRE packets arriving from the IPv4 tunnel:

1. Configure the router loopback address.

```
[edit]
user@PE2# set interfaces lo0 unit 0 family inet address 10.255.2.2
user@PE2# set interfaces lo0 unit 0 family inet6 address 2001:fc3::2
```

2. Configure the de-encapsulating interfaces.

```
[edit]
user@PE2# set interfaces xe-0/0/0 unit 0 family inet address 10.0.2.20/30
user@PE2# set interfaces xe-0/0/1 unit 0 family inet address 10.0.2.21/30
user@PE2# set interfaces xe-0/0/2 unit 0 family inet address 10.0.2.22/30
```

3. Configure the customer-facing egress interface to CE2.

```
[edit]
user@PE2# set interfaces xe-0/0/3 unit 0 family inet address 10.0.2.23/30
user@PE2# set interfaces xe-0/0/3 unit 0 family inet6 address ::20.34.2.23/120
```

4. Apply the ingress de-encapsulating firewall filter to all forwarded packets.

```
[edit]
user@PE2# set forwarding-options family inet filter input gre_decap_1
```

5. Define IPv4 filter **gre_decap_1**.

   Define an IPv4 filter that de-encapsulates and forwards all GRE packets.

```
[edit]
user@PE2# set firewall family inet filter gre_decap_1
```

6. Configure term **t1** to match packets transported across the tunnel **tunnel_1** defined on Router PE1. The tunnel sends packets from Router PE1 (configured with IPv4 loopback address 10.255.1.1) to Router PE2 (configured with IPv4 loopback address 10.255.2.2).

```
[edit firewall family inet filter gre_decap_1]
user@PE2# set term t1 from source-address 10.255.1.1
user@PE2# set term t1 from destination-address 10.255.2.2
```

7. Configure term **t1** to count and de-encapsulate matched packets.

```
[edit firewall family inet filter gre_decap_1]
user@PE2# set term t1 then count c_gre_decap_1
user@PE2# set term t1 then decapsulate gre
```

   If the de-encapsulating filter action **decapsulate** references the **blue** routing instance, make sure that the routing instance is configured and that the RIB group **blue_group** defines the sharing of the alternate routes into the primary table.

8. If you are done configuring the device, commit the configuration.

```
[edit]
user@PE2# commit
```

Results From configuration mode, confirm your configuration by entering the **show firewall**, **show forwarding-options**, and **show interfaces** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Router PE2 Confirm the firewall filter on the de-encapsulator.

```
user@PE2# show firewall
family inet {
  filter gre_decap_1 {
    term t1 {
      from {
        source-address 10.255.1.1;
        destination-address 10.255.2.2;
      }
      then {
        count c_gre_decap_1;
        decapsulate gre routing-instance blue;
      }
    }
  }
}
```

NOTE: If the de-encapsulating filter action **decapsulate** references the **blue** routing instance, make sure that the routing instance is configured and that the RIB group **blue_group** defines the sharing of the alternate routes into the primary table.

Router PE2 Confirm the forwarding options (for attaching the de-encapsulating firewall filter to all input forwarded packets) on the de-encapsulator.

```
user@PE2# show forwarding-options
forwarding-options {
    family inet {
        filter {
            input gre_decap_1;
        }
    }
}
```

Router PE2 Confirm the interfaces on the de-encapsulator.

```
user@PE2# show interfaces
lo0 {
    unit 0 {
        family inet {
            address 10.255.2.2;
        }
        family inet6 {
            address 2001:fc3::2;
        }
    }
}
xe-0/0/0 {
    unit 0 {
        family inet {
            address 10.0.2.20/30;
            filter input gre_decap_1;
        }
    }
}
xe-0/0/1 {
    unit 0 {
        family inet {
            address 10.0.2.21/30;
            filter input gre_decap_1;
        }
    }
}
xe-0/0/2 {
    unit 0 {
        family inet {
            address 10.0.2.22/30;
            filter input gre_decap_1;
        }
    }
}
xe-0/0/3 {
    unit 0 {
        family inet {
            address 10.0.2.23/30;
        }
        family inet6 {
```



```
            address ::20.34.2.23/120;
          }
        }
      }
```

Optional: Configuring PE2 with an Alternate Routing Table

Step-by-Step Procedure

To configure Router PE2 with an alternate routing table:

1. Configure the routing instance **blue**, and add static routes to CE2.

   ```
   [edit ]
   user@PE2# set routing-instances blue instance-type forwarding
   user@PE2# set routing-instances blue routing-options rib blue.inet6.0 static route
     0::/0 next-hop fec0:0:2003::2
   ```

   The Junos OS software generates the routing table **blue.inet6.0** using the routing information learned within the instance.

2. Enable routes to remain in routing and forwarding tables, even if the routes become inactive. This allows a static route to remain in the table if the next hop is unavailable.

   ```
   [edit ]
   user@PE2# set routing-options passive
   ```

3. Create a RIB group by explicitly creating the default routing table.

   ```
   [edit ]
   user@PE2# set routing-options rib inet6.0
   ```

4. Define the RIB group **blue_group**.

   ```
   [edit ]
   user@PE2# set routing-options rib-groups blue_group import-rib inet6.0
   user@PE2# set routing-options rib-groups blue_group import-rib blue.inet6.0
   ```

   In the **import-rib** statement, specify the primary routing table first.

5. Associate the router interfaces with routing information specified by the RIB group.

   ```
   [edit ]
   user@PE2# set routing-options interface-routes rib-group inet6 blue_group
   ```

6. If you are done configuring the device, commit the configuration.

   ```
   [edit ]
   user@PE2# commit
   ```

Results

From configuration mode, confirm your configuration by entering the **show firewall**, **show routing-instances**, and **show routing-options**commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Router PE2 If you configured an alternate routing table on Router PE2, confirm the routing instance configuration.

```
user@PE2# show routing-instances
blue {
  instance-type forwarding;
  routing-options {
    static route 0::/0 next-hop fec0:0:2003::2;
  }
}
```

Router PE2 If you configured an alternate routing table on Router PE2, confirm the RIB group and direct routing configurations.

```
user@PE2# show routing-options
interface-routes {
  rib-group blue_group;
}
passive;
rib inet6.0;
rib-groups {
  blue_group {
    import-rib [ inet6.0 blue.inet6.0 ];
  }
}
```

Verification

Confirm that the configurations are working properly.

Verifying Routing Information

Purpose Verify that the direct routes include the alternate routing table information.

Action To perform the verification:

1. (Optional) To verify the routing instance **blue** on PE2, use the **show route instance** operational mode command to display the primary table and number of routes for that routing instance.

```
user@PE2> show route instance blue summary
Instance             Type
         Primary RIB                                     Active/holddown/hidden
blue                 forwarding
         blue.inet6.0                                      2/0/0
```

2. (Optional) To view the routing table associated with the routing instance **blue** on PE2, use the **show route table** operational mode command

```
user@PE2> show route table blue.inet6.0

blue.inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::192:168:239:17/128
                   *[Direct/0] 00:02:26
                    > via lo0.0
fe80::2a0:a50f:fc64:e032/128
                   *[Direct/0] 00:02:26
                    > via lo0.0
```

3. (Optional) To verify that the alternate routes from routing instance blue have been imported to the PE2 forwarding table, use the **show route forwarding-table** operational mode command to display the contents of the router forwarding table and the routing instance forwarding table.

```
user@PE2> show route forwarding-table blue
Routing table: blue.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   689     1
0.0.0.0/32         perm     0                    dscd   687     1
172.16.233.0/4        perm     0                    mdsc   688     1
172.16.233.1/32       perm     0 172.16.233.1          mcst   684     1
255.255.255.255/32 perm     0                    bcst   685     1

Routing table: blue.iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   695     1

Routing table: blue.inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   701     1
::/128             perm     0                    dscd   699     1
2001:db8::192:168:239:17/128
                   user     0                    rtbl     2     3
fe80::2a0:a50f:fc64:e032/128
                   user     0                    rtbl     2     3
ff00::/8           perm     0                    mdsc   700     1
ff02::1/128        perm     0 ff02::1            mcst   697     1
```

Verifying Encapsulation on PE1

Purpose Verify the encapsulating interface on PE1.

Action To perform the verification:

1. Use the **show interfaces filters** operational mode command to verify that the encapsulating firewall filter is attached to the ingress of the encapsulating interface.


```
user@PE1> show interfaces filters xe-0/0/0.0
Interface        Admin  Link  Proto  Input Filter        Output Filter
xe-0/0/0.0       up     down  inet6  gre_encap_1
```

2. Use the **show interfaces** operational mode command to verify that the encapsulating interface is receiving packets.

```
user@PE1> show interfaces xe-0/0/0.0 detail | filter "Ingress traffic"
...
Physical interface: xe-0/0/0, Enabled, Physical link is Up
...
  Ingress traffic statistics at Packet Forwarding Engine:
   Input  bytes  :           6970299398                    0 bps
   Input  packets:             81049992                    0 pps
   Drop   bytes  :                    0                    0 bps
   Drop   packets:                    0                    0 pps
...
```

3. Use the **show firewall filter** operational mode command to verify that ingress passenger protocol traffic triggers the encapsulating filter.

```
user@PE1> show firewall filter gre_encap_1
Filter: gre_encap_1
Counters:
Name                                Bytes              Packets
c_gre_encap_1                  6970299398             81049992
```

Meaning If the encapsulating filter is attached to the encapsulating interface, and the encapsulating interface receives passenger protocol traffic, and the firewall filter statistics show that ingress passenger protocol traffic is being encapsulated, then GRE packets are being forwarded through the tunnel.

Verifying De-Encapsulation on PE2

Purpose Verify the de-encapsulating interfaces on PE2.

Action To perform the verification:

1. On PE1, use the **ping** operational mode command to verify that PE2 is reachable.

```
user@PE1> ping 10.255.2.2
PING 10.255.2.2 (10.255.2.2): 56 data bytes
64 bytes from 10.255.2.2: icmp_seq=0 ttl=64 time=0.576 ms
64 bytes from 10.255.2.2: icmp_seq=1 ttl=64 time=0.269 ms
^C [abort]
```

2. On PE2, use the **show interfaces filter** operational mode command to verify that the de-encapsulating firewall filter is attached to the ingress of the de-encapsulating interfaces.

```
user@PE2> show interfaces filter | match xe-
Interface        Admin  Link  Proto  Input Filter        Output Filter
xe-0/0/0.0       up     down  inet   gre_decap_1
```



```
xe-0/0/1.0        up      down   inet    gre_decap_1
xe-0/0/2.0        up      down   inet    gre_decap_1
```

3. On PE2, use the **show interfaces** operational mode command to verify that the de-encapsulating interfaces are receiving packets.

```
user@PE2> show interfaces xe-0/0/0.0 detail | filter "Ingress traffic"
Physical interface: xe-0/0/0, Enabled, Physical link is Up
 ...
  Ingress traffic statistics at Packet Forwarding Engine:
   Input  bytes   :          6970299398                    0 bps
   Input  packets:             81049992                    0 pps
   Drop   bytes   :                   0                    0 bps
   Drop   packets:                    0                    0 pps
 ...
```

```
user@PE2> show interfaces xe-0/0/1.0 detail | filter "Ingress traffic"
Physical interface: xe-0/0/2, Enabled, Physical link is Up
 ...
```

```
user@PE2> show interfaces xe-0/0/2.0 detail | filter "Ingress traffic"
Physical interface: xe-0/0/2, Enabled, Physical link is Up
 ...
```

Depending on how routing is configured and which links are up and which links are down, some of the de-encapsulating interfaces might not be receiving packets although the tunnel is operating properly.

4. On PE2, use the **show firewall filter** operational mode command to verify that ingress GRE traffic triggers the de-encapsulating filter.

```
user@PE2> show firewall filter gre_decap_1

Filter: gre_decap_1
Counters:
Name                                                 Bytes              Packets
c_gre_decap_1                                   6970299398             81049992
```

Meaning The verification confirms the following operational states and activities of the encapsulator:

- PE2 is reachable from the PE1.
- The de-encapsulating filter is attached to the input of all de-encapsulating interfaces.
- The de-encapsulator is receiving traffic at de-encapsulating interfaces as expected.
- GRE packets received at the de-encapsulating interfaces trigger the de-encapsulating firewall filter action.

Related Documentation

- Understanding Filter-Based Tunneling Across IPv4 Networks on page 875
- Interfaces That Support Filter-Based Tunneling Across IPv4 Networks on page 881

- Components of Filter-Based Tunneling Across IPv4 Networks on page 883
- Firewall Filter Terminating Actions on page 680
- **tunnel-end-point on page 1269**
- **clear firewall on page 1589**
- *show chassis fpc*
- **show firewall on page 1591**
- **show firewall log on page 1600**
- *show interfaces (10-Gigabit Ethernet)*
- *show interfaces (Aggregated Ethernet)*
- *show interfaces (Gigabit Ethernet)*
- **show route forwarding-table on page 1465**
- *Junos OS Support for IPv4 Routing Protocols*
- *Junos OS Support for IPv6 Routing Protocols*

CHAPTER 23

Configuring Service Filters

Service Filter Overview

This topic covers the following information:

Services

The Adaptive Services Physical Interface Cards (PICs), Multiservices PICs, and Multiservices Dense Port Concentrators (DPCs) provide *adaptive services interfaces*. Adaptive services interfaces enable you to coordinate a special range of services on a single PIC or DPC by configuring a set of services and applications.

NOTE: Service filters are not supported on T4000 routers.

Service Rules

A *service set* is an optional definition you can apply to the traffic at an adaptive services interface. A service set enables you to configure combinations of directional rules and default settings that control the behavior of each service in the service set.

Service Rule Refinement

When you apply a service set to the traffic at an adaptive services interface, you can optionally use *service filters* to refine the target of the set of services and also to process traffic. Service filters enable you to manipulate traffic by performing packet filtering to a defined set of services on an adaptive services interface before the traffic is delivered to its destination. You can apply a service filter to traffic before packets are accepted for input or output service processing or after packets return from input service processing.

Service Filter Counters

Like standard firewall filters, service filters support counting of matched packets. When you display counters for a service filter, however, the syntax for specifying the filter name includes the name of the *service set* to which the service filter is applied.

- To enable counting of the packets matched by a service filter term, specify the **count *counter-name*** nonterminating action in that term.
- To display counters for service filters, use the show firewall **filter *filter-name* <counter *counter-name*>** operational mode command, and specify the ***filter-name*** as follows:

  __service-*service-set-name*:*service-filter-name*

For example, suppose you configure a service filter named **out_filter** with a counter named **out_counter** and apply that service filter to a logical interface to direct certain packets for processing by the output services associated with the service set **nat_set**. In this scenario, the syntax for using the **show firewall** operational mode command to display the counter is as follows:

```
[edit]
user@host> show firewall filter __service-nat_set:out_filter counter out_counter
```

Related Documentation

- Stateless Firewall Filter Types on page 560
- How Service Filters Evaluate Packets on page 906
- Guidelines for Configuring Service Filters on page 908
- Guidelines for Applying Service Filters on page 910
- Example: Configuring and Applying Service Filters on page 913
- *Adaptive Services Overview*
- *Configuring Service Sets to be Applied to Services Interfaces*
- *Configuring Service Rules*

How Service Filters Evaluate Packets

This topic covers the following information:

- Service Filters That Contain a Single Term on page 907
- Service Filters That Contain Multiple Terms on page 907

Service Filters That Contain a Single Term

For a service filter that consists of a single term, the policy framework software evaluates a packet as follows:

- If the packet matches all the conditions, the actions are taken.
- If the packet matches all the conditions and no actions are specified, the packet is accepted.
- If the packet does not match all the conditions, it is discarded.

Service Filters That Contain Multiple Terms

For a service filter that consists of multiple terms, the policy framework software evaluates a packet against the terms in the filter sequentially, beginning with the first term in the filter, until either the packet matches all the conditions in one of the terms or there are no more terms in the filter.

- If the packet matches all the conditions in a term, the actions in that term are performed and evaluation of the packet ends at that term. Any subsequent terms in the filter are not used.
- If the packet does not match all the conditions in the term, evaluation of the packet proceeds to the next term in the filter.

Service Filter Terms That Do Not Contain Any Match Conditions

For service filters with a single term and for filters with multiple terms, if a term does not contain any match conditions, the actions are taken on any packet evaluated.

Service Filter Terms That Do Not Contain Any Actions

If a term does not contain any actions, and if the packet matches the conditions in the term, the packet is accepted.

Service Filter Default Action

Each service filter has an *implicit* **skip** action at the end of the filter, which is equivalent to including the following example term **explicit_skip** as the final term in the service filter:

```
term explicit_skip {
  then skip;
}
```

By default, if a packet matches none of the terms in a service filter, the packet bypasses service processing.

Related Documentation

- Service Filter Overview on page 905

Guidelines for Configuring Service Filters

This topic covers the following information:

Statement Hierarchy for Configuring Service Filters

To configure a service filter, include the **service-filter *service-filter-name*** statement at the **[edit firewall family (inet | inet6)]** hierarchy level:

```
[edit]
firewall {
  family (inet | inet6) {
    service-filter service-filter-name {
      term term-name {
        from {
          match-conditions;
        }
        then {
          actions;
        }
      }
    }
  }
}
```

Individual statements supported under the **service-filter *service-filter-name*** statement are described separately in this topic and are illustrated in the example of configuring and applying a service filter.

Service Filter Protocol Families

You can configure service filters to filter IPv4 traffic (**family inet**) and IPv6 traffic (**family inet6**) only. No other protocol families are supported for service filters.

Service Filter Names

Under the **family inet** or **family inet6** statement, you can include **service-filter *service-filter-name*** statements to create and name service filters. The filter name can contain letters, numbers, and hyphens (-) and be up to 64 characters long. To include spaces in the name, enclose the entire name in quotation marks (“ ”).

Service Filter Terms

Under the **service-filter** ***service-filter-name*** statement, you can include **term** ***term-name*** statements to create and name filter terms.

- You must configure at least one term in a firewall filter.
- You must specify a unique name for each term within a firewall filter. The term name can contain letters, numbers, and hyphens (-) and can be up to 64 characters long. To include spaces in the name, enclose the entire name in quotation marks (“ ”).
- The order in which you specify terms within a firewall filter configuration is important. Firewall filter terms are evaluated in the order in which they are configured. By default, new terms are always added to the end of the existing filter. You can use the **insert** configuration mode command to reorder the terms of a firewall filter.

Service Filter Match Conditions

Service filter terms support only a subset of the IPv4 and IPv6 match conditions that are supported for standard stateless firewall filters.

If you specify an IPv6 address in a match condition (the **address**, **destination-address**, or **source-address** match conditions), use the syntax for text representations described in RFC 4291, *IP Version 6 Addressing Architecture*. For more information about IPv6 addresses, see “*IPv6 Overview*” in the *Junos OS Routing Protocols Library*.

Service Filter Terminating Actions

When configuring a service filter term, you must specify one of the following filter-terminating actions:

- **service**
- **skip**

NOTE: **These actions are unique to service filters.**

Service filter terms support only a subset of the IPv4 and IPv6 nonterminating actions that are supported for standard stateless firewall filters:

- **count** ***counter-name***
- **log**
- **port-mirror**
- **sample**

Service filters do not support the **next** action.

Related Documentation

- Service Filter Overview on page 905
- How Service Filters Evaluate Packets on page 906
- Guidelines for Applying Service Filters on page 910
- Service Filter Match Conditions for IPv4 or IPv6 Traffic on page 918
- Service Filter Terminating Actions on page 925
- Service Filter Nonterminating Actions on page 924
- Example: Configuring and Applying Service Filters on page 913

Guidelines for Applying Service Filters

This topic covers the following information:

Restrictions for Adaptive Services Interfaces

The following restrictions apply to adaptive services interfaces and service filters.

Adaptive Services Interfaces

You can apply a service filter to IPv4 or IPv6 traffic associated with a service set at an *adaptive services interface* only. Adaptive services interfaces are supported for the following hardware only:

- Adaptive Services (AS) PICs on M Series and T Series routers
- Multiservices (MS) PICs on M Series and T Series routers
- Multiservices (MS) DPCs on MX Series routers (and EX Series switches)

System Logging to a Remote Host from M Series Routers

Logging of adaptive services interfaces messages to an external server by means of the **fxp0** or **em0** port is not supported on M Series routers. The architecture does not support system logging traffic out of a management interface. Instead, access to an external server is supported on a Packet Forwarding Engine interface.

Statement Hierarchy for Applying Service Filters

You can enable packet filtering of IPv4 or IPv6 traffic before a packet is accepted for input or output service processing. To do this, apply a service filter to the adaptive services interface input or output in conjunction with an interface service set.

You can also enable packet filtering of IPv4 or IPv6 traffic that is returning to the Packet Forwarding Engine after input service processing completes. To do this, apply a post-service filter to the adaptive services interface input.

The following configuration shows the hierarchy levels at which you can apply the service filters to adaptive services interfaces:

```
[edit]
interfaces {
  interface-name {
    unit unit-number {
      family (inet | inet6) {
        service {
          input {
            service-set service-set-name service-filter service-filter-name;
            post-service-filter service-filter-name;
          }
          output {
            service-set service-set-name service-filter service-filter-name;
          }
        }
      }
    }
  }
}
```

Associating Service Rules with Adaptive Services Interfaces

To define and group the service rules be applied to an adaptive services interface, you define an *interface service set* by including the **service-set *service-set-name*** statement at the **[edit services]** hierarchy level.

To apply an interface service set to the input and output of an adaptive services interface, you include the **service-set *service-set-name*** at the following hierarchy levels:

- **[edit interfaces *interface-name* unit *unit-number* input]**
- **[edit interfaces *interface-name* unit *unit-number* output]**

If you apply a service set to one direction of an adaptive services interface but do not apply a service set to the other direction, an error occurs when you commit the configuration.

The adaptive services PIC performs different actions depending on whether the packet is sent to the PIC for input service or for output service. For example, you can configure a single service set to perform Network Address Translation (NAT) in one direction and destination NAT (dNAT) in the other direction.

Filtering Traffic Before Accepting Packets for Service Processing

To filter IPv4 or IPv6 traffic before accepting packets for input or output service processing, include the **service-set *service-set-name* service-filter *service-filter-name*** at one of the following interfaces:

- **[edit interfaces *interface-name* unit *unit-number* family (inet | inet6) service input]**

- **[edit interfaces *interface-name* unit *unit-number* family (inet | inet6) service output]**

For the ***service-set-name***, specify a service set configured at the **[edit services service-set]** hierarchy level.

The service set retains the input interface information even after services are applied, so that functions such as filter-class forwarding and destination class usage (DCU) that depend on input interface information continue to work.

The following requirements apply to filtering inbound or outbound traffic before accepting packets for service processing:

- You configure the same service set on the input and output sides of the interface.
- If you include the **service-set** statement without an optional **service-filter** definition, the Junos OS assumes the match condition is true and selects the service set for processing automatically.
- The service filter is applied only if a service set is configured and selected.

You can include more than one service set definition on each side of an interface. The following guidelines apply:

- If you include multiple service sets, the router (or switch) software evaluates them in the order in which they appear in the configuration. The system executes the first service set for which it finds a match in the service filter and ignores the subsequent definitions.
- A maximum of six service sets can be applied to an interface.
- When you apply multiple service sets to an interface, you must also configure and apply a service filter to the interface.

Postservice Filtering of Returning Service Traffic

As an option to filtering of IPv4 or IPv6 input service traffic, you can apply a service filter to IPv4 or IPv6 traffic that is returning to the services interface after the service set is executed. To apply a service filter in this manner, include the **post-service-filter** ***service-filter-name*** statement at the **[edit interfaces *interface-name* unit *unit-number* family (inet | inet6) service input]** hierarchy level.

Related Documentation

- Service Filter Overview on page 905
- How Service Filters Evaluate Packets on page 906
- Guidelines for Configuring Service Filters on page 908
- Example: Configuring and Applying Service Filters on page 913

- *Adaptive Services Overview*
- *Configuring Service Sets to be Applied to Services Interfaces*
- *Configuring Service Rules*

Example: Configuring and Applying Service Filters

This example shows how to configure and apply service filters.

Requirements

This example use the logical interface **xe-0/1/0.0** on any of the following hardware components:

- Adaptive Services (AS) PIC on an M Series or T Series router
- Multiservices (MS) PIC on an M Series or T Series router
- Multiservices (MS) DPC on an MX Series router
- EX Series switch

Before you begin, make sure that you have:

- Installed your supported router (or switch) and PICs or DPCs and performed the initial router (or switch) configuration.
- Configured basic Ethernet in the topology, and verified that traffic is flowing in the topology and that IPv4 traffic is flowing through logical interface **xe-0/1/0.0**.
- Configured the service set **vrf_svcs** with service input and output rules and default settings for services at a service interface.

For guidelines for configuring service sets, see *Configuring Service Sets to be Applied to Services Interfaces*.

Overview

In this example, you create three types of service filters for IPv4 traffic: one input service filter, one postservice input filter, and one output service filter.

Topology

You apply the input service filter and postservice input filter to input traffic at logical interface **xe-0/1/0.0**, and you apply the output service filter to the output traffic at the same logical interface.

- Filtering IPv4 traffic before it is accepted for input service processing—At logical interface **xe-0/1/0.0**, you use the service filter **in_filter_presvc** to filter IPv4 input traffic before the traffic can be accepted for processing by services associated with service set **vrf_svcs**. The **in_filter_presvc** service filter counts packets sent from ICMP port 179, directs these packets to the input services associated with the service set **vrf_svcs**, and discards all other packets.

- Filtering IPv4 traffic after it has completed input service processing—At logical interface **xe-0/1/0.0**, you use the service filter **in_filter_postsvc** to filter traffic that is returning to the services interface after the input service set **in_filter_presvc** is executed. The **in_filter_postsvc** service filter counts packets sent from ICMP port 179 and then discards them.
- Filtering IPv4 traffic before it is accepted for output service processing—At logical interface **xe-0/1/0.0**, you use the service-filter **out_filter_presvc** to filter IPv4 output traffic before the traffic can be accepted for processing by the services associated with service set **vrf_svcs**. The **out_filter_presvc** service filter counts packets destined for TCP port 179 and then directs the packets to the output services associated with the service set **vrf_svcs**.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring the Three Service Filters on page 914
- Applying the Three Service Filters on page 916

CLI Quick Configuration

To quickly configure this example, copy the following commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet service-filter in_filter_presvc term t1 from protocol tcp
set firewall family inet service-filter in_filter_presvc term t1 from source-port bgp
set firewall family inet service-filter in_filter_presvc term t1 then count svc_in_pkts
set firewall family inet service-filter in_filter_presvc term t1 then service
set firewall family inet service-filter in_filter_postsvc term t2 from protocol tcp
set firewall family inet service-filter in_filter_postsvc term t2 from source-port bgp
set firewall family inet service-filter in_filter_postsvc term t2 then count svc_in_pkts_rtn
set firewall family inet service-filter in_filter_postsvc term t2 then skip
set firewall family inet service-filter out_filter_presvc term t3 from protocol icmp
set firewall family inet service-filter out_filter_presvc term t3 from destination-port bgp
set firewall family inet service-filter out_filter_presvc term t3 then count svc_out_pkts
set firewall family inet service-filter out_filter_presvc term t3 then service
set interfaces xe-0/1/0 unit 0 family inet service input service-set vrf_svcs service-filter
  in_filter_presvc
set interfaces xe-0/1/0 unit 0 family inet service input post-service-filter in_filter_postsvc
set interfaces xe-0/1/0 unit 0 family inet service output service-set vrf_svcs service-filter
  out_filter_presvc
```

Configuring the Three Service Filters

Step-by-Step Procedure

To configure the three service filters:

1. Configure the input service filter.

```
[edit]
user@host# edit firewall family inet service-filter in_filter_presvc
```



```
[edit firewall family inet service-filter in_filter_presvc]
user@host# set term t1 from protocol tcp
user@host# set term t1 from source-port bgp
user@host# set term t1 then count svc_in_pkts
user@host# set term t1 then service
```

2. Configure the postservice input filter.

```
[edit]
user@host# edit firewall family inet service-filter in_filter_postsvc

[edit firewall family inet service-filter in_filter_postsvc]
user@host# set term t2 from protocol tcp
user@host# set term t2 from source-port bgp
user@host# set term t2 then count svc_in_pkts_rtn
user@host# set term t2 then skip
```

3. Configure the output service filter.

```
[edit]
user@host# edit firewall family inet service-filter out_filter_presvc

[edit firewall family inet service-filter out_filter_presvc]
user@host# set term t3 from protocol icmp
user@host# set term t3 from destination-port bgp
user@host# set term t3 then count svc_out_pkts
user@host# set term t3 then service
```

Results Confirm the configuration of the input and output service filters and the postservice input filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  service-filter in_filter_presvc {
    term t1 {
      from {
        protocol tcp;
        source-port bgp;
      }
      then {
        count svc_in_pkts;
        service;
      }
    }
  }
  service-filter in_filter_postsvc {
    term t2 {
      from {
        protocol tcp;
```

```
                source-port bgp;
            }
            then {
                count svc_in_pkts_rtn;
                skip;
            }
        }
    }
    service-filter out_filter_presvc {
        term t3 {
            from {
                protocol icmp;
                destination-port bgp;
            }
            then {
                count svc_out_pkts;
                service;
            }
        }
    }
}
```

Applying the Three Service Filters

Step-by-Step Procedure

To apply the three service filters:

1. Access the IPv4 protocol on the input interface **xe-0/1/0.0**.

   ```
   [edit]
   user@host# edit interfaces xe-0/1/0 unit 0 family inet
   ```

2. Apply the input service filter and the postservice input filter.

   ```
   [edit interfaces xe-0/1/0 unit 0 family inet]
   user@host# set service input service-set vrf_svcs service-filter in_filter_presvc
   user@host# set service input post-service-filter in_filter_postsvc
   user@host# set service output service-set vrf_svcs service-filter out_filter_presvc
   ```

Results

Confirm the configuration of the interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
xe-0/1/0 {
    unit 0 {
        family inet {
            service {
                input {
                    service-set vrf_svcs service-filter in_filter_presvc;
                    post-service-filter in_filter_postsvc;
                }
                output {
```



```
                    service-set vrf_svcs service-filter out_filter_presvc;
                }
            }
        }
    }
}
```

When you are done configuring the device, commit your candidate configuration.

Verification

Confirm that the configuration is working properly.

- Verifying That Inbound Traffic Is Filtered Before Input Service on page 917
- Verifying That Inbound Traffic Is Filtered After Input Service Processing on page 917
- Verifying That Outbound Traffic Is Filtered Before Output Service Processing on page 917

Verifying That Inbound Traffic Is Filtered Before Input Service

Purpose Verify that inbound packets sent from TCP port 179 are sent for processing by the *input* services associated with the service set **vrf_svcs**.

Action Display the count of packets sent for processing by the *input* services associated with the service set **vrf_svcs**.

```
[edit]
user@host> show firewall filter in_filter_presvc-vrf_svcs counter svc_in_pkts
```

Verifying That Inbound Traffic Is Filtered After Input Service Processing

Purpose Verify that inbound packets sent from TCP port 179 are returned from processing by the *input* services associated with the service set **vrf_svcs**.

Action Display the count of packets returned from processing by the *input* services associated with the service set **vrf_svcs**.

```
[edit]
user@host> show firewall filter in_filter_postsvc-vrf_svcs counter svc_in_pkts_rtn
```

Verifying That Outbound Traffic Is Filtered Before Output Service Processing

Purpose Verify that outbound packets sent to ICMP port 179 are sent for processing by the *output* services associated with the service set **vrf_svcs**.

Action Display the count of packets sent for processing by the *output* services associated with the service set **vrf_svcs**.

```
[edit]
user@host> show firewall filter out_filter_presvc-vrf_svcs counter svc_out_pkts
```

Related Documentation

- Service Filter Overview on page 905
- How Service Filters Evaluate Packets on page 906
- Guidelines for Configuring Service Filters on page 908
- Guidelines for Applying Service Filters on page 910

Service Filter Match Conditions for IPv4 or IPv6 Traffic

Service filters support only a subset of the stateless firewall filter match conditions for IPv4 and IPv6 traffic. Table 59 on page 918 describes the service filter match conditions.

Table 59: Service Filter Match Conditions for IPv4 or IPv6 Traffic

Match Condition	Description	Protocol Families
address *address*	Match the IP source or destination address field.	• **family inet** • **family inet6**
address *address* except	Do not match the IP source or destination address field.	• **family inet** • **family inet6**
ah-spi *spi-value*	(M Series routers, except M120 and M320) Match on the IPsec authentication header (AH) security parameter index (SPI) value.	• **family inet**
ah-spi-except *spi-value*	(M Series routers, except M120 and M320) Do not match on the IPsec AH SPI value.	• **family inet**
destination-address *address*	Match the IP destination address field. You cannot specify both the **address** and **destination-address** match conditions in the same term.	• **family inet** • **family inet6**
destination-address *address* except	Do not match the IP destination address field. You cannot specify both the **address** and **destination-address** match conditions in the same term.	• **family inet** • **family inet6**

Table 59: Service Filter Match Conditions for IPv4 or IPv6 Traffic *(continued)*

Match Condition	Description	Protocol Families
destination-port *number*	Match the UDP or TCP destination port field. You cannot specify both the **port** and **destination-port** match conditions in the same term. If you configure this match condition for IPv4 traffic, we recommend that you also configure the **protocol udp** or **protocol tcp** match statement in the same term to specify which protocol is being used on the port. If you configure this match condition for IPv6 traffic, we recommend that you also configure the **next-header udp** or **next-header tcp** match condition in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the following text synonyms (the port numbers are also listed): **afs** (1483), **bgp** (179), **biff** (512), **bootpc** (68), **bootps** (67), **cmd** (514), **cvspserver** (2401), **dhcp** (67), **domain** (53), **eklogin** (2105), **ekshell** (2106), **exec** (512), **finger** (79), **ftp** (21), **ftp-data** (20), **http** (80), **https** (443), **ident** (113), **imap** (143), **kerberos-sec** (88), **klogin** (543), **kpasswd** (761), **krb-prop** (754), **krbupdate** (760), **kshell** (544), **ldap** (389), **ldp** (646), **login** (513), **mobileip-agent** (434), **mobilip-mn** (435), **msdp** (639), **netbios-dgm** (138), **netbios-ns** (137), **netbios-ssn** (139), **nfsd** (2049), **nntp** (119), **ntalk** (518), **ntp** (123), **pop3** (110), **pptp** (1723), **printer** (515), **radacct** (1813), **radius** (1812), **rip** (520), **rkinit** (2108), **smtp** (25), **snmp** (161), **snmptrap** (162), **snpp** (444), **socks** (1080), **ssh** (22), **sunrpc** (111), **syslog** (514), **tacacs** (49), **tacacs-ds** (65), **talk** (517), **telnet** (23), **tftp** (69), **timed** (525), **who** (513), or **xdmcp** (177).	• **family inet** • **family inet6**
destination-port-except *number*	Do not match the UDP or TCP destination port field. For details, see the **destination-port** match description.	• **family inet** • **family inet6** **family inet6**
destination-prefix-list *name*	Match the list of destination prefixes. The prefix list is defined at the **[edit policy-options prefix-list** ***prefix-list-name*****]** hierarchy level.	• **family inet** • **family inet6**
esp-spi *value*	Match the IPsec encapsulating security payload (ESP) SPI value. Specify a single value or a range of values. You can specify a *value* in hexadecimal, binary, or decimal form. To specify the value in hexadecimal form, include **0x** as a prefix. To specify the value in binary form, include **b** as a prefix.	• **family inet** • **family inet6**
esp-spi-except *value*	Do not match the IPsec ESP SPI value or range of values. For details, see the **esp-spi** match condition.	• **family inet** • **family inet6**

Table 59: Service Filter Match Conditions for IPv4 or IPv6 Traffic *(continued)*

Match Condition	Description	Protocol Families
first-fragment	Match if the packet is the first fragment of a fragmented packet. Do not match if the packet is a trailing fragment of a fragmented packet. The first fragment of a fragmented packet has a fragment offset value of **0**. This match condition is an alias for the bit-field match condition **fragment-offset 0** match condition. To match both first and trailing fragments, you can use two terms that specify different match conditions: **first-fragment** and **is-fragment**.	• **family inet**
fragment-flags *number*	(Ingress only) Match the three-bit IP fragmentation flags field in the IP header. In place of the numeric field value, you can specify one of the following keywords (the field values are also listed): **dont-fragment** (0x4), **more-fragments** (0x2), or **reserved** (0x8).	• **family inet**
fragment-offset *number*	Match the 13-bit fragment offset field in the IP header. The value is the offset, in 8-byte units, in the overall datagram message to the data fragment. Specify a numeric value, a range of values, or a set of values. An offset value of **0** indicates the first fragment of a fragmented packet. The **first-fragment** match condition is an alias for the **fragment-offset 0** match condition. To match both first and trailing fragments, you can use two terms that specify different match conditions (**first-fragment** and **is-fragment**).	• **family inet**
fragment-offset-except *number*	Do not match the 13-bit fragment offset field.	• **family inet**
interface-group *group-number*	Match the interface group (set of one or more logical interfaces) on which the packet was received. For *group-number*, specify a value from **0** through **255**. For information about configuring interface groups, see "Filtering Packets Received on a Set of Interface Groups Overview" on page 861.	• **family inet** • **family inet6**
interface-group-except *group-number*	Do not match the interface group on which the packet was received. for details, see the **interface-group** match condition.	• **family inet** • **family inet6**

Table 59: Service Filter Match Conditions for IPv4 or IPv6 Traffic *(continued)*

Match Condition	Description	Protocol Families
ip-options *values*	Match the 8-bit IP option field, if present, to the specified value or list of values. In place of a numeric value, you can specify one of the following text synonyms (the option values are also listed): **loose-source-route** (131), **record-route** (7), **router-alert** (148), **security** (130), **stream-id** (136), **strict-source-route** (137), or **timestamp** (68). To match *any* value for the IP option, use the text synonym **any**. To match on *multiple* values, specify the list of values within square brackets ('**[**' and '**]**'). To match a *range* of values, use the value specification **[** ***value1-value2*** **]**. For example, the match condition **ip-options [0-147]** matches on an IP options field that contains the **loose-source-route**, **record-route**, or **security** values, or any other value from 0 through 147. However, this match condition does not match on an IP options field that contains only the **router-alert** value (148). For most interfaces, a filter term that specifies an **ip-option** match on one or more *specific* IP option values (a value other than **any**) causes packets to be sent to the Routing Engine so that the kernel can parse the IP option field in the packet header. • For a firewall filter term that specifies an **ip-option** match on one or more specific IP option values, you cannot specify the **count**, **log**, or **syslog** nonterminating actions *unless* you also specify the **discard** terminating action in the same term. This behavior prevents double-counting of packets for a filter applied to a transit interface on the router (or switch). • Packets processed on the kernel might be dropped in case of a system bottleneck. To ensure that matched packets are instead sent to the Packet Forwarding Engine (where packet processing is implemented in hardware), use the **ip-options any** match condition. The 10-Gigabit Ethernet Modular Port Concentrator (MPC), 60-Gigabit Ethernet MPC, 60-Gigabit Queuing Ethernet MPC, 60-Gigabit Ethernet Enhanced Queuing MPC on MX Series routers and EX Series switches are capable of parsing the IP option field of the IPv4 packet header. This capability is supported on EX Series switches also. For interfaces configured on those MPCs, *all* packets that are matched using the **ip-options** match condition are sent to the Packet Forwarding Engine for processing.	**family inet** • **family inet**
ip-options-except *values*	Do not match the IP option field to the specified value or list of values. For details about specifying the ***values***, see the **ip-options** match condition.	• **family inet**
is-fragment	Match if the packet is a trailing fragment of a fragmented packet. Do not match the first fragment of a fragmented packet. This match condition is an alias for the bit-field match condition **fragment-offset 0 except** bits. NOTE: To match both first and trailing fragments, you can use two terms that specify different match conditions (**first-fragment** and **is-fragment**).	• **family inet**

Table 59: Service Filter Match Conditions for IPv4 or IPv6 Traffic *(continued)*

Match Condition	Description	Protocol Families
port *number*	Match the UDP or TCP source or destination port field. If you configure this match condition, you cannot configure the **destination-port** match condition or the **source-port** match condition in the same term. If you configure this match condition for IPv4 traffic, we recommend that you also configure the **protocol udp** or **protoco tcp** match statement in the same term to specify which protocol is being used on the port. If you configure this match condition for IPv6 traffic, we recommend that you also configure the **next-header udp** or **next-header tcp** match condition in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the text synonyms listed under **destination-port**.	• **family inet** • **family inet6**
port-except *number*	Do not match the UDP or TCP source or destination port field. For details, see the **port** match condition.	• **family inet** • **family inet6**
prefix-list *prefix-list-name*	Match the prefixes of the source or destination address fields to the prefixes in the specified list. The prefix list is defined at the **[edit policy-options prefix-list** *prefix-list-name***]** hierarchy level.	• **family inet** • **family inet6**
protocol *number*	Match the IP protocol type field. In place of the numeric value, you can specify one of the following text synonyms (the field values are also listed): **ah** (51), **dstopts** (60), **egp** (8), **esp** (50), **fragment** (44), **gre** (47), **hop-by-hop** (0), **icmp** (1), **icmp6** (58), **icmpv6** (58), **igmp** (2), **ipip** (4), **ipv6** (41), **ospf** (89), **pim** (103), **rsvp** (46), **sctp** (132), **tcp** (6), **udp** (17), or **vrrp** (112).	• **family inet** —
protocol-except *number*	Do not match the IP protocol type field. For details, see the **protocol** match condition.	• **family inet**
source-address *address*	Match the IP source address. You cannot specify both the **address** and **source-address** match conditions in the same term.	• **family inet** • **family inet6**
source-address *address* **except**	Do not match the IP source address. You cannot specify both the **address** and **source-address** match conditions in the same term.	• **family inet** • **family inet6**

Table 59: Service Filter Match Conditions for IPv4 or IPv6 Traffic *(continued)*

Match Condition	Description	Protocol Families
source-port *number*	Match the UDP or TCP source port field. You cannot specify the **port** and **source-port** match conditions in the same term. If you configure this match condition for IPv4 traffic, we recommend that you also configure the **protocol udp** or **protocol tcp** match statement in the same term to specify which protocol is being used on the port. If you configure this match condition for IPv6 traffic, we recommend that you also configure the **next-header udp** or **next-header tcp** match condition in the same term to specify which protocol is being used on the port. In place of the numeric value, you can specify one of the text synonyms listed with the **destination-port** *number* match condition.	• **family inet** • **family inet6**
source-port-except *number*	Do not match the UDP or TCP source port field. For details, see the **source-port** match condition.	• **family inet** • **family inet6**
source-prefix-list *name*	Match source prefixes in the specified list. Specify the name of a prefix list defined at the **[edit policy-options prefix-list** *prefix-list-name*] hierarchy level.	• **family inet** • **family inet6**

Table 59: Service Filter Match Conditions for IPv4 or IPv6 Traffic *(continued)*

Match Condition	Description	Protocol Families
tcp-flags *value*	Match one or more of the low-order 6 bits in the 8-bit TCP flags field in the TCP header. To specify individual bit fields, you can specify the following text synonyms or hexadecimal values: • **fin** (0x01) • **syn** (0x02) • **rst** (0x04) • **push** (0x08) • **ack** (0x10) • **urgent** (0x20) In a TCP session, the SYN flag is set only in the initial packet sent, while the ACK flag is set in all packets sent after the initial packet. You can string together multiple flags using the bit-field logical operators. For combined bit-field match conditions, see the **tcp-established** and **tcp-initial** match conditions. If you configure this match condition for IPv4 traffic, we recommend that you also configure the **protocol tcp** match statement in the same term to specify that the TCP protocol is being used on the port. If you configure this match condition for IPv6 traffic, we recommend that you also configure the **next-header tcp** match condition in the same term to specify that the TCP protocol is being used on the port.	• **family inet** • **family inet6**

NOTE: If you specify an IPv6 address in a match condition (the **address**, **destination-address**, or **source-address** match conditions), use the syntax for text representations described in RFC 4291, *IP Version 6 Addressing Architecture*. For more information about IPv6 addresses, see “*IPv6 Overview*” in the *Junos OS Routing Protocols Library*.

Related Documentation

- Service Filter Overview on page 905
- Guidelines for Configuring Service Filters on page 908
- Example: Configuring and Applying Service Filters on page 913
- Service Filter Terminating Actions on page 925
- Service Filter Nonterminating Actions on page 924

Service Filter Nonterminating Actions

Service filters support different sets of terminating actions for each protocol family.

NOTE: **Service filters do not support the next term action.**

Table 60 on page 925 describes the nonterminating actions you can configure in a service filter term.

Table 60: Nonterminating Actions for Service Filters

Nonterminating Action	Description	Protocol Families
count ***counter-name***	Count the packet in the named counter.	• **inet** • **inet6**
log	Log the packet header information in a buffer within the Packet Forwarding Engine. You can access this information by issuing the **show firewall log** command at the command-line interface (CLI).	• **inet** • **inet6**
port-mirror	Port-mirror the packet based on the specified family. Supported on M120 routers, M320 routers configured with Enhanced III FPCs, MX Series routers, and EX Series switches only.	• **inet** • **inet6**
sample	Sample the packet.	• **inet** • **inet6**

Related Documentation

- Service Filter Overview on page 905
- Guidelines for Configuring Service Filters on page 908
- Example: Configuring and Applying Service Filters on page 913
- Service Filter Match Conditions for IPv4 or IPv6 Traffic on page 918
- Service Filter Terminating Actions on page 925

Service Filter Terminating Actions

Service filters support different sets of terminating actions than standard stateless firewall filters or simple filters.

NOTE: **Service filters do not support the next term action.**

Table 61 on page 926 describes the terminating actions you can configure in a service filter term.

Table 61: Terminating Actions for Service Filters

Terminating Action	Description	Protocol Families
service	Direct the packet to service processing.	• inet • inet6
skip	Let the packet bypass service processing.	• inet • inet6

Related Documentation

- Service Filter Overview on page 905
- Guidelines for Configuring Service Filters on page 908
- Example: Configuring and Applying Service Filters on page 913
- Service Filter Match Conditions for IPv4 or IPv6 Traffic on page 918
- Service Filter Nonterminating Actions on page 924

CHAPTER 24

Configuring Simple Filters

Simple Filter Overview

Simple filters are supported on Gigabit Ethernet intelligent queuing 2 (IQ2) and Enhanced Queuing Dense Port Concentrator (DPC) interfaces only.

Simple filters are recommended for metropolitan Ethernet applications.

Related Documentation

- How Simple Filters Evaluate Packets on page 927
- Guidelines for Configuring Simple Filters on page 929
- Guidelines for Applying Simple Filters on page 932
- Example: Configuring and Applying a Simple Filter on page 933

How Simple Filters Evaluate Packets

This topic covers the following information:

Simple Filters That Contain a Single Term

For a simple filter that consists of a single term, the policy framework software evaluates a packet as follows:

- If the packet matches all the conditions, the actions are taken.
- If the packet matches all the conditions and no actions are specified, the packet is accepted.
- If the packet does not match all the conditions, it is discarded.

Simple Filters That Contain Multiple Terms

For a simple filter that consists of multiple terms, the policy framework software evaluates a packet against the terms in the filter sequentially, beginning with the first term in the filter, until either the packet matches all the conditions in one of the terms or there are no more terms in the filter.

- If the packet matches all the conditions in a term, the actions in that term are performed and evaluation of the packet ends at that term. Any subsequent terms in the filter are not used.
- If the packet does not match all the conditions in the term, evaluation of the packet proceeds to the next term in the filter.

Simple Filter Terms That Do Not Contain Any Match Conditions

For simple filters with a single term and for filters with multiple terms, if a term does not contain any match conditions, the actions are taken on any packet evaluated.

Simple Filter Terms That Do Not Contain Any Actions

If a simple filter term does not contain any actions, and if the packet matches the conditions in the term, the packet is accepted.

Simple Filter Default Action

Each simple filter has an *implicit* **discard** action at the end of the filter, which is equivalent to including the following example term **explicit_discard** as the final term in the simple filter:

```
term explicit_discard {
  then discard;
}
```

By default, if a packet matches none of the terms in a simple filter, the packet is discarded.

Related Documentation

- Simple Filter Overview on page 927
- Guidelines for Configuring Simple Filters on page 929
- Guidelines for Applying Simple Filters on page 932
- Example: Configuring and Applying a Simple Filter on page 933

Guidelines for Configuring Simple Filters

This topic covers the following information:

Statement Hierarchy for Configuring Simple Filters

To configure a simple filter, include the **simple-filter** ***simple-filter-name*** statement at the **[edit firewall family inet]** hierarchy level.

```
[edit]
firewall {
  family inet {
    simple-filter simple-filter-name {
      term term-name {
        from {
          match-conditions;
        }
        then {
          actions;
        }
      }
    }
  }
}
```

Individual statements supported under the **simple-filter** ***simple-filter-name*** statement are described separately in this topic and are illustrated in the example of configuring and applying a simple filter.

Simple Filter Protocol Families

You can configure simple filters to filter IPv4 traffic (**family inet**) only. No other protocol family is supported for simple filters.

Simple Filter Names

Under the **family inet** statement, you can include **simple-filter** ***simple-filter-name*** statements to create and name simple filters. The filter name can contain letters, numbers, and hyphens (-) and be up to 64 characters long. To include spaces in the name, enclose the entire name in quotation marks (“ ”).

Simple Filter Terms

Under the **simple-filter** ***simple-filter-name*** statement, you can include **term** ***term-name*** statements to create and name filter terms.

- You must configure at least one term in a firewall filter.
- You must specify a unique name for each term within a firewall filter. The term name can contain letters, numbers, and hyphens (-) and can be up to 64 characters long. To include spaces in the name, enclose the entire name in quotation marks (" ").
- The order in which you specify terms within a firewall filter configuration is important. Firewall filter terms are evaluated in the order in which they are configured. By default, new terms are always added to the end of the existing filter. You can use the **insert** configuration mode command to reorder the terms of a firewall filter.

Simple filters do *not* support the **next term** action.

Simple Filter Match Conditions

Simple filter terms support only a subset of the IPv4 match conditions that are supported for standard stateless firewall filters.

Unlike standard stateless firewall filters, the following restrictions apply to simple filters:

- On MX Series routers with the Enhanced Queuing DPC and on EX Series switches, simple filters do *not* support the **forwarding- class** match condition.
- Simple filters support only one **source-address** and one **destination-address** prefix for each filter term. If you configure multiple prefixes, only the last one is used.
- Simple filters do *not* support multiple source addresses and destination addresses in a single term. If you configure multiple addresses, only the last one is used.
- Simple filters do *not* support negated match conditions, such as the **protocol-except** match condition or the **exception** keyword.
- Simple filters support a range of values for **source-port** and **destination-port** match conditions only. For example, you can configure **source-port 400-500** or **destination-port 600-700**.
- Simple filters do *not* support noncontiguous mask values.

Table 62 on page 930 lists the simple filter match conditions.

Table 62: Simple Filter Match Conditions

Match Condition	Description
destination-address ***destination-address***	Match IP destination address.

Table 62: Simple Filter Match Conditions *(continued)*

Match Condition	Description
destination-port *number*	TCP or UDP destination port field. If you configure this match condition, we recommend that you also configure the **protocol** match statement to determine which protocol is being used on the port. In place of the numeric value, you can specify one of the following text aliases (the port numbers are also listed): **afs** (1483), **bgp** (179), **biff** (512), **bootpc** (68), **bootps** (67), **cmd** (514), **cvspserver** (2401), **dhcp** (67), **domain** (53), **eklogin** (2105), **ekshell** (2106), **exec** (512), **finger** (79), **ftp** (21), **ftp-data** (20), **http** (80), **https** (443), **ident** (113), **imap** (143), **kerberos-sec** (88), **klogin** (543), **kpasswd** (761), **krb-prop** (754), **krbupdate** (760), **kshell** (544), **ldap** (389), **login** (513), **mobileip-agent** (434), **mobilip-mn** (435), **msdp** (639), **netbios-dgm** (138), **netbios-ns** (137), **netbios-ssn** (139), **nfsd** (2049), **nntp** (119), **ntalk** (518), **ntp** (123), **pop3** (110), **pptp** (1723), **printer** (515), **radacct** (1813), **radius** (1812), **rip** (520), **rkinit** (2108), **smtp** (25), **snmp** (161), **snmptrap** (162), **snpp** (444), **socks** (1080), **ssh** (22), **sunrpc** (111), **syslog** (514), **tacacs-ds** (65), **talk** (517), **telnet** (23), **tftp** (69), **timed** (525), **who** (513), or **xdmcp** (177).
forwarding-class *class*	Match the forwarding class of the packet. Specify **assured-forwarding**, **best-effort**, **expedited-forwarding**, or **network-control**. For information about forwarding classes and router-internal output queues, see *Understanding How Forwarding Classes Assign Classes to Output Queues*.
protocol *number*	IP protocol field. In place of the numeric value, you can specify one of the following text aliases (the field values are also listed): **ah** (51), **dstopts** (60), **egp** (8), **esp** (50), **fragment** (44), **gre** (47), **hop-by-hop** (0), **icmp** (1), **icmp6** (58), **icmpv6** (58), **igmp** (2), **ipip** (4), **ipv6** (41), **ospf** (89), **pim** (103), **rsvp** (46), **sctp** (132), **tcp** (6), **udp** (17), or **vrrp** (112).
source-address *ip-source-address*	Match the IP source address.
source-port *number*	Match the UDP or TCP source port field. If you configure this match condition, we recommend that you also configure the **protocol** match statement to determine which protocol is being used on the port. In place of the numeric field, you can specify one of the text aliases listed for **destination-port**.

Simple Filter Terminating Actions

Simple filters do *not* support explicitly configurable terminating actions, such as **accept**, **reject**, and **discard**. Terms configured in a simple filter always accept packets.

Simple filters do *not* support the **next** action.

Simple Filter Nonterminating Actions

Simple filters support only the following nonterminating actions:

- **forwarding-class (*forwarding-class* | assured-forwarding |best-effort | expedited-forwarding | network-control)**

NOTE: On the MX Series routers and EX Series switches with the Enhanced Queuing DPC, the forwarding class is not supported as a **from** match condition.

- **loss-priority (high | low | medium-high | medium-low)**

Simple filters do not support actions that perform other functions on a packet (such as incrementing a counter, logging information about the packet header, sampling the packet data, or sending information to a remote host using the system log functionality).

Related Documentation

- Simple Filter Overview on page 927
- How Simple Filters Evaluate Packets on page 927
- Guidelines for Applying Simple Filters on page 932
- Example: Configuring and Applying a Simple Filter on page 933

Guidelines for Applying Simple Filters

This topic covers the following information:

- Statement Hierarchy for Applying Simple Filters on page 932
- Restrictions for Applying Simple Filters on page 932

Statement Hierarchy for Applying Simple Filters

You can apply a simple filter to the IPv4 ingress traffic at a logical interface by including the **simple-filter input** ***simple-filter-name*** statement at the **[edit interfaces** ***interface-name*** **unit** ***unit-number*** **family inet]** hierarchy level.

```
[edit]
interfaces {
  interface-name {
    unit logical-unit-number {
      family inet {
        simple-filter {
          input filter-name;
        }
      }
    }
  }
}
```

Restrictions for Applying Simple Filters

You can apply a simple filter to the ingress IPv4 traffic at a logical interface configured on the following hardware only:

- Gigabit Ethernet intelligent queuing (IQ2) PICs installed on M120, M320, or T Series routers.

- Enhanced Queuing Dense Port Concentrators (EQ DPCs) installed on MX Series routers and EX Series switches.

The following additional restrictions pertain to applying simple filters:

- Simple filters are not supported on Modular Port Concentrator (MPC) interfaces, including Enhanced Queuing MPC interfaces.
- Simple filters are not supported for interfaces in an aggregated-Ethernet bundle.
- You can apply simple filters to **family inet** traffic only. No other protocol family is supported.
- You can apply simple filters to ingress traffic only. Egress traffic is not supported.
- You can apply only a single simple filter to a supported logical interface. Input lists are not supported.

Related Documentation

- Simple Filter Overview on page 927
- How Simple Filters Evaluate Packets on page 927
- Guidelines for Configuring Simple Filters on page 929
- Example: Configuring and Applying a Simple Filter on page 933

Example: Configuring and Applying a Simple Filter

This example shows how to configure a simple filter.

Requirements

This example uses one of the following hardware components:

- One Gigabit Ethernet intelligent queuing (IQ2) PIC installed on an M120, M320, or T Series router
- One Enhanced Queuing Dense Port Concentrator (EQ DPC) installed on an MX Series router or an EX Series switch

Before you begin, make sure that you have:

- Installed your supported router (or switch) and PIC or DPC and performed the initial router (or switch) configuration.
- Configured basic Ethernet in the topology, and verified that traffic is flowing in the topology and that ingress IPv4 traffic is flowing into logical interface **ge-0/0/1.0**.

Overview

This simple filter sets the loss priority to low for TCP traffic with source address **172.16.1.1**, sets the loss priority to high for HTTP (Web) traffic with source addresses in the 172.16.4.0/8 range, and sets the loss priority to low for all traffic with destination address **172.16.6.6**.

Topology

The simple filter is applied as an input filter (arriving packets are checking for destination address **6.6.6.6**, not queued output packets) on interface **ge-0/0/1.0**.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring the Simple Firewall Filter on page 934
- Applying the Simple Filter to the Logical Interface Input on page 936

CLI Quick Configuration

To quickly configure this example, copy the following commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall family inet simple-filter sf_classify_1 term 1 from source-address 172.16.1.1/32
set firewall family inet simple-filter sf_classify_1 term 1 from protocol tcp
set firewall family inet simple-filter sf_classify_1 term 1 then loss-priority low
set firewall family inet simple-filter sf_classify_1 term 2 from source-address 172.16.4.0/8
set firewall family inet simple-filter sf_classify_1 term 2 from protocol tcp
set firewall family inet simple-filter sf_classify_1 term 2 from source-port http
set firewall family inet simple-filter sf_classify_1 term 2 then loss-priority high
set firewall family inet simple-filter sf_classify_1 term 3 from destination-address
  6.6.6.6/32
set firewall family inet simple-filter sf_classify_1 term 3 then loss-priority low
set firewall family inet simple-filter sf_classify_1 term 3 then forwarding-class best-effort
set interfaces ge-0/0/1 unit 0 family inet simple-filter input sf_classify_1
set interfaces ge-0/0/1 unit 0 family inet address 10.1.2.3/30
```

Configuring the Simple Firewall Filter

Step-by-Step Procedure

To configure the simple filter:

1. Create the simple filter **sf_classify_1**.

```
[edit]
user@host# edit firewall family inet simple-filter sf_classify_1
```

2. Configure classification of TCP traffic based on the source IP address.

```
[edit firewall family inet simple-filter sf_classify_1]
user@host# set term 1 from source-address 172.16.1.1/32
```



```
user@host# set term 1 from protocol tcp
user@host# set term 1 then loss-priority low
```

3. Configure classification of HTTP traffic based on the source IP address.

```
[edit firewall family inet simple-filter sf_classify_1]
user@host# set term 2 from source-address 172.16.4.0/8
user@host# set term 2 from protocol tcp
user@host# set term 2 from source-port http
user@host# set term 2 then loss-priority high
```

4. Configure classification of other traffic based on the destination IP address.

```
[edit firewall family inet simple-filter sf_classify_1]
user@host# set term 3 from destination-address 6.6.6.6/32
user@host# set term 3 then loss-priority low
user@host# set term 3 then forwarding-class best-effort
```

Results Confirm the configuration of the simple filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  simple-filter sf_classify_1 {
    term 1 {
      from {
        source-address {
          172.16.1.1/32;
        }
        protocol {
          tcp;
        }
      }
      then loss-priority low;
    }
    term 2 {
      from {
        source-address {
          172.16.4.0/8;
        }
        source-port {
          http;
        }
        protocol {
          tcp;
        }
      }
      then loss-priority high;
    }
    term 3 {
      from {
```

```
                destination-address {
                    6.6.6.6/32;
                }
            }
            then {
                loss-priority low;
                forwarding-class best-effort;
            }
        }
    }
}
```

Applying the Simple Filter to the Logical Interface Input

Step-by-Step Procedure

To apply the simple filter to the logical interface input:

1. Configure the logical interface to which you will apply the simple filter.

   ```
   [edit]
   user@host# edit interfaces ge-0/0/1 unit 0 family inet
   ```

2. Configure the interface address for the logical interface.

   ```
   [edit interfaces ge-0/0/1 unit 0 family inet]
   user@host# set address 10.1.2.3/30
   ```

3. Apply the simple filter to the logical interface input.

   ```
   [edit interfaces ge-0/0/1 unit 0 family inet]
   user@host# set simple-filter input sf_classify_1
   ```

Results

Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
ge-0/0/1 {
    unit 0 {
        family inet {
            simple-filter {
                input sf_classify_1;
            }
            address 10.1.2.3/30;
        }
    }
}
```

When you are done configuring the device, commit your candidate configuration.

Verification

Confirm that the configuration is working properly.

- Displaying the Mapping of Forwarding Class Maps and Names to Queue Numbers on page 937
- Displaying CoS Queue Counters for the Interface on page 937
- Displaying CoS Queue Counter Details for the Physical Interface on page 937

Displaying the Mapping of Forwarding Class Maps and Names to Queue Numbers

Purpose Display the mapping of forwarding class names to queue numbers.

Action Enter the **show class-of-service forwarding-class** operational mode command.

```
[edit]
user@host> show class-of-service forwarding-class
```

For information about the command output, see "**show class-of-service forwarding-class**" in the CLI Explorer.

Displaying CoS Queue Counters for the Interface

Purpose Verify that the class-of-service (CoS) queue counters for the interface reflect the simple filter applied to the logical interface.

Action Enter the **show interfaces** command for the physical interface on which the simple filter is applied, and specify **detail** or **extensive** output level.

```
[edit]
user@host> show interfaces ge-0/0/1 detail
```

In the **Physical interface** section, under **Ingress queues**, the **Queue counters** section displays ingress queue counters for each forwarding class.

For more detailed information about the command output, see "**show interfaces (Gigabit Ethernet)**" or "**show interfaces (10-Gigabit Ethernet)**" in the CLI Explorer.

Displaying CoS Queue Counter Details for the Physical Interface

Purpose Verify that the CoS queue counter details for the physical interface reflect the simple filter applied to the logical interface.

Action Enter the **show interfaces queue** command for the physical interface on which the simple filter is applied, and specify the **ingress** option.

```
[edit]
user@host> show interfaces queue ge-0/0/1 ingress
```


For information about the command output, see “**show interfaces queue**” in the CLI Explorer.

Related Documentation

- Simple Filter Overview on page 927
- How Simple Filters Evaluate Packets on page 927
- Guidelines for Configuring Simple Filters on page 929
- Guidelines for Applying Simple Filters on page 932

CHAPTER 25

Configuring Firewall Filters for Forwarding, Fragments, and Policing

Filter-Based Forwarding Overview

Firewall filters can be used to block specific packets. They can also be used to affect how specific packets are forwarded.

Filters That Classify Packets or Direct Them to Routing Instances

For IPv4 or IPv6 traffic only, you can use stateless firewall filters in conjunction with forwarding classes and routing instances to control how packets travel in a network. This is called *filter-based forwarding* (FBF).

You can define a filtering term that matches incoming packets based on source address and then classifies matching packets to a specified forwarding class. This type of filtering can be configured to grant certain types of traffic preferential treatment or to improve load balancing. To configure a stateless firewall filter to classify packets to a forwarding class, configure a term with the *nonterminating action* **forwarding-class *class-name***.

You can also define a filtering term that directs matching packets to a specified routing instance. This type of filtering can be configured to route specific types of traffic through a firewall or other security device before the traffic continues on its path. To configure a stateless firewall filter to direct traffic to a routing instance, configure a term with the *terminating action* **routing-instance** ***routing-instance-name*** **<topology *topology-name*>** to specify the routing instance to which matching packets will be forwarded.

NOTE: Unicast Reverse Path Forwarding (uRPF) check is compatible with FBF actions. uRPF check is processed for source address checking before any FBF actions are enabled for static and dynamic interfaces. This applies to both IPv4 and IPv6 families.

To forward traffic to the master routing instance, reference **routing-instance default** in the firewall configuration, as shown here:

```
[edit firewall]
family inet {
   filter test {
      term 1 {
         then {
            routing-instance default;
         }
      }
   }
}
```

NOTE: Do not reference **routing-instance master**. This does not work.

Input Filtering to Classify and Forward Packets Within the Router or Switch

You can configure filters to classify packets based on source address and specify the forwarding path the packets take within the router or switch by configuring a filter on the ingress interface.

For example, you can use this filter for applications to differentiate traffic from two clients that have a common access layer (for example, a Layer 2 switch) but are connected to different Internet service providers (ISPs). When the filter is applied, the router or switch can differentiate the two traffic streams and direct each to the appropriate network. Depending on the media type the client is using, the filter can use the source IP address to forward the traffic to the corresponding network through a tunnel. You can also configure filters to classify packets based on IP protocol type or IP precedence bits.

Output Filtering to Forward Packets to Another Routing Table

You can also forward packets based on output filters by configuring a filter on the egress interfaces. In the case of port mirroring, it is useful for port-mirrored packets to be distributed to multiple monitoring PICs and collection PICs based on patterns in packet headers. FBF on the port-mirroring egress interface must be configured.

Packets forwarded to the output filter have been through at least one route lookup when an FBF filter is configured on the egress interface. After the packet is classified at the egress interface by the FBF filter, it is redirected to another routing table for further route lookup.

Restrictions for Applying Filter-Based Forwarding

An interface configured with filter-based forwarding does not support source-class usage (SCU) filter matching or source-class and destination-class usage (SCU/DCU) accounting.

Related Documentation

- Example: Configuring Filter-Based Forwarding on the Source Address on page 942
- Example: Configuring Filter-Based Forwarding on Logical Systems on page 804

Firewall Filters That Handle Fragmented Packets Overview

You can create stateless firewall filters that handle fragmented packets destined for the Routing Engine. By applying these policies to the Routing Engine, you protect against the use of IP fragmentation as a means to disguise TCP packets from a firewall filter.

For example, consider an IP packet that is fragmented into the smallest allowable fragment size of 8 bytes (a 20-byte IP header plus an 8-byte payload). If this IP packet carries a TCP packet, the first fragment (fragment offset of 0) that arrives at the device contains only the TCP source and destination ports (first 4 bytes), and the sequence number (next 4 bytes). The TCP flags, which are contained in the next 8 bytes of the TCP header, arrive in the second fragment (fragment offset of 1).

See RFC 1858, *Security Considerations for IP Fragment Filtering*.

Related Documentation

- Understanding How to Use Standard Firewall Filters on page 559
- Example: Configuring a Stateless Firewall Filter to Handle Fragments on page 776

Stateless Firewall Filters That Reference Policers Overview

Policing, or rate limiting, is an important component of firewall filters that lets you limit the amount of traffic that passes into or out of an interface.

A firewall filter that references a policer can provide protection from denial-of-service (DOS) attacks. Traffic that exceeds the rate limits configured for the policer is either discarded or marked as lower priority than traffic that conforms to the configured rate limits. Packets can be marked for a lower priority by being set to a specific output queue, set to a specific packet loss priority (PLP) level, or both. When necessary, low-priority traffic can be discarded to prevent congestion.

A policer specifies two types of rate limits on traffic:

- Bandwidth limit—The average traffic rate permitted, specified as a number of bits per second.
- Maximum burst size—The packet size permitted for bursts of data that exceed the bandwidth limit.

Policing uses an algorithm to enforce a limit on average bandwidth while allowing bursts up to a specified maximum value. You can use policing to define specific classes of traffic on an interface and apply a set of rate limits to each class. After you name and configure a policer, it is stored as a template. You can then apply the policer in an interface configuration or, to rate-limit packet-filtered traffic only, in a firewall filter configuration.

For an IPv4 firewall filter term only, you can also specify a *prefix-specific action* as a nonterminating action that applies a policer to the matched packets. A prefix-specific action applies additional matching criteria on the filter-matched packets based on specified address prefix bits and then associates the matched packets with a counter and policer instance for that filter term or for all terms in the firewall filter.

To apply a policer or a prefix action to packet-filtered traffic, you can use the following firewall filter nonterminating actions:

- **policer *policer-name***
- **three-color-policer (single-rate | two-rate) *policer-name***
- **prefix-action *action-name***

NOTE: The packet lengths that a policer considers depends on the address family of the firewall filter. See "Understanding the Frame Length for Policing Packets" on page 980.

Related Documentation

- Firewall Filter Nonterminating Actions on page 673
- Controlling Network Access Using Traffic Policing Overview on page 971
- Prefix-Specific Counting and Policing Overview on page 1072

Example: Configuring Filter-Based Forwarding on the Source Address

This example shows how to configure filter-based forwarding (FBF), which is sometimes also called Policy Based Routing (PBR). The filter classifies packets to determine their forwarding path within the ingress routing device.

Filter-based forwarding is supported for IP version 4 (IPv4) and IP version 6 (IPv6).

Requirements

No special configuration beyond device initialization is required for this example.

Overview

In this example, we use FBF for service provider selection when customers have Internet connectivity provided by different ISPs yet share a common access layer. When a shared media (such as a cable modem) is used, a mechanism on the common access layer looks at Layer 2 or Layer 3 addresses and distinguishes between customers. You can use filter-based forwarding when the common access layer is implemented using a combination of Layer 2 switches and a single router.

With FBF, all packets received on an interface are considered. Each packet passes through a filter that has match conditions. If the match conditions are met for a filter and you have created a routing instance, FBF is applied to the packet. The packet is forwarded based on the next hop specified in the routing instance. For static routes, the next hop can be a specific LSP.

NOTE: **Source-class usage filter matching and unicast reverse-path forwarding checks are not supported on an interface configured for FBF.**

To configure FBF, perform the following tasks:

- Create a match filter on the ingress device. To specify a match filter, include the **filter *filter-name*** statement at the **[edit firewall]** hierarchy level. A packet that passes through the filter is compared against a set of rules to classify it and to determine its membership in a set. Once classified, the packet is forwarded to a routing table specified in the accept action in the filter description language. The routing table then forwards the packet to the next hop that corresponds to the destination address entry in the table.
- Create routing instances that specify the routing table(s) to which a packet is forwarded, and the destination to which the packet is forwarded at the **[edit routing-instances]** hierarchy level. For example:

```
[edit]
routing-instances {
  routing-table-name1 {
    instance-type forwarding;
    routing-options {
      static {
        route 0.0.0.0/0 next-hop 172.16.0.14;
      }
    }
  }
  routing-table-name2 {
    instance-type forwarding;
    routing-options {
      static {
        route 0.0.0.0/0 next-hop 172.16.0.18;
      }
    }
  }
}
```

- Create a RIB group to share interface routes with the forwarding routing instances used in filter-based forwarding (FBF). This part of the configuration resolves the routes installed in the routing instances to directly connected next hops on that interface. Create the routing table group at the **[edit routing-options]** hierarchy level.

```
[edit]
```

```
routing-options {
   interface-routes {
      rib-group;
      inet {
         int-routes;
         }
      }
   }
}
routing-options {
   rib-groups {
      int-routes {
      import-rib {
         inet.0;
         webtraffic.inet.0;
         }
      }
   }
}
```

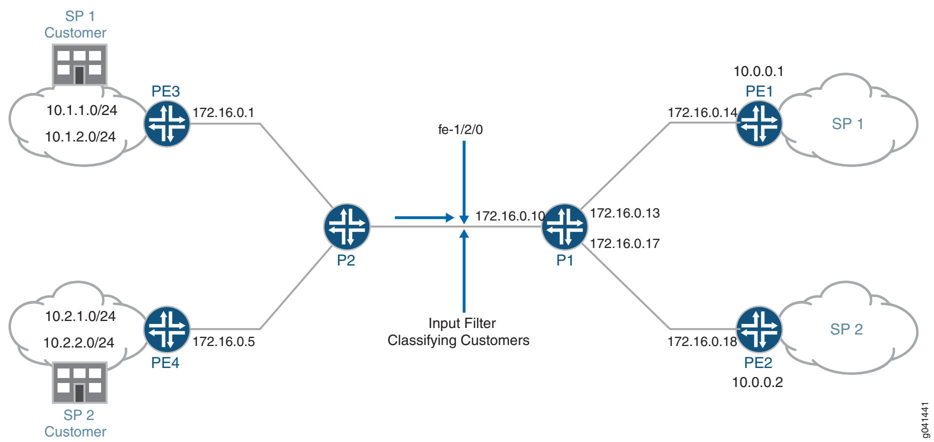
This example shows a packet filter that directs customer traffic to a next-hop router in the domains, SP1 or SP2, based on the packet's source address.

If the packet has a source address assigned to an SP1 customer, destination-based forwarding occurs using the sp1-route-table.inet.0 routing table. If the packet has a source address assigned to an SP2 customer, destination-based forwarding occurs using the sp2-route-table.inet.0 routing table. If a packet does not match either of these conditions, the filter accepts the packet, and destination-based forwarding occurs using the standard inet.0 routing table.

Figure 58 on page 944 shows the topology used in this example.

On Device P1, an input filter classifies packets received from Device PE3 and Device PE4. The packets are routed based on the source addresses. Packets with source addresses in the 10.1.1.0/24 and 10.1.2.0/24 networks are routed to Device PE1. Packets with source addresses in the 10.2.1.0/24 and 10.2.2.0/24 networks are routed to Device PE2.

Figure 58: Filter-Based Forwarding



To establish connectivity, OSPF is configured on all of the interfaces. For demonstration purposes, loopback interface addresses are configured on the routing devices to represent networks in the clouds.

The "CLI Quick Configuration" on page 945 section shows the entire configuration for all of the devices in the topology. The "Configuring Filter-Based Forwarding on Device P1" on page 947 section shows the step-by-step configuration of the ingress routing device, Device P1.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device P1

```
set firewall filter classify-customers term sp1-customers from source-address 10.1.1.0/24
set firewall filter classify-customers term sp1-customers from source-address 10.1.2.0/24
set firewall filter classify-customers term sp1-customers then log
set firewall filter classify-customers term sp1-customers then routing-instance
  sp1-route-table
set firewall filter classify-customers term sp2-customers from source-address 10.2.1.0/24
set firewall filter classify-customers term sp2-customers from source-address 10.2.2.0/24
set firewall filter classify-customers term sp2-customers then log
set firewall filter classify-customers term sp2-customers then routing-instance
  sp2-route-table
set firewall filter classify-customers term default then accept
set interfaces fe-1/2/0 unit 0 family inet filter input classify-customers
set interfaces fe-1/2/0 unit 0 family inet address 172.16.0.10/30
set interfaces fe-1/2/1 unit 0 family inet address 172.16.0.13/30
set interfaces fe-1/2/2 unit 0 family inet address 172.16.0.17/30
set protocols ospf rib-group fbf-group
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
set routing-instances sp1-route-table instance-type forwarding
set routing-instances sp1-route-table routing-options static route 0.0.0.0/0 next-hop
  172.16.0.14
set routing-instances sp2-route-table instance-type forwarding
set routing-instances sp2-route-table routing-options static route 0.0.0.0/0 next-hop
  172.16.0.18
set routing-options rib-groups fbf-group import-rib inet.0
set routing-options rib-groups fbf-group import-rib sp1-route-table.inet.0
set routing-options rib-groups fbf-group import-rib sp2-route-table.inet.0
```

Device P2

```
set interfaces fe-1/2/0 unit 0 family inet address 172.16.0.2/30
set interfaces fe-1/2/1 unit 0 family inet address 172.16.0.6/30
set interfaces fe-1/2/2 unit 0 family inet address 172.16.0.9/30
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable
```

Device PE1

```
set interfaces fe-1/2/0 unit 0 family inet address 172.16.0.14/30
set interfaces lo0 unit 0 family inet address 172.16.1.1/32
set protocols ospf area 0.0.0.0 interface all
```


set protocols ospf area 0.0.0.0 interface fxp0.0 disable

Device PE2

set interfaces fe-1/2/0 unit 0 family inet address 172.16.0.18/30
set interfaces lo0 unit 0 family inet address 172.16.2.2/32
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable

Device PE3

set interfaces fe-1/2/0 unit 0 family inet address 172.16.0.1/30
set interfaces lo0 unit 0 family inet address 10.1.1.1/32
set interfaces lo0 unit 0 family inet address 10.1.2.1/32
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable

Device PE4

set interfaces fe-1/2/0 unit 0 family inet address 172.16.0.5/30
set interfaces lo0 unit 0 family inet address 10.2.1.1/32
set interfaces lo0 unit 0 family inet address 10.2.2.1/32
set protocols ospf area 0.0.0.0 interface all
set protocols ospf area 0.0.0.0 interface fxp0.0 disable

Configuring the Firewall Filter on P1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the firewall filter on the main router or switch:

1. Configure the source addresses for SP1 customers.

   [edit firewall filter classify-customers term sp1-customers]
   user@host# **set from source-address 10.1.1.0/24**
   user@host# **set from source-address 10.1.2.0/24**

2. Configure the actions that are taken when packets are received with the specified source addresses; they are logged, and they are passed to the sp1-route-table routing instance for routing via the sp1-route-table.inet.0 routing table.

   [edit firewall filter classify-customers term sp1-customers]
   user@host# **set then log**
   user@host# **set then routing-instance sp1-route-table**

3. Configure the source addresses for SP2 customers.

   [edit firewall filter classify-customers term sp2-customers]
   user@host# **set from source-address 10.2.1.0/24**
   user@host# **set from source-address 10.2.2.0/24**

4. Configure the actions that are taken when packets are received with the specified source addresses; they are logged, and they are passed to the sp2-route-table routing instance for routing via the sp2-route-table.inet.0 routing table.

   [edit firewall filter classify-customers term sp2-customers]


```
user@host# set then log
user@host# set then routing-instance sp2-route-table
```

5. Configure the action to take when packets are received from any other source address; they are accepted and routed using the default IPv4 unicast routing table, inet.0.

```
[edit firewall filter classify-customers term default]
user@host# set then accept
```

Configuring Filter-Based Forwarding on Device P1

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure the routing instances:

1. Configure the interfaces.

```
[edit interfaces fe-1/2/0]
user@host# set unit 0 family inet address 172.16.0.10/30

[edit interfaces fe-1/2/1]
user@host# set unit 0 family inet address 172.16.0.13/30

[edit interfaces fe-1/2/2]
user@host# set unit 0 family inet address 172.16.0.17/30
```

2. Assign the **classify-customers** firewall filter to router interface fe-1/2/0.0 as an input packet filter.

```
[edit interfaces fe-1/2/0]
user@host# set unit 0 family inet filter input classify-customers
```

3. Configure connectivity, using either a routing protocol or static routing.

   As a best practice, disable routing on the management interface.

```
[edit protocols ospf area 0.0.0.0]
user@host# set interface all
user@host# set interface fxp0.0 disable
```

4. Create the routing instances that are referenced in the **classify-customers** firewall filter. The forwarding instance type provides support for filter-based forwarding, where interfaces are not associated with instances.

```
[edit routing-instances]
user@host# set sp1-route-table instance-type forwarding

user@host# set sp2-route-table instance-type forwarding
```

5. For each routing instance, define a default route to forward traffic to the specified next hop (PE1 and PE2 in this example).

```
[edit routing-instances ]
user@host# set sp1-route-table routing-options static route 0.0.0.0/0 next-hop
  172.16.0.14

user@host# set sp2-route-table routing-options static route 0.0.0.0/0 next-hop
  172.16.0.18
```

6. Associate the routing tables to form a routing table group. The first routing table, inet.0, is the primary routing table, and the others are secondary routing tables. The primary routing table determines the address family of the routing table group, in this case IPv4.

```
[edit routing-options]
user@host# set rib-groups fbf-group import-rib inet.0
user@host# set rib-groups fbf-group import-rib sp1-route-table.inet.0
user@host# set rib-groups fbf-group import-rib sp2-route-table.inet.0
```

7. Specify the fbf-group routing table group within the OSPF configuration to install OSPF routes into the three routing tables.

```
[edit protocols ospf]
user@host# set rib-group fbf-group
```

8. Commit the configuration when you are done.

```
[edit]
user@host# commit
```

Results

Confirm your configuration by issuing the **show interfaces**, **show firewall**, **show protocols**, **show routing-instances**, and **show routing-options** commands.

```
user@host# show interfaces
fe-1/2/0 {
  unit 0 {
    family inet {
      filter {
        input classify-customers;
      }
      address 172.16.0.10/30;
    }
  }
}
fe-1/2/1 {
  unit 0 {
    family inet {
      address 172.16.0.13/30;
    }
```



```
    }
}
fe-1/2/2 {
    unit 0 {
        family inet {
            address 172.16.0.17/30;
        }
    }
}
```

user@host# **show firewall**

```
filter classify-customers {
    term sp1-customers {
        from {
            source-address {
                10.1.1.0/24;
                10.1.2.0/24;
            }
        }
        then {
            log;
            routing-instance sp1-route-table;
        }
    }
    term sp2-customers {
        from {
            source-address {
                10.2.1.0/24;
                10.2.2.0/24;
            }
        }
        then {
            log;
            routing-instance sp2-route-table;
        }
    }
    term default {
        then accept;
    }
}
```

user@host# **show protocols**

```
ospf {
    rib-group fbf-group;
    area 0.0.0.0 {
        interface all;
        interface fxp0.0 {
            disable;
        }
    }
}
```

user@host# **show routing-instances**

```
sp1-route-table {
    instance-type forwarding;
    routing-options {
        static {
```

```
            route 0.0.0.0/0 next-hop 172.16.0.14;
          }
        }
      }
      sp2-route-table {
        instance-type forwarding;
        routing-options {
          static {
            route 0.0.0.0/0 next-hop 172.16.0.18;
          }
        }
      }
```

```
user@host# show routing-options
rib-groups {
  fbf-group {
    import-rib [ inet.0 sp1-route-table.inet.0 sp2-route-table.inet.0 ];
  }
}
```

Verification

Confirm that the configuration is working properly.

Pinging with Specified Source Addresses

Purpose Send some ICMP packets across the network to test the firewall filter.

Action

1. Run the **ping** command, pinging the lo0.0 interface on Device PE1.

   The address configured on this interface is 172.16.1.1.

   Specify the source address 10.1.2.1, which is the address configured on the lo0.0 interface on Device PE3.

   ```
   user@PE3> ping 172.16.1.1 source 10.1.2.1
   PING 172.16.1.1 (172.16.1.1): 56 data bytes
   64 bytes from 172.16.1.1: icmp_seq=0 ttl=62 time=1.444 ms
   64 bytes from 172.16.1.1: icmp_seq=1 ttl=62 time=2.094 ms
   ^C
   --- 172.16.1.1 ping statistics ---
   2 packets transmitted, 2 packets received, 0% packet loss
   round-trip min/avg/max/stddev = 1.444/1.769/2.094/0.325 ms
   ```

2. Run the **ping** command, pinging the lo0.0 interface on Device PE2.

   The address configured on this interface is 172.16.2.2.

   Specify the source address 10.2.1.1, which is the address configured on the lo0.0 interface on Device PE4.

   ```
   user@PE4> ping 172.16.2.2 source 10.2.1.1
   PING 172.16.2.2 (172.16.2.2): 56 data bytes
   64 bytes from 172.16.2.2: icmp_seq=0 ttl=62 time=1.473 ms
   64 bytes from 172.16.2.2: icmp_seq=1 ttl=62 time=1.407 ms
   ^C
   --- 172.16.2.2 ping statistics ---
   ```



```
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.407/1.440/1.473/0.033 ms
```

Meaning Sending these pings activates the firewall filter actions.

Verifying the Firewall Filter

Purpose Make sure the firewall filter actions take effect.

Action

1. Run the **show firewall log** command on Device P1.

```
user@P1> show firewall log
Log :
Time       Filter    Action Interface     Protocol        Src Addr
Dest Addr
13:52:20  pfe       A      fe-1/2/0.0    ICMP            10.2.1.1
172.16.2.2
13:52:19  pfe       A      fe-1/2/0.0    ICMP            10.2.1.1
172.16.2.2
13:51:53  pfe       A      fe-1/2/0.0    ICMP            10.1.2.1
172.16.1.1
13:51:52  pfe       A      fe-1/2/0.0    ICMP            10.1.2.1
172.16.1.1
```

Related Documentation

- *Configuring Filter-Based Forwarding*
- *Copying and Redirecting Traffic with Port Mirroring and Filter-Based Forwarding*
- *Using Filter-Based Forwarding to Export Monitored Traffic to Multiple Destinations*
- Filter-Based Forwarding Overview on page 939

Example: Configuring Filter-Based Forwarding to a Specific Outgoing Interface or Destination IP Address

Understanding Filter-Based Forwarding to a Specific Outgoing Interface or Destination IP Address

Policy-based routing (also known as filter-based forwarding) refers to the use of firewall filters that are applied to an interface to match certain IP header characteristics and to route only those matching packets differently than the packets would normally be routed.

Starting in Junos OS Release 12.2, you can use **then next-interface**, **then next-ip**, or **then next-ip6** as an action in a firewall filter. From specific match conditions, IPv4 and IPv6 addresses or an interface name can be specified as the response action to a match.

The set of match conditions can be as follows:

- Layer-3 properties (for example, the source or destination IP address or the TOS byte)
- Layer-4 properties (for example, the source or destination port)

The route for the given IPv4 or IPv6 address has to be present in the routing table for policy-based routing to take effect. Similarly, the route through the given interface has to be present in the forwarding table for **next-interface** action to take effect. This can be achieved by configuring an interior gateway protocol (IGP), such as OSPF or IS-IS, to advertise Layer 3 routes.

The firewall filter matches the conditions and forwards the packet to one of the following:

- An IPv4 address (using the **next-ip** firewall filter action)
- An IPv6 address (using the **next-ip6** firewall filter action)
- An interface (using the **next-interface** firewall filter action)

Suppose, for example, that you want to offer services to your customers, and the services reside on different servers. An example of a service might be hosted DNS or hosted FTP. As customer traffic arrives at the Juniper Networks routing device, you can use filter-based forwarding to send traffic to the servers by applying a match condition on a MAC address or an IP address or simply an incoming interface and send the packets to a certain outgoing interface that is associated with the appropriate server. Some of your destinations might be IPv4 or IPv6 addresses, in which case the **next-ip** or **next-ip6** action is useful.

Optionally, you can associate the outgoing interfaces or IP addresses with routing instances.

For example:

```
firewall {
  filter filter1 {
    term t1 {
      from {
        source-address {
          10.1.1.3/32;
        }
      }
      then {
        next-interface {
```



```
                xe-0/1/0.1;
                routing-instance rins1;
            }
        }
    }
    term t2 {
        from {
            source-address {
                10.1.1.4/32;
            }
        }
        then {
            next-interface {
                xe-0/1/0.2;
                routing-instance rins2;
            }
        }
    }
  }
}
routing-instances {
  rins1 {
    instance-type virtual-router;
    interface xe-0/1/0.1;
  }
  rins2 {
    instance-type virtual-router;
    interface xe-0/1/0.2;
  }
}
```

Example: Configuring Filter-Based Forwarding to a Specific Outgoing Interface

This example shows how to use **then next-interface** as an action in a firewall filter.

Requirements

This example has the following hardware and software requirements:

- MX Series 3D Universal Edge Router as the routing device with the firewall filter configured.
- Junos OS Release 12.2 running on the routing device with the firewall filter configured.
- The filter with the **next-interface** (or **next-ip**) action can only be applied to an interface that is hosted on a Trio MPC. If you apply the filter to an I-chip based DPC, the commit operation fails.
- The outgoing interface referred to in the **next-interface** ***interface-name*** action can be hosted on a Trio MPC or an I-chip based DPC.

Overview

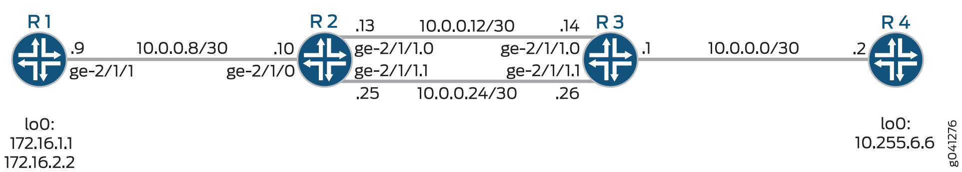
In this example, Device R1 has two loopback interface addresses configured: 172.16.1.1 and 172.16.2.2.

On Device R2, a firewall filter has multiple terms configured. Each term matches one of the source addresses in incoming traffic, and routes the traffic to specified outgoing interfaces. The outgoing interfaces are configured as VLAN-tagged interfaces between Device R2 and Device R3.

IS-IS is used for connectivity among the devices.

Figure 59 on page 954 shows the topology used in this example.

Figure 59: Filter-Based Forwarding to Specified Outgoing Interfaces



This example shows the configuration on Device R2.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R2

```
set interfaces ge-2/1/0 unit 0 family inet filter input filter1
set interfaces ge-2/1/0 unit 0 family inet address 10.0.0.10/30
set interfaces ge-2/1/0 unit 0 description to-R1
set interfaces ge-2/1/0 unit 0 family iso
set interfaces ge-2/1/1 vlan-tagging
set interfaces ge-2/1/1 description to-R3
set interfaces ge-2/1/1 unit 0 vlan-id 1001
set interfaces ge-2/1/1 unit 0 family inet address 10.0.0.13/30
set interfaces ge-2/1/1 unit 0 family iso
set interfaces ge-2/1/1 unit 1 vlan-id 1002
set interfaces ge-2/1/1 unit 1 family inet address 10.0.0.25/30
set interfaces ge-2/1/1 unit 1 family iso
set interfaces lo0 unit 0 family inet address 10.255.4.4/32
set interfaces lo0 unit 0 family iso address 49.0001.0010.0000.0404.00
set firewall family inet filter filter1 term t1 from source-address 172.16.1.1/32
set firewall family inet filter filter1 term t1 then next-interface ge-2/1/1.0
set firewall family inet filter filter1 term t2 from source-address 172.16.2.2/32
set firewall family inet filter filter1 term t2 then next-interface ge-2/1/1.1
set protocols isis interface all level 1 disable
set protocols isis interface fxp0.0 disable
set protocols isis interface lo0.0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set ge-2/1/0 unit 0 family inet filter input filter1
user@R2# set ge-2/1/0 unit 0 family inet address 10.0.0.10/30
user@R2# set ge-2/1/0 unit 0 description to-R1
user@R2# set ge-2/1/0 unit 0 family iso

user@R2# set ge-2/1/1 vlan-tagging
user@R2# set ge-2/1/1 description to-R3

user@R2# set ge-2/1/1 unit 0 vlan-id 1001
user@R2# set ge-2/1/1 unit 0 family inet address 10.0.0.13/30
user@R2# set ge-2/1/1 unit 0 family iso

user@R2# set ge-2/1/1 unit 1 vlan-id 1002
user@R2# set ge-2/1/1 unit 1 family inet address 10.0.0.25/30
user@R2# set ge-2/1/1 unit 1 family iso

user@R2# set lo0 unit 0 family inet address 10.255.4.4/32
user@R2# set lo0 unit 0 family iso address 49.0001.0010.0000.0404.00
```

2. Configure the firewall filter.

```
[edit firewall family inet filter filter1]
user@R2# set term t1 from source-address 172.16.1.1/32
user@R2# set term t1 then next-interface ge-2/1/1.0

user@R2# set term t2 from source-address 172.16.2.2/32
user@R2# set term t2 then next-interface ge-2/1/1.1
```

3. Enable IS-IS on the interfaces.

```
[edit protocols is-is]
user@R2# set interface all level 1 disable
user@R2# set interface fxp0.0 disable
user@R2# set interface lo0.0
```

Results

From configuration mode, confirm your configuration by entering the **show interfaces**, **show firewall**, and **show protocols** commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
user@R2# show interfaces
ge-2/1/0 {
  unit 0 {
    description to-R1;
```

```
            family inet {
                filter {
                    input filter1;
                }
                address 10.0.0.10/30;
            }
            family iso;
        }
    }
    ge-2/1/1 {
        description to-R3;
        vlan-tagging;
        unit 0 {
            vlan-id 1001;
            family inet {
                address 10.0.0.13/30;
            }
            family iso;
        }
        unit 1 {
            vlan-id 1002;
            family inet {
                address 10.0.0.25/30;
            }
            family iso;
        }
    }
    lo0 {
        unit 0 {
            family inet {
                address 10.255.4.4/32;
            }
            family iso {
                address 49.0001.0010.0000.0404.00;
            }
        }
    }
```

```
user@R2# show firewall
family inet {
    filter filter1 {
        term t1 {
            from {
                source-address {
                    172.16.1.1/32;
                }
            }
            then {
                next-interface {
                ge-2/1/1.0;
            }
        }
        term t2 {
            from {
                source-address {
                    172.16.2.2/32;
```



```
                }
              }
              then {
                next-interface {
                  ge-2/1/1.1;
                }
              }
            }
          }
        }
```

```
user@R2# show protocols
isis {
  interface all {
    level 1 disable;
  }
  interface fxp0.0 {
    disable;
  }
  interface lo0.0;
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Checking the Paths Used

Purpose Make sure that the expected paths are used when sending traffic from Device R1 to Device R4.

Action On Device R1, enter the **traceroute** command.

```
user@R1> traceroute 10.255.6.6 source 172.16.1.1
traceroute to 10.255.6.6 (10.255.6.6) from 172.16.1.1, 30 hops max, 40 byte packets

 1  10.0.0.10 (10.0.0.10)  0.976 ms  0.895 ms  0.815 ms
 2  10.0.0.14 (10.0.0.14)  0.868 ms  0.888 ms  0.813 ms
 3  10.255.6.6 (10.255.6.6)  1.715 ms  1.442 ms  1.382 ms
```

```
user@R1> traceroute 10.255.6.6 source 172.16.2.2
traceroute to 10.255.6.6 (10.255.6.6) from 172.16.2.2, 30 hops max, 40 byte packets

 1  10.0.0.10 (10.0.0.10)  0.973 ms  0.907 ms  0.782 ms
 2  10.0.0.26 (10.0.0.26)  0.844 ms  0.890 ms  0.852 ms
 3  10.255.6.6 (10.255.6.6)  1.384 ms  1.516 ms  1.462 ms
```

Meaning The output shows that the second hop changes, depending on the source address used in the **traceroute** command.

To verify this feature, a traceroute operation is performed on Device R1 to Device R4. When the source IP address is 172.16.1.1, packets are forwarded out the ge-2/1/1.0 interface on Device R2. When the source IP address is 172.16.2.2, packets are forwarded out the ge-2/1/1.1 interface on Device R2.

Example: Configuring Filter-Based Forwarding to a Specific Destination IP Address

This example shows how to use **then next-ip** as an action in a firewall filter.

Requirements

This example has the following hardware and software requirements:

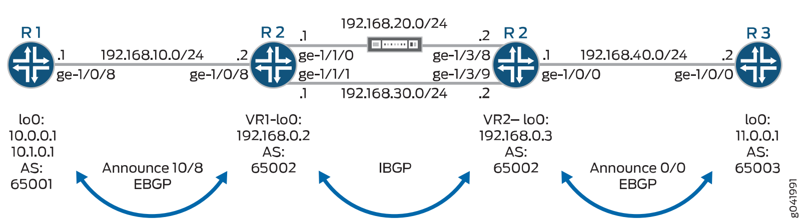
- MX Series 3D Universal Edge Router as the routing device with the firewall filter configured.
- Junos OS Release 12.2 running on the routing device with the firewall filter configured.
- The filter with the **next-interface** (or **next-ip**) action can only be applied to an interface that is hosted on a Trio MPC. If you apply the filter to an I-chip based DPC, the commit operation fails.
- The outgoing interface referred to in the next-interface interface-name action can be hosted on a Trio MPC or an I-chip based DPC.

Overview

In this example, Device R2 has two routing instances that are interconnected with physical links. Traffic from certain sources is required to be directed across the upper link for inspection by a traffic optimizer, which acts transparently on the IP layer. When the traffic optimizer fails, the traffic moves to the lower link. Flows in direction R1>R3 and R3>R1 follow identical paths.

Figure 60 on page 958 shows the topology used in this example.

Figure 60: Filter-Based Forwarding to Specified Outgoing Interfaces



On Device R2, a firewall filter is applied to interface ge-1/0/8 in the input direction. The second term matches the specific source addresses 10.0.0.0/24, and routes the traffic to address 192.168.0.3. This address resolves to next-hop 192.168.20.2. If the link connected to interface ge-1/1/0 goes down, the address 192.168.0.3 will resolve to next-hop 192.168.30.2.

On Device R2, a firewall filter is applied to interface ge-1/0/0 in the input direction. The second term matches the specific destination addresses 10.0.0.0/24, and routes the traffic to address 192.168.0.2. This address resolves to next-hop 192.168.20.1. If the link connected to interface ge-1/3/8 goes down, the address 192.168.0.2 will resolve to next-hop 192.168.30.1.

NOTE: **The address configured using the next-ip action is not automatically resolved. On Ethernet interfaces, it is assumed that the configured address is resolved using a routing protocol or static routes.**

Internal BGP (IBGP) is used between Device R2-VR1 and Device R2-VR2. External BGP (EBGP) is used between Device R1 and Device R2-VR1, as well as between Device R2-VR2 and Device R3.

BGP operations proceed as follows:

- R2-VR1 learns 10/8 from R1, and 0/0 from R2-VR2.
- R2-VR2 learns 0/0 from R3, and 10/8 from R2-VR1.
- R1 advertises 10/8, and receives 0/0 from R2-VR1.
- R3 advertises 0/0, and receives 10/8 from R2-VR2.

The firewall filter applied to Device R2 needs to allow control-plane traffic for the directly connected interfaces, in this case the EBGP sessions.

This example shows the configuration on Device R2.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces lo0 unit 0 family inet address 10.0.0.1/32
set interfaces lo0 unit 0 family inet address 10.1.0.1/32
set interfaces ge-1/0/8 unit 0 family inet address 192.168.10.1/24
set routing-options autonomous-system 64501
set protocols bgp group eBGP neighbor 192.168.10.2 peer-as 64502
set protocols bgp group eBGP export Announce10
set policy-options policy-statement Announce10 term 1 from route-filter 10.0.0.0/8 exact
set policy-options policy-statement Announce10 term 1 then accept
set policy-options policy-statement Announce10 term 2 then reject
```

Device R2

```
set interfaces ge-1/0/8 unit 0 family inet address 192.168.10.2/24
set interfaces ge-1/0/8 unit 0 family inet filter input SteerSrcTrafficOptimizer
```

```
set interfaces ge-1/1/0 unit 0 family inet address 192.168.20.1/24
set interfaces ge-1/1/1 unit 0 family inet address 192.168.30.1/24
set routing-instances VR1 instance-type virtual-router
set routing-instances VR1 interface ge-1/0/8.0
set routing-instances VR1 interface ge-1/1/0.0
set routing-instances VR1 interface ge-1/1/1.0
set routing-instances VR1 routing-options static route 192.168.0.3 next-hop 192.168.20.2
set routing-instances VR1 routing-options static route 192.168.0.3 qualified-next-hop
  192.168.30.2 metric 100
set routing-instances VR1 routing-options autonomous-system 64502
set routing-instances VR1 protocols bgp group eBGP neighbor 192.168.10.1 peer-as 64501
set routing-instances VR1 protocols bgp group iBGP neighbor 192.168.30.2 peer-as 64502
set routing-instances VR1 protocols bgp group iBGP neighbor 192.168.30.2 export
  AcceptExternal
set firewall family inet filter SteerSrcTrafficOptimizer term 0 from source-address
  192.168.10.0/24
set firewall family inet filter SteerSrcTrafficOptimizer term 0 then accept
set firewall family inet filter SteerSrcTrafficOptimizer term 1 from source-address
  10.0.0.0/24
set firewall family inet filter SteerSrcTrafficOptimizer term 1 then next-ip 192.168.0.3
  routing-instance VR1
set firewall family inet filter SteerSrcTrafficOptimizer term 2 from source-address
  10.0.0.0/8
set firewall family inet filter SteerSrcTrafficOptimizer term 2 then accept
set interfaces ge-1/0/0 unit 0 family inet address 192.168.40.1/24
set interfaces ge-1/0/0 unit 0 family inet filter input SteerDstTrafficOptimizer
set interfaces ge-1/3/8 unit 0 family inet address 192.168.20.2/24
set interfaces ge-1/3/9 unit 0 family inet address 192.168.30.2/24
set routing-instances VR2 instance-type virtual-router
set routing-instances VR2 interface ge-1/0/0.0
set routing-instances VR2 interface ge-1/3/8.0
set routing-instances VR2 interface ge-1/3/9.0
set routing-instances VR2 routing-options static route 192.168.0.2/32 next-hop 192.168.20.1
set routing-instances VR2 routing-options static route 192.168.0.2/32 qualified-next-hop
  192.168.30.1 metric 100
set routing-instances VR2 routing-options autonomous-system 64502
set routing-instances VR2 protocols bgp group eBGP neighbor 192.168.40.2 peer-as 64503
set routing-instances VR2 protocols bgp group iBGP neighbor 192.168.30.1 peer-as 64502
set routing-instances VR2 protocols bgp group iBGP neighbor 192.168.30.1 export
  AcceptExternal
set firewall family inet filter SteerDstTrafficOptimizer term 0 from source-address
  192.168.40.0/24
set firewall family inet filter SteerDstTrafficOptimizer term 0 then accept
set firewall family inet filter SteerDstTrafficOptimizer term 1 from destination-address
  10.0.0.0/24
set firewall family inet filter SteerDstTrafficOptimizer term 1 then next-ip 192.168.0.2
  routing-instance VR2
set firewall family inet filter SteerDstTrafficOptimizer term 2 from destination-address
  10.0.0.0/8
set firewall family inet filter SteerDstTrafficOptimizer term 2 then accept
set policy-options policy-statement AcceptExternal term 1 from route-type external
set policy-options policy-statement AcceptExternal term 1 then accept
```

Device R3

```
set interfaces lo0 unit 0 family inet address 10.11.0.1/32
set interfaces ge-1/0/0 unit 0 family inet address 192.168.40.2/24
```



```
set routing-options autonomous-system 64503
set protocols bgp group eBGP neighbor 192.168.40.1 peer-as 64502
set protocols bgp group eBGP export Announce0
set policy-options policy-statement Announce0 term 1 from route-filter 0.0.0.0/0 exact
set policy-options policy-statement Announce0 term 1 then accept
set policy-options policy-statement Announce0 term 2 then reject
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R2:

1. Configure the interfaces.

```
[edit interfaces]
user@R2# set ge-1/0/8 unit 0 family inet address 192.168.10.2/24
user@R2# set ge-1/0/8 unit 0 family inet filter input SteerSrcTrafficOptimizer
user@R2# set ge-1/1/0 unit 0 family inet address 192.168.20.1/24
user@R2# set ge-1/1/1 unit 0 family inet address 192.168.30.1/24

user@R2# set ge-1/0/0 unit 0 family inet address 192.168.40.1/24
user@R2# set ge-1/0/0 unit 0 family inet filter input SteerDstTrafficOptimizer
user@R2# set ge-1/3/8 unit 0 family inet address 192.168.20.2/24
user@R2# set ge-1/3/9 unit 0 family inet address 192.168.30.2/24
```

2. Configure the routing instance.

```
[edit routing-instances]
user@R2# set VR1 instance-type virtual-router
user@R2# set VR1 interface ge-1/0/8.0
user@R2# set VR1 interface ge-1/1/0.0
user@R2# set VR1 interface ge-1/1/1.0

user@R2# set VR2 instance-type virtual-router
user@R2# set VR2 interface ge-1/0/0.0
user@R2# set VR2 interface ge-1/3/8.0
user@R2# set VR2 interface ge-1/3/9.0
```

3. Configure the static and BGP routing.

```
[edit routing-instances]
user@R2# set VR1 routing-options static route 192.168.0.3 next-hop 192.168.20.2
user@R2# set VR1 routing-options static route 192.168.0.3 qualified-next-hop
   192.168.30.2 metric 100
user@R2# set VR1 routing-options autonomous-system 64502
user@R2# set VR1 protocols bgp group eBGP neighbor 192.168.10.1 peer-as 64501
user@R2# set VR1 protocols bgp group iBGP neighbor 192.168.30.2 peer-as 64502
user@R2# set VR1 protocols bgp group iBGP neighbor 192.168.30.2 export
   AcceptExternal

user@R2# set VR2 routing-options static route 192.168.0.2/32 next-hop 192.168.20.1
user@R2# set VR2 routing-options static route 192.168.0.2/32 qualified-next-hop
   192.168.30.1 metric 100
```



```
user@R2# set VR2 routing-options autonomous-system 64502
user@R2# set VR2 protocols bgp group eBGP neighbor 192.168.40.2 peer-as 64503
user@R2# set VR2 protocols bgp group iBGP neighbor 192.168.30.1 peer-as 64502
user@R2# set VR2 protocols bgp group iBGP neighbor 192.168.30.1 export
  AcceptExternal
```

4. Configure the firewall filters.

```
[edit firewall family inet]
user@R2# set filter SteerSrcTrafficOptimizer term 0 from source-address
  192.168.10.0/24
user@R2# set filter SteerSrcTrafficOptimizer term 0 then accept
user@R2# set filter SteerSrcTrafficOptimizer term 1 from source-address 10.0.0.0/24
user@R2# set filter SteerSrcTrafficOptimizer term 1 then next-ip 192.168.0.3
  routing-instance VR1
user@R2# set filter SteerSrcTrafficOptimizer term 2 from source-address 10.0.0.0/8
user@R2# set filter SteerSrcTrafficOptimizer term 2 then accept

user@R2# set filter SteerDstTrafficOptimizer term 0 from source-address
  192.168.40.0/24
user@R2# set filter SteerDstTrafficOptimizer term 0 then accept
user@R2# set filter SteerDstTrafficOptimizer term 1 from destination-address
  10.0.0.0/24
user@R2# set filter SteerDstTrafficOptimizer term 1 then next-ip 192.168.0.2
  routing-instance VR2
user@R2# set filter SteerDstTrafficOptimizer term 2 from destination-address
  10.0.0.0/8
user@R2# set filter SteerDstTrafficOptimizer term 2 then accept
```

5. Configure the routing policy.

```
[edit policy-options policy-statement AcceptExternal term 1]
user@R2# set from route-type external
user@R2# set term 1 then accept
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show firewall**, and **show protocols** commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
user@R2# show interfaces
ge-1/0/0 {
  unit 0 {
    family inet {
      filter {
        input SteerDstTrafficOptimizer;
      }
      address 192.168.40.1/24;
    }
  }
}
ge-1/0/8 {
  unit 0 {
    family inet {
```



```
          filter {
            input SteerSrcTrafficOptimizer;
          }
          address 192.168.10.2/24;
        }
      }
    }
    ge-1/1/0 {
      unit 0 {
        family inet {
          address 192.168.20.1/24;
        }
      }
    }
    ge-1/1/1 {
      unit 0 {
        family inet {
          address 192.168.30.1/24;
        }
      }
    }
    ge-1/3/8 {
      unit 0 {
        family inet {
          address 192.168.20.2/24;
        }
      }
    }
    ge-1/3/9 {
      unit 0 {
        family inet {
          address 192.168.30.2/24;
        }
      }
    }
```

```
user@R2# show firewall
family inet {
  filter SteerSrcTrafficOptimizer {
    term 0 {
      from {
        source-address {
          192.168.10.0/24;
        }
      }
      then accept;
    }
    term 1 {
      from {
        source-address {
          10.0.0.0/24;
        }
      }
      then {
        next-ip 192.168.0.3/32 routing-instance VR1;
      }
```

```
        }
        term 2 {
            from {
                source-address {
                    10.0.0.0/8;
                }
            }
            then accept;
        }
    }
    filter SteerDstTrafficOptimizer {
        term 0 {
            from {
                source-address {
                    192.168.40.0/24;
                }
            }
            then accept;
        }
        term 1 {
            from {
                destination-address {
                    10.0.0.0/24;
                }
            }
            then {
                next-ip 192.168.0.2/32 routing-instance VR2;
            }
        }
        term 2 {
            from {
                destination-address {
                    10.0.0.0/8;
                }
            }
            then accept;
        }
    }
}
```

```
user@R2# show policy-options
policy-statement AcceptExternal {
    term 1 {
        from route-type external;
        then accept;
    }
}
```

```
user@R2# show routing-instances
VR1 {
    instance-type virtual-router;
    interface ge-1/0/8.0;
    interface ge-1/1/0.0;
    interface ge-1/1/1.0;
    routing-options {
        static {
            route 192.168.0.3/32 {
```



```
                next-hop 192.168.20.2;
                qualified-next-hop 192.168.30.2 {
                    metric 100;
                }
            }
        }
        autonomous-system 64502;
    }
    protocols {
        bgp {
            group eBGP {
                neighbor 192.168.10.1 {
                    peer-as 64501;
                }
            }
            group iBGP {
                neighbor 192.168.30.2 {
                    export NextHopSelf;
                    peer-as 64502;
                }
            }
        }
    }
}
VR2 {
    instance-type virtual-router;
    interface ge-1/0/0.0;
    interface ge-1/3/8.0;
    interface ge-1/3/9.0;
    routing-options {
        static {
            route 192.168.0.2/32 {
                next-hop 192.168.20.1;
                qualified-next-hop 192.168.30.1 {
                    metric 100;
                }
            }
        }
        autonomous-system 64502;
    }
    protocols {
        bgp {
            group eBGP {
                neighbor 192.168.40.2 {
                    peer-as 64503;
                }
            }
            group iBGP {
                neighbor 192.168.30.1 {
                    export NextHopSelf;
                    peer-as 64502;
                }
            }
        }
    }
}
```


If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Checking the Paths Used

Purpose Make sure that the expected paths are used when sending traffic from Device R1 to Device R3.

Action On Device R1, enter the **traceroute** command before and after the link failure

Before Failure of the Traffic Optimizer

```
user@R1> traceroute 10.11.0.1 source 10.0.0.1
traceroute to 10.11.0.1 (10.11.0.1) from 10.0.0.1, 30 hops max, 40 byte packets
 1  192.168.10.2 (192.168.10.2)  0.519 ms  0.403 ms  0.380 ms
 2  192.168.20.2 (192.168.20.2)  0.404 ms  0.933 ms  0.402 m0
 3  10.11.0.1 (10.11.0.1)  0.709 ms  0.656 ms  0.644 ms
```

```
user@R1> traceroute 10.11.0.1 source 10.1.0.1
traceroute to 10.11.0.1 (10.11.0.1) from 10.1.0.1, 30 hops max, 40 byte packets
 1  192.168.10.2 (192.168.10.2)  0.524 ms  0.396 ms  0.380 ms
 2  192.168.30.2 (192.168.30.2)  0.412 ms  0.410 ms  0.911 ms
 3  10.11.0.1 (10.11.0.1)  0.721 ms  0.639 ms  0.659 ms
```

After Failure of the Traffic Optimizer

```
user@R1> traceroute 10.11.0.1 source 10.0.0.1
traceroute to 10.11.0.1 (10.11.0.1) from 10.0.0.1, 30 hops max, 40 byte packets
 1  192.168.10.2 (192.168.10.2)  0.506 ms  0.400 ms  0.378 ms
 2  192.168.30.2 (192.168.30.2)  0.433 ms  0.550 ms  0.415 ms
 3  10.11.0.1 (10.11.0.1)  0.723 ms  0.638 ms  0.638 ms
```

```
user@R1> traceroute 10.11.0.1 source 10.1.0.1
traceroute to 10.11.0.1 (10.11.0.1) from 10.1.0.1, 30 hops max, 40 byte packets
 1  192.168.10.2 (192.168.10.2)  0.539 ms  0.411 ms  0.769 ms
 2  192.168.30.2 (192.168.30.2)  0.426 ms  0.413 ms  2.429 ms
 3  10.11.0.1 (10.11.0.1)  10.868 ms  0.662 ms  0.647 ms
```

Meaning The output shows that the second hop changes, depending on the source address used in the **traceroute** command.

To verify this feature, a traceroute operation is performed on Device R1 to Device R3. When the source IP address is 10.0.0.1, packets are forwarded out the ge-1/1/0.0 interface on Device R2. When the source IP address is 10.1.0.1, packets are forwarded out the ge-1/1/1.0 interface on Device R2.

When the link between ge-1/1/0 and ge-1/3/8 fails, packets with source IP address 10.0.0.1 are forwarded out the ge-1/1/1.0 interface on Device R2.

Related Documentation

- Example: Configuring Filter-Based Forwarding on Logical Systems on page 804
- Example: Configuring Filter-Based Forwarding on the Source Address on page 942
- Firewall Filter Nonterminating Actions on page 673

PART 4

Configuring Traffic Policers

CHAPTER 26

Understanding Traffic Policers

Controlling Network Access Using Traffic Policing Overview

This topic covers the following information:

Congestion Management for IP Traffic Flows

Traffic policing, also known as *rate limiting*, is an essential component of network access security that is designed to thwart denial-of-service (DoS) attacks. Traffic policing enables you to control the maximum rate of IP traffic sent or received on an interface and also to partition network traffic into multiple priority levels, also known as *classes of service*. A policer defines a set of traffic rate limits and sets consequences for traffic that does not conform to the configured limits. Packets in a traffic flow that do not conform to traffic limits are either discarded or marked with a different forwarding class or packet loss priority (PLP) level.

With the exception of policers configured to rate-limit aggregate traffic (all protocol families and logical interfaces configured on a physical interface), you can apply a policer to all IP packets in a Layer 2 or Layer 3 traffic flow at a logical interface.

With the exception of policers configured to rate-limit based on physical interface media rate, you can apply a policer to specific IP packets in a Layer 3 traffic flow at a logical interface by using a stateless firewall filter.

You can apply a policer to inbound or outbound interface traffic. Policers applied to inbound traffic help to conserve resources by dropping traffic that does not need to be routed through a network. Dropping inbound traffic also helps to thwart denial-of-service (DoS) attacks. Policers applied to outbound traffic control the bandwidth used.

NOTE: **Traffic policers are instantiated on a per-PIC basis. Traffic policing does not work when the traffic for one local policy decision function (L-PDF) subscriber is distributed over multiple Multiservices PICs in an AMS group.**

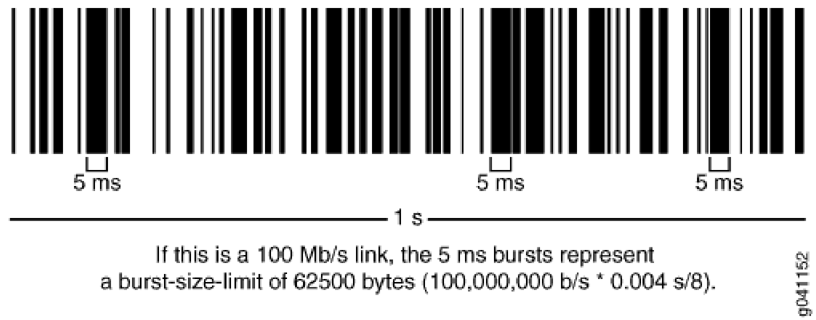
Traffic Limits

Junos OS policers use a *token bucket algorithm* to enforce a limit on an average transmit or receive rate of traffic at an interface while allowing bursts of traffic up to a maximum value based on the configured bandwidth limit and configured burst size. The token bucket algorithm offers more flexibility than a *leaky bucket algorithm* in that you can allow a specified traffic burst before starting to discard packets or apply a penalty such as packet output-queuing priority or packet-drop priority.

In the token-bucket model, the bucket represents the rate-limiting function of the policer. Tokens are added to the bucket at a fixed rate, but once the specified depth of the bucket is reached, tokens allocated after cannot be stored and used. Each token represents a “credit” for some number of bits, and tokens in the bucket are “cashed in” for the ability to transmit or receive traffic at the interface. When sufficient tokens are present in the bucket, a traffic flow continues unrestricted. Otherwise, packets might be dropped or else re-marked with a lower forwarding class, a higher packet loss priority (PLP) level, or both.

- The rate at which tokens are added to the bucket represents the highest average transmit or receive rate in bits per second allowed for a given service level. You specify this highest average traffic rate as the *bandwidth limit* of the policer. If the traffic arrival rate (or fixed bits-per-second) is so high that at some point insufficient tokens are present in the bucket, then the traffic flow is no longer conforming to the traffic limit. During periods of relatively low traffic (traffic that arrives at or departs from the interface at average rates below the token arrival rate), unused tokens accumulate in the bucket.
- The depth of the bucket in bytes controls the amount of back-to-back bursting allowed. You specify this factor as the *burst-size limit* of the policer. This second limit affects the average transmit or receive rate by limiting the number of bytes permitted in a transmission burst for a given interval of time. Bursts exceeding the current burst-size limit are dropped until there are sufficient tokens available to permit the burst to proceed.

Figure 61: Network Traffic and Burst Rates



As shown in the figure above, a UPC bar code is a good facsimile of what traffic looks like on the line; an interface is either transmitting (bursting at full rate) or it is not. The black lines represent periods of data transmission and the white space represents periods of silence when the token bucket can replenish.

Depending on the type of policer used, packets in a policed traffic flow that surpasses the defined limits might be implicitly set to a higher PLP level, assigned to a configured forwarding class or set to a configured PLP level (or both), or simply discarded. If packets encounter downstream congestion, packets with a **low** PLP level are less likely to be discarded than those with a **medium-low**, **medium-high**, or **high** PLP level.

Traffic Color Marking

Based on the particular set of traffic limits configured, a policer identifies a traffic flow as belonging to one of either two or three categories that are similar to the colors of a traffic light used to control automobile traffic.

- *Single-rate two-color*—A two-color marking policer (or "policer" when used without qualification) meters the traffic stream and classifies packets into two categories of packet loss priority (PLP) according to a configured bandwidth and burst-size limit. You can mark packets that exceed the bandwidth and burst-size limit in some way, or simply discard them.

  A policer is most useful for metering traffic at the port (physical interface) level.

- *Single-rate three-color*—This type of policer is defined in RFC 2697, *A Single Rate Three Color Marker*, as part of an assured forwarding (AF) per-hop-behavior (PHB) classification system for a Differentiated Services (DiffServ) environment. This type of policer meters traffic based on the configured committed information rate (CIR), committed burst size (CBS), and the excess burst size (EBS). Traffic is marked as belonging to one of three categories (green, yellow, or red) based on whether the packets arriving are below the CBS (green), exceed the CBS (yellow) but not the EBS, or exceed the EBS (red).

  A single-rate three-color policer is most useful when a service is structured according to packet length and not peak arrival rate.

- *Two-rate three-color*—This type of policer is defined in RFC 2698, *A Two Rate Three Color Marker*, as part of an assured forwarding (AF) per-hop-behavior (PHB) classification system for a Differentiated Services (DiffServ) environment. This type of policer meters traffic based on the configured CIR and peak information rate (PIR), along with their associated burst sizes, the CBS and *peak burst size* (PBS). Traffic is marked as belonging to one of three categories (green, yellow, or red) based on whether the packets arriving are below the CIR (green), exceed the CIR (yellow) but not the PIR, or exceed the PIR (red).

A two-rate three-color policer is most useful when a service is structured according to arrival rates and not necessarily packet length.

Policer actions are implicit or explicit and vary by policer type. The term *Implicit* means that Junos assigns the loss-priority automatically. Table 63 on page 974 describes the policer actions.

Table 63: Policer Actions

Policer	Marking	Implicit Action	Configurable Action
Single-rate two-color	Green (Conforming)	Assign low loss priority	None
	Red (Nonconforming)	None	Assign low or high loss priority, assign a forwarding class, or discard On some platforms, you can assign medium-low or medium-high loss priority
Single-rate three-color	Green (Conforming)	Assign low loss priority	None
	Yellow (Above the CIR and CBS)	Assign medium-high loss priority	None
	Red (Above the EBS)	Assign high loss priority	Discard
Two-rate three-color	Green (Conforming)	Assign low loss priority	None
	Yellow (Above the CIR and CBS)	Assign medium-high loss priority	None
	Red (Above the PIR and PBS)	Assign high loss priority	Discard

Forwarding Classes and PLP Levels

A packet’s forwarding class assignment and PLP level are used by the Junos OS class of service (CoS) features. The Junos OS CoS features include a set of mechanisms that you can use to provide differentiated services when best-effort traffic delivery is insufficient. For router (and switch) interfaces that carry IPv4, IPv6, and MPLS traffic, you can configure CoS features to take in a single flow of traffic entering at the edge of your network and provide different levels of service across the network—internal forwarding and scheduling (queuing) for output—based on the forwarding class assignments and PLP levels of the individual packets.

NOTE: **Forwarding-class or loss-priority assignments performed by a policer or a stateless firewall filter override any such assignments performed on the ingress by the CoS default IP precedence classification at all logical interfaces or by any configured behavior aggregate (BA) classifier that is explicitly mapped to a logical interface.**

Based on CoS configurations, packets of a given forwarding class are transmitted through a specific output queue, and each output queue is associated with a transmission service level defined in a *scheduler*.

Based on other CoS configurations, when packets in an output queue encounter congestion, packets with higher loss-priority values are more likely to be dropped by the random early detection (RED) algorithm. Packet loss priority values affect the scheduling of a packet without affecting the packet's relative ordering within the traffic flow.

Policer Application to Traffic

After you have defined and named a policer, it is stored as a template. You can later use the same policer name to provide the same policer configuration each time you want to use it. This eliminates the need to define the same policer values more than once.

You can apply a policer to a traffic flow in either of two ways:

- You can configure a standard stateless firewall filter that specifies the **policer *policer-name*** nonterminating action or the **three-color-policer (single-rate | two-rate) *policer-name*** nonterminating action. When you apply the standard filter to the input or output at a logical interface, the policer is applied to all packets of the filter-specific protocol family that match the conditions specified in the filter configuration.

  With this method of applying a policer, you can define specific classes of traffic on an interface and apply traffic rate-limiting to each class.
- You can apply a policer directly to an interface so that traffic rate-limiting applies to all traffic on that interface, regardless of protocol family or any match conditions.

You can configure policers at the queue, logical interface, or Layer 2 (MAC) level. Only a single policer is applied to a packet at the egress queue, and the search for policers occurs in this order:

- Queue level
- Logical interface level
- Layer 2 (MAC) level

Related Documentation

- Stateless Firewall Filter Overview on page 558.
- Traffic Policer Types on page 976
- Order of Policer and Firewall Filter Operations on page 979
- *Packet Flow Through the Junos OS CoS Process Overview*

Traffic Policer Types

This topic covers the following information:

Single-Rate Two-Color Policers

You can use a single-rate two-color policer, or “policer” when used without qualification, to rate-limit a traffic flow to an average bits-per-second arrival rate (specified by the single specified bandwidth limit) while allowing bursts of traffic for short periods (controlled by the single specified burst-size limit). This type of policer categorizes a traffic flow as either green (conforming) or red (nonconforming). Packets in a green flow are implicitly set to a **low** loss priority and then transmitted. Packets in a red flow are handled according to actions specified in the policer configuration. Packets in a red flow can be marked—set to a specified forwarding class, set to a specified loss priority, or both—or they can be discarded.

A single-rate two-color policer is most useful for metering traffic at the port (physical interface) level.

Basic Single-Rate Two-Color Policer

You can apply a basic single-rate two-color policer to Layer 3 traffic in either of two ways: as an interface policer or as a firewall filter policer. You can apply the policer as an *interface policer*, meaning that you apply the policer directly to a logical interface at the protocol family level. If you want to apply the policer to selected packets only, you can apply the policer as a *firewall filter policer*, meaning that you reference the policer in a stateless firewall filter term and then apply the filter to a logical interface at the protocol family level.

Bandwidth Policer

A bandwidth policer is simply a single-rate two-color policer that is defined using a bandwidth limit specified as a percentage value rather than as an absolute number of bits per second. When you apply the policer (as an interface policer or as a firewall filter policer) to a logical interface at the protocol family level, the effective bandwidth limit is calculated based on either the physical interface media rate or the logical interface configured shaping rate.

Logical Bandwidth Policer

A logical bandwidth policer is a bandwidth policer for which the effective bandwidth limit is calculated based on the logical interface configured shaping rate. You can apply the policer as a firewall filter policer only, and the firewall filter must be configured as an interface-specific filter. When you apply an interface-specific filter to multiple logical interfaces on supported routing platforms, any **count** or **policer** actions act on the traffic stream entering or exiting each individual interface, regardless of the sum of traffic on the multiple interfaces.

Three-Color Policers

The Junos OS supports two types of three-color policers: single-rate and two-rate. The main difference between a single-rate and a two-rate policer is that the single-rate policer allows bursts of traffic for short periods, while the two-rate policer allows more sustained bursts of traffic. Single-rate policing is implemented using a single token-bucket model, so that periods of relatively low traffic must occur between traffic bursts to allow the token bucket to refill. Two-rate policing is implemented using a dual token-bucket model, which allows bursts of traffic for longer periods.

Single-Rate Three-Color Policers

The single-rate three-color type of policer is defined in RFC 2697, *A Single Rate Three Color Marker*. You use this type of policer to rate-limit a traffic flow to a single rate and three traffic categories (green, yellow, and red). A single-rate three-color policer defines a *committed* bandwidth limit and burst-size limit plus an *excess* burst-size limit. Traffic that conforms to the committed traffic limits is categorized as green (conforming). Traffic that conforms to the bandwidth limit while allowing bursts of traffic as controlled by the excess burst-size limit is categorized as yellow. All other traffic is categorized as red.

A single-rate three-color policer is most useful when a service is structured according to packet length, not peak arrival rate.

Two-Rate Three-Color Policers

The two-rate three-color type of policer is defined in RFC 2698, *A Two Rate Three Color Marker*. You use this type of policer to rate-limit a traffic flow to two rates and three traffic categories (green, yellow, and red). A two-rate three-color policer defines a *committed* bandwidth limit and burst-size limit plus a *peak* bandwidth limit and burst-size limit. Traffic that conforms to the committed traffic limits is categorized as green (conforming). Traffic that exceeds the committed traffic limits but remains below the peak traffic limits is categorized as yellow. Traffic that exceeds the peak traffic limits is categorized as red.

A two-rate three-color policer is most useful when a service is structured according to arrival rates and not necessarily packet length.

Hierarchical Policers

You can use a hierarchical policer to rate-limit ingress Layer 2 traffic at a physical or logical interface and apply different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority output queue. This feature is supported on SONET interfaces hosted on M40e, M120, and M320 edge routers with incoming Flexible PIC Concentrators (FPCs) as SFPC and outgoing FPCs as FFPC, and on T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs.

Two-Color and Three-Color Policer Options

Both two-color and three-color policers can be configured with the following options:

Logical Interface (Aggregate) Policers

A logical interface policer—also called an aggregate policer—is a two-color or three-color policer that you can apply to multiple protocol families on the same logical interface without creating multiple instances of the policer. You apply a logical interface policer directly to a logical interface configuration (and not by referencing the policer in a stateless firewall filter and then applying the filter to the logical interface).

- You can apply the policer at the interface logical unit level to rate-limit all traffic types, regardless of the protocol family.

  When applied in this manner, the logical interface policer will be used by all traffic types (inet, intet6, etc.) and across all layers (layer 2, layer 3) no matter where the policer is attached on the logical interface.

- You can also apply the policer at the logical interface protocol family level, to rate-limit traffic for a specific protocol family.

You can apply a logical interface policer to unicast traffic only. For information about configuring a stateless firewall filter for flooded traffic, see “*Applying Forwarding Table Filters*” in the “Traffic Sampling, Forwarding, and Monitoring” section of the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*.

Physical Interface Policers

A physical interface policer is a two-color or three-color policer that applies to all logical interfaces and protocol families configured on a physical interface, even if the logical interfaces belong to different routing instances. You apply a physical interface policer to a logical interface at the protocol level through a physical interface filter only, but rate limiting is performed aggregately for all logical interfaces and protocol families configured on the underlying physical interface.

This feature enables you to use a single policer instance to perform aggregate policing for different protocol families and different logical interfaces on the same physical interface.

Policers Applied to Layer 2 Traffic

In addition to hierarchical policing, you can also apply single-rate two-color policers and three-color policers (both single-rate and two-rate) to Layer 2 input or output traffic. You must configure the two-color or three-color policer as a logical interface policer and reference the policer in the interface configuration at the logical unit level, and not at the protocol level. You cannot apply a two-color or three-color policer to Layer 2 traffic as a stateless firewall filter action.

Multifield Classification

Like behavior aggregate (BA) classification, which is sometimes referred to as class-of-service (CoS) value traffic classification, multifield classification is a method of classifying incoming traffic by associating each packet with a forwarding class, a packet loss priority level, or both. The CoS scheduling configuration assigns packets to output queues based on forwarding class. The CoS random early detection (RED) process uses the drop probability configuration, output queue fullness percentage, and packet loss priority to drop packets as needed to control congestion at the output stage.

BA classification and multifield classification use different fields of a packet to perform traffic classification. BA classification is based on a *CoS value* in the IP packet header. Multifield classification can be based on *multiple fields* in the IP packet header, including CoS values. Multifield classification is used instead of BA classification when you need to classify packets based on information in the packet other than the CoS values only. Multifield classification is configured using a stateless firewall filter term that matches on any packet header fields and associates matched packets with a forwarding class, a loss priority, or both. The forwarding class or loss priority can be set by a firewall filter action or by a policer referenced as a firewall filter action.

Related Documentation

- Controlling Network Access Using Traffic Policing Overview on page 971
- Order of Policer and Firewall Filter Operations on page 979
- Two-Color Policer Configuration Overview on page 1027
- Three-Color Policer Configuration Overview on page 1115
- Hierarchical Policer Configuration Overview on page 981
- Two-Color Policing at Layer 2 Overview on page 1016
- Three-Color Policing at Layer 2 Overview on page 1018

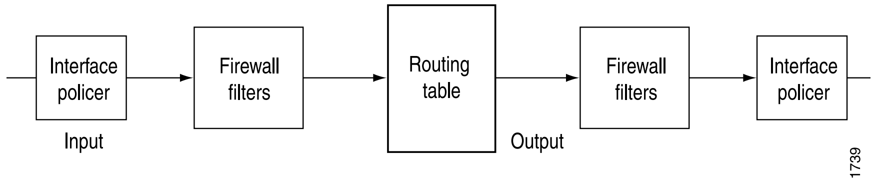
Order of Policer and Firewall Filter Operations

You can apply a both a traffic policer and a stateless firewall filter (with or without policing actions) to a single logical interface at the same time. In this case, the order of precedence of operations is such that policers applied directly to the logical interface are evaluated before input filters but after output filters.

- If an input firewall filter is configured on the same logical interface as a policer, the policer is executed first.
- If an output firewall filter is configured on the same logical interface as a policer, the firewall filter is executed first.

Figure 62 on page 980 illustrates the order of policer and firewall filter processing at the same interface.

Figure 62: Incoming and Outgoing Policers and Firewall Filters



Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Three-Color Policer Configuration Overview on page 1115
- Hierarchical Policer Configuration Overview on page 981

Understanding the Frame Length for Policing Packets

Table 64 on page 980 describes the packet lengths that are considered when you use a traffic policer.

Table 64: Packet Lengths Considered for Traffic Policers

Protocol	Policing Packet Lengths
Any	L3 frame including header
IPv4	L3 frame including header
IPv6	L3 frame including header
MPLS	L3 frame including header
VPLS	Preamble/SFD, IFG and MAC, and FCS
Bridge	Preamble/SFD, IFG and MAC, and FCS
CCC	Preamble/SFD, IFG and MAC, and FCS

Related Documentation

- Policer Overhead to Account for Rate Shaping in the Traffic Manager on page 1106

Supported Standards for Policing

Three-color policers are part of an assured forwarding (AF) per-hop-behavior (PHB) classification system for a Differentiated Services (DiffServ) environment, which is described and defined in the following RFCs:

- RFC 2474, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
- RFC 2475, *An Architecture for Differentiated Service*

- RFC 2597, *Assured Forwarding PHB Group*
- RFC 2598, *An Expedited Forwarding PHB*
- RFC 2698, *A Two Rate Three Color Marker*

In a DiffServ environment, the most significant 6 bits of the type-of-service (ToS) octet in the IP header contain a value called the *Differentiated Services code point* (DSCP). Within the DSCP field, the most significant 3 bits are interpreted as the *IP precedence* field, which can be used to select different per-hop forwarding treatments for the packet.

Hierarchical Policer Configuration Overview

Hierarchically rate-limits Layer 2 ingress traffic for all protocol families. Cannot be applied to egress traffic, Layer 3 traffic, or at a specific protocol level of the interface hierarchy.

Supported on the following interfaces:

- SONET interfaces hosted on M40e, M120, and M320 edge routers with incoming FPCs as SFPC and outgoing FPCs as FFPC.
- SONET interfaces hosted on T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs.
- Ethernet interfaces on Gigabit Ethernet Intelligent Queuing 2 (IQ2) and Ethernet Enhanced IQ2 (IQ2E) PICs.
- MX Series routers with MPC or DPC.

Table 65 on page 981 describes the hierarchy levels at which you can configure and apply hierarchical policers.

Table 65: Hierarchical Policer Configuration and Application Summary

Policer Configuration	Layer 2 Application	Key Points
Hierarchical Policer		

Aggregate and premium policing components of a hierarchical policer:
 	
[edit firewall]
hierarchical-policer *policer-name* {
  aggregate {
    if-exceeding {
      bandwidth-limit *bps*;
      burst-size-limit *bytes*;
    }
    then {
      discard;
      forwarding-class *class-name*;
      loss-priority *supported-value*;
    }
  }
  premium {
	Option A—Apply directly to Layer 2 input traffic on a physical interface:
	
	[edit interfaces]
	interface-name {
	  layer2-policer {
	    input-hierarchical-policer *policer-name*;
	  }
	}
		Hierarchically rate-limit Layer 2 ingress traffic for all protocol families and logical interfaces configured on a physical interface.
		
		Include the **layer2-policer** configuration statement at the **[edit interfaces *interface-name*]** hierarchy level.
		
		NOTE: If you apply a hierarchical policer at a physical interface, you cannot also apply a hierarchical policer to any of the member logical interfaces.

Table 65: Hierarchical Policer Configuration and Application Summary *(continued)*

Policer Configuration	Layer 2 Application	Key Points

if-exceeding {
  bandwidth-limit *bps*;
  burst-size-limit *bytes*;
}
then {
  discard;
}
}
}
	Option B—Apply directly to Layer 2 input traffic on a logical interface.
	
	[edit interfaces]
	interface-name {
	  unit *unit-number* {
	    layer2-policer {
	      input-hierarchical-policer *policer-name*;
	    }
	  }
	}
		Hierarchically rate-limit Layer 2 ingress traffic for all protocol families configured on a specific logical interface.
		
		Include the **layer2-policer** configuration statement at the **[edit interfaces *interface-name* unit *unit-number*]** hierarchy level.
		
		NOTE: You must configure at least one protocol family for the logical interface.

Related Documentation

- Hierarchical Policers on page 1009

Packets-Per-Second (pps)-Based Policer Overview

In a modern network environment, both denial-of-service (DoS) and distributed denial-of-service (DDoS) attacks are very common. Over time, these attacks have evolved from brute force types of attacks, where the attacker might try to overrun a connection's available bandwidth with a vast amount of directed traffic to more low-and-slow attacks that use smaller packets, sent at a slower rate to target specific resources in order to deny service.

Traffic policers, both interface-based and filter-based, have been available to mitigate brute force types of DDoS attacks since Junos OS Release 9.6. These policers operate by limiting the traffic rate through a logical interface or by limiting the traffic rate as the "nonterminating action" on page 673 within a firewall filter.

In Junos OS Release 15.1 and earlier releases, there were two parameters available for policers: bandwidth and burst-size. The unit of measure for the bandwidth parameter is bits per second (bps), and the unit of measure for the burst-size parameter is bytes (B). See "Policer Bandwidth and Burst-Size Limits" on page 989 for details. Policers defined within these parameters are not effective at stopping low-and-slow types of DDoS attacks.

Starting in Junos OS Release 16.1, traffic policers can be defined using packets per second (pps) with the **pps-limit** and **packet-burst** statements. The unit of measure for **pps-limit** is packets per second (pps), and the unit of measure for **packet-burst** is packets. These pps-based policers are more effective at mitigating low-and-slow types of DDoS attacks.

Policers configured with the **if-exceeding-pps** statement are applied in the same manner and in the same locations as bandwidth-based policers. Pps-based policers cannot be applied simultaneously with bandwidth-based policers. Only one policer can be applied at a time except for hierarchical policers, which allow the configuration of two policing actions based on traffic classification.

Related Documentation

- *if-exceeding-pps*
- pps-limit on page 1325
- packet-burst on page 1314

Guidelines for Applying Traffic Policers

The following general guidelines pertain to applying traffic policers:

- Only one type of policer can be applied to the input or output of the same physical or logical interface. For example, you are not allowed to apply a policer and a hierarchical policer in the same direction at the same logical interface.
- Chaining of policers—that is, applying policers to both a port and the logical interfaces of that port—is not allowed.
- A maximum of 64 policers is supported per physical or logical interface, provided no behavior aggregate (BA) classification—traffic classification based on CoS values in the packet headers—is applied to the logical interface.
- The policer should be independent of BA classification. Without BA classification, all traffic on an interface is treated either as expedited forwarding (EF) or non-EF, based on the configuration. With BA classification, a physical or logical interface can support up to 64 policers. The interface might be a physical interface or logical interface.
- With BA classification, the miscellaneous traffic (the traffic *not* matching any of the BA classification DSCP/EXP bits) is policed as non-EF traffic. No separate policers are installed for this traffic.
- Policers can be applied to unicast packets only. For information about configuring a filter for flooded traffic, see *Applying Forwarding Table Filters*.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Three-Color Policer Configuration Overview on page 1115
- Hierarchical Policer Configuration Overview on page 981

Policer Support for Aggregated Ethernet Interfaces Overview

Aggregated interfaces support single-rate policers, three-color marking policers, two-rate three-color marking policers, hierarchical policers, and percentage-based policers. By default, policer bandwidth and burst-size applied on aggregated bundles is not matched to the user-configured bandwidth and burst-size.

You can configure interface-specific policers applied on an aggregated Ethernet bundle or an aggregated SONET bundle to match the effective bandwidth and burst-size to user-configured values. The **shared-bandwidth-policer** statement is required to achieve this match behavior.

This capability applies to all interface-specific policers of the following types: single-rate policers, single-rate three-color marking policers, two-rate three-color marking policers, and hierarchical policers. Percentage-based policers match the bandwidth to the user-configured values by default, and do not require shared-bandwidth-policer configuration. The **shared-bandwidth-policer** statement causes a split in burst-size for percentage-based policers.

NOTE: This feature is supported on the following platforms: T Series routers (excluding T4000 Type 5 FPCs), M120, M10i, M7i (CFEB-E only), M320 (SFPC only), MX240, MX480, and MX960 with DPC, MIC, and MPC interfaces, and EX Series switches.

The following usage scenarios are supported:

- Interface policers used by the following configuration:

  [edit] interfaces (ae*X* | as*X*) unit *unit-num* **family** *family* **policer [input | output | arp]**

- Policers and three-color policers (both single-rate three-color marking and two-rate three-color marking) used inside interface-specific filters; that is, filters that have an interface-specific keyword and are used by the following configuration:

  [edit] interfaces (ae*X* | as*X*) unit *unit-num* **family** *family* **filter [input | output]**

- Common-edge service filters, which are derived from CLI-configured filters and thus inherit interface-specific properties. All policers and three-color policers used by these filters are also affected.

The following usage scenarios are not supported:

- Policers and three-color policers used inside filters that are not interface specific; such a filter is meant to be shared across multiple interfaces.
- Any implicit policers or policers that are part of implicit filters; for example, the default ARP policer applied to an aggregate Ethernet interface. Such a policer is meant to be shared across multiple interfaces.
- Prefix-specific action policers.

To configure this feature, include the **shared-bandwidth-policer** statement at the following hierarchy levels: **[edit firewall policer *policer-name*]**, **[edit firewall three-color-policer *policer-name*]**, or **[edit firewall hierarchical-policer *policer-name*]**.

Related Documentation

- shared-bandwidth-policer on page 1330

Firewall and Policing Differences Between PTX Series Packet Transport Routers and T Series Matrix Routers

This topic provides a list of firewall and policier features available on PTX Packet Transport Routers and compares them with firewall and policing features on T Series routers.

Firewall Filters

Junos OS firewall and policing software on PTX Series Packet Transport Routers supports IPv4 filters, IPv6 filters, MPLS filters, CCC filters, interface policing, LSP policing, MAC filtering, ARP policing, L2 policing, and other features. Exceptions are noted below.

- PTX Series Packet Transport Routers do not support:
  - Egress Forwarding Table Filters
  - Forwarding Table Filters for MPLS/CCC
  - Family VPLS
- PTX Series Packet Transport Routers do not support nested firewall filters. The **filter** statement at the **[edit firewall family *family-name* filter *filter-name* term *term-name*]** hierarchy level is disabled.
- Because no service PICs are present in PTX Series Packet Transport Routers, service filters are not supported for both IPv4 and IPv6 traffic. The **service-filter** statement at **[edit firewall family (inet | inet6)]** hierarchy level is disabled.
- The PTX Series Packet Transport Routers exclude simple filters. These filters are supported on Gigabit Ethernet intelligent queuing (IQ2) and Enhanced Queuing Dense Port Concentrator (EQ DPC) interfaces only. The **simple-filter** statement at the **[edit firewall family inet)]** hierarchy level is disabled.
- Physical interface filtering is not supported. The **physical-interface-filter** statement at the **[edit firewall family *family-name* filter *filter-name*]** hierarchy level is disabled.
- The prefix action feature is not supported on PTX Series Packet Transport Routers. The **prefix-action** statement at **[edit firewall family inet]** hierarchy level is disabled.
- On T Series routers, you can collect a variety of information about traffic passing through the device by setting up one or more accounting profiles that specify some common characteristics of the data. The PTX Series Packet Transport Routers do not support accounting configurations for firewall filters. The **accounting-profile** statement at the **[edit firewall family *family-name* filter *filter-name*]** hierarchy level is disabled.
- The **reject** action is not supported on the loopback (**lo0**) interface. If you apply a filter to the **lo0** interface and the filter includes a **reject** action, an error message appears.

- PTX Series Packet Transport Routers do not support aggregated ethernet logical interface match conditions. However, child link interface matching is supported.
- PTX Series Packet Transport Routers displays both counts if two different terms in a filter have the same match condition but they have different counts. T Series routers display one count only.
- PTX Series Packet Transport Routers do not have separate policer instances when a filter is bound to multiple interfaces. Use the **interface-specific** configuration statement to create the configuration.
- On PTX Series Packet Transport Routers, when an ingress interface has CCC encapsulation, packets coming in through the ingress CCC interface will not be processed by the egress filters.
- For CCC encapsulation, the PTX Series Packet Transport Routers append an extra 8 bytes for egress Layer 2 filtering. The T Series routers do not. Therefore, egress counters on PTX Series Packet Transport Routers show an extra eight bytes for each packet which impacts policer accuracy.
- On PTX Series Packet Transport Routers, output for the **show pfe statistics traffic** CLI command includes the packets discarded by DMAC and SMAC filtering. On T Series routers, the command output does not include these discarded packets because MAC filters are implemented in the PIC and not in the FPC.
- The last-fragment packet that goes through a PTX firewall cannot be matched by the **is-fragment** matching condition. This feature is supported on T Series routers.

  A possible workaround on PTX Series Packet Transport Routers is to configure two separate terms with same the actions: one term contains a match to **is-fragment** and the other term contains a match to **fragment-offset -except 0**.
- On PTX Series Packet Transport Routers, MAC pause frames are generated when packet discards exceed 100 Mbps. This occurs only for frame sizes that are less than 105 bytes.

Traffic Policiers

Junos OS firewall and policing software on PTX Series Packet Transport Routers supports IPv4 filters, IPv6 filters, MPLS filters, CCC filters, interface policing, LSP policing, MAC filtering, ARP policing, L2 policing, and other features. Exceptions are noted below.

- PTX Series Packet Transport Routers support ARP policing. T Series routers do not.
- PTX Series Packet Transport Routers do not support LSP policing.
- PTX Series Packet Transport Routers do not support the **hierarchical-policer** configuration statement. .
- PTX Series Packet Transport Routers do not support the **interface-set** configuration statement. This statement groups a number of interfaces into a single, named interface set.
- PTX Series Packet Transport Routers do not support the following policer types for both normal policers and three-color policers:

    - **logical-bandwidth-policer** — Policer uses logical interface bandwidth.
    - **physical-interface-policer** — Policer is a physical interface policer.
    - **shared-bandwidth-policer** — Share policer bandwidth among bundle links.

- When a policer action and forwarding-class, loss-priority actions are configured within the same rule (a *Multifield Classification*), the PTX Series Packet Transport Routers work differently than T Series routers. As shown below, you can configure two rules in the filter to make the PTX filter behave the same as the T Series filter:

  PTX Series configuration:

  ```
  rule-1 {
     match: {x, y, z}
     action: {forwarding-class, loss-prio, next}
  }
     rule-2 {
        match: {x, y, z}
        action: {policer}
     }
  ```

  T Series configuration:

  ```
  rule-1 {
     match: {x, y, z}
     action: {forwarding-class, loss-prio, policer}
  }
  ```

Related Documentation

- *Junos OS Firewall Filters and Traffic Policers Library for Routing Devices*

CHAPTER 27

Configuring Policer Rate Limits and Actions

Policer Bandwidth and Burst-Size Limits

Table 66 on page 989 lists each of the Junos OS policer types supported. For each policer type, the table summarizes the bandwidth limits and burst-size limits used to rate-limit traffic.

Table 66: Policer Bandwidth Limits and Burst-Size Limits

Policer Type	Bandwidth Limits	Burst-Size Limits
Single-Rate Two-Color Policer		
Defines a single rate limit: a bandwidth limit and an allowed burst size for conforming traffic. For a single-rate two-color policer only, you can specify the bandwidth limit as a percentage value from 1 through 100 instead of as an absolute number of bits per second. The effective bandwidth limit is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.	**bandwidth-limit** ***bps***; M and T Series routers: 8000..100000000000 MX Series routers: 8000..18446744073709551615 **bandwidth-percent** 1..100 percent	**burst-size-limit** ***bytes***; M, MX, and T Series routers: 1500..10000000000
Single-Rate Three-Color Policer		
Defines a single rate limit: a bandwidth limit and an allowed burst size for conforming traffic. Also defines a second, larger burst size. This second burst size is used to differentiate between two categories of nonconforming traffic (yellow or red).	**committed-information-rate** ***bps***; M and T Series routers: 1500..100000000000 MX Series routers: 8000..18446744073709551615	**committed-burst-size** ***bytes***; M, MX, and T Series routers: 1500..100000000000 **excess-burst-size** ***bytes***; M, MX, and T Series routers:

Table 66: Policer Bandwidth Limits and Burst-Size Limits *(continued)*

Policer Type	Bandwidth Limits	Burst-Size Limits
		1500..100000000000
Two-Rate Three-Color Policer		
Defines a committed rate limit: a bandwidth limit and an allowed burst size for conforming traffic. Also defines a peak rate limit: a second, larger burst size and a second, higher bandwidth limit. These additional rate-limit components are used to differentiate between two categories of nonconforming traffic (yellow or red).	**committed-information-rate** *bps*; M and T Series routers: 1500..100000000000 MX Series routers: 8000..18446744073709551615	**committed-burst-size** *bytes*; M, MX, and T Series routers: 1500..100000000000
	peak-information-rate *bps*; M and T Series routers: 1500..100000000000 MX Series routers: 8000..18446744073709551615	**peak-burst-size** *bytes*; M, MX, and T Series routers: 1500..100000000000
Hierarchical Policer		
Defines two policers, each with a bandwidth limit and an allowed burst size for conforming traffic. Different policing actions are applied based on whether the packets are classified for expedited forwarding (EF) or for a lower priority. Rate-limits ingress Layer 2 traffic at a SONET physical or logical interface hosted on supported routing platforms only.	**bandwidth-limit** *bps*; M40e, M120, and M320 (with FFPC and SFPC) edge routers and T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs: 32000..50000000000 MX Series routers: 8000..18446744073709551615	**burst-size-limit** *bytes*; M40e, M120, and M320 (with FFPC and SFPC) edge routers and T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs: 1500..2147450880

Related Documentation

- Policer Color-Marking and Actions on page 990
- Determining Proper Burst Size for Traffic Policers on page 1002

Policer Color-Marking and Actions

Table 67 on page 990 lists each of the Junos OS policer types supported. For each policer type, the table summarizes the color-marking criteria used to categorize a traffic flow and, for each color, the actions taken on packets in that type of traffic flow.

Table 67: Implicit and Configurable Policer Actions Based on Color Marking

Policer Rate Limits and Color Marking	Implicit Action	Configurable Actions
Single-Rate Two-Color Policer		
• Bandwidth limit • Burst size		

Table 67: Implicit and Configurable Policer Actions Based on Color Marking *(continued)*

Policer Rate Limits and Color Marking	Implicit Action	Configurable Actions
Green Conforms to rate and burst size limits	Set PLP to **low**	–
Red Exceeds rate and burst size limits	–	• Discard the packet. • Assign to a forwarding class. • Set PLP to **low** or **high**. On some platforms, you can also set the PLP to **medium-low** or **medium-high**.
Single-Rate Three-Color Policer • Committed information rate (CIR) • Committed burst size (CBS) • Excess burst size (EBS)		
Green Conforms to the CIR and CBS	Set PLP to **low**	–
Yellow Exceeds the CIR and CBS but conforms to the EBS	Set PLP to **medium-high**	–
Red Exceeds the EBS	Set PLP to **high**	• Discard the packet.
Two-Rate Three-Color Policer • Committed information rate (CIR) • Committed burst size (CBS) • Peak information rate (PIR) • Peak burst size (PBS)		
Green Conforms to the CIR and CBS	Set PLP to **low**	–
Yellow Exceeds the CIR and CBS, but conforms to the PIR	Set PLP to **medium-high**	–
Red Exceeds the PIR and PBS	Set PLP to **high**	• Discard the packet.
Hierarchical Policer		
Aggregate policer • Bandwidth limit • Burst size		

Table 67: Implicit and Configurable Policer Actions Based on Color Marking *(continued)*

Policer Rate Limits and Color Marking	Implicit Action	Configurable Actions
Green Conforms to rate limits	Set PLP to **low**	–
Red Exceeds rate limits	–	• Discard the packet. • Assign to a forwarding class. • Set PLP to **low** or **high**. On some platforms, you can also set the PLP to **medium-low** or **medium-high**.
Premium policer • Bandwidth limit • Burst size		
Green Conforms to rate limits	Set PLP to **low**	–
Red Exceeds rate limits	–	• Discard the packet.

Related Documentation

- Policer Bandwidth and Burst-Size Limits on page 989
- Determining Proper Burst Size for Traffic Policers on page 1002

Single Token Bucket Algorithm

This topic covers the following information:

- Token Bucket Concepts on page 992
- Single Token Bucket Algorithm on page 992
- Conformance Measurement for Two-Color Marking on page 993

Token Bucket Concepts

When you apply traffic policing to the input or output traffic at an interface, the rate limits and actions specified in the policer configuration are used to enforce a limit on the average throughput rate at the interface while also allowing bursts of traffic up to a maximum number of bytes based on the overall traffic load. Junos OS policers measure traffic-flow conformance to a policing rate limit by using a *token bucket algorithm*. An algorithm based on a single token bucket allows burst of traffic for short periods, whereas an algorithm based dual token buckets allows more sustained bursts of traffic.

Single Token Bucket Algorithm

A single-rate two-color policer limits traffic throughput at an interface based on how the traffic conforms to rate-limit values specified in the policer configuration. Similarly, a hierarchical policer limits traffic throughput at an interface based on how aggregate and premium traffic subflows conform to aggregate and premium rate-limit values specified in the policer configuration. For both two-color policer types, packets in a conforming traffic flow are categorized as *green*, and packets in a non-conforming traffic flow are categorized as *red*.

The single token bucket algorithm measures traffic-flow conformance to a two-color policer rate limit as follows:

- The token arrival rate represents the single *bandwidth limit* configured for the policer. You can specify the bandwidth limit as an absolute number of bits per second by including the **bandwidth-limit *bps*** statement. Alternatively, for single-rate two-color policers only, you can use the **bandwidth-percent *percentage*** statement to specify the bandwidth limit as a percentage of either the physical interface port speed or the configured logical interface shaping rate.
- The token bucket depth represents the single *burst size* configured for the policer. You specify the burst size by including the **burst-size-limit *bytes*** statement.
- If the bucket is filled to capacity, arriving tokens “overflow” the bucket and are lost.

When the bucket contains insufficient tokens for receiving or transmitting the traffic at the interface, packets might be dropped or else re-marked with a lower forwarding class, a higher packet loss priority (PLP) level, or both.

Conformance Measurement for Two-Color Marking

In two-color-marking policing, a traffic flow whose average arrival or departure rate does not exceed the token arrival rate (bandwidth limit) is considered *conforming traffic*. Packets in a conforming traffic flow (categorized as green traffic) are implicitly marked with a packet loss priority (PLP) level of **low** and then passed through the interface.

For a traffic flow whose average arrival or departure rate exceeds the token arrival rate, conformance to a two-color policer rate limit depends on the tokens in the bucket. If sufficient tokens remain in the bucket, the flow is considered conforming traffic. If the bucket does not contain sufficient tokens, the flow is considered *non-conforming traffic*. Packets in a non-conforming traffic flow (categorized as red traffic) are handled according to policing actions. Depending on the configuration of the two-color policer, packets might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface.

NOTE: **The number of tokens remaining in the bucket at any given time is a function of the token bucket depth and the overall traffic load.**

The token bucket is initially filled to capacity, and so the policer allows an initial traffic burst (back-to-back traffic at average rates that exceed the token arrival rate) up to the size of the token bucket depth.

During periods of relatively low traffic (traffic that arrives at or departs from the interface at average rates below the token arrival rate), unused tokens accumulate in the bucket, but only up to the configured token bucket depth.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Hierarchical Policer Configuration Overview on page 981
- Policer Color-Marking and Actions on page 990
- **bandwidth-limit (Hierarchical Policer) on page 1274**
- **bandwidth-limit (Policer) on page 1276**
- **bandwidth-percent on page 1278**
- **burst-size-limit (Hierarchical Policer) on page 1280**
- **burst-size-limit (Policer) on page 1281**

Dual Token Bucket Algorithms

This topic covers the following information:

Token Bucket Concepts

When you apply traffic policing to the input or output traffic at an interface, the rate limits and actions specified in the policer configuration are used to enforce a limit on the average throughput rate at the interface while also allowing bursts of traffic up to a maximum number of bytes based on the overall traffic load. Junos OS policers measure traffic-flow conformance to a policing rate limit by using a *token bucket algorithm*. An algorithm based on a single token bucket allows burst of traffic for short periods, whereas an algorithm based dual token buckets allows more sustained bursts of traffic.

Guaranteed Bandwidth for Three-Color Marking

A committed information rate (CIR) defines the guaranteed bandwidth for traffic arriving at or departing from the interface under normal line conditions. A flow of traffic at an average rate that conforms to the CIR is categorized green, and packets in a green flow are implicitly marked with **low** packet loss priority (PLP) and then passed through the interface. During periods of relatively low traffic (traffic that arrives at or departs from the interface at average rates below the CIR), any unused bandwidth capacity accumulates in the first token bucket, but only up to a configured number of bytes. If any unused bandwidth capacity overflows the first bucket, the excess accumulates in a second token bucket.

The committed burst size (CBS) defines the maximum number of bytes for which unused amounts of the guaranteed bandwidth can be accumulated in the first token bucket. A burst of traffic at an average rate that exceeds the CIR is also categorized as green provided that sufficient unused bandwidth capacity is available in the first token bucket.

Nonconformance Measurement for Single-Rate Three-Color Marking

Single-rate three-color policer configurations specify a second burst size—the excess burst size (EBS)—that defines the maximum number of bytes for which the second token bucket can accumulate unused bandwidth that overflows from the first bucket.

A traffic flow is categorized yellow if its average rate exceeds the CIR and the available bandwidth capacity accumulated in the first bucket if sufficient unused bandwidth capacity is available in the second token bucket. Packets in a yellow flow are implicitly marked with **medium-high** PLP and then passed through the interface.

A traffic flow is categorized red its average rate exceeds the CIR and the available bandwidth capacity accumulated in the second bucket. Packets in a red flow are implicitly marked with **high** PLP and then either passed through the interface or optionally discarded.

Nonconformance Measurement for Two-Rate Three-Color Marking

Two-rate three-color policer configurations include a second rate limit—the peak-information-rate (PIR)—that you set to the expected average data rate for traffic arriving at or departing from the interface under peak conditions.

Two-rate three-color policer configurations also include a second burst size—the peak burst size (PBS)—that defines the maximum number of bytes for which the second token bucket can accumulate unused peak bandwidth capacity. During periods of relatively little peak traffic (traffic that arrives at or departs from the interface at average rates that exceed the PIR), any unused peak bandwidth capacity accumulates in the second token bucket, but only up to the maximum number of bytes specified by the PBS.

A traffic flow is categorized yellow if it exceeds the CIR and the available committed bandwidth capacity accumulated in the first token bucket but conforms to the PIR. Packets in a yellow flow are implicitly marked with **medium-high** PLP and then passed through the interface.

A traffic flow is categorized red if it exceeds the PIR and the available peak bandwidth capacity accumulated in the second token bucket. Packets in a red flow are implicitly marked with **high** PLP and then either passed through the interface or optionally discarded.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Policer Color-Marking and Actions on page 990
- **committed-burst-size on page 1286**
- **committed-information-rate on page 1288**
- **excess-burst-size on page 1291**
- **peak-burst-size on page 1316**

- peak-information-rate on page 1318

CHAPTER 28

Implementing Traffic Policers on MX Series, M120, and M320 Routers

Policer Implementation Overview

The Juniper Networks® Junos® operating system (Junos OS) supports three types of policers:

- *Single-rate two-color policer* — The most common policer. Single-rate means that there is only a single bandwidth and burst rate referenced in the policer. The two colors associated with this policer are red (nonconforming) and green (conforming).
- *Single-rate three-color policer* — Similar to the single-rate two-color policer with the addition of the color yellow. This type also introduces the *committed information rate* (CIR) and a *committed burst rate* (CBR).
- *Two-rate three-color policer* — Builds off of the single-rate three-color policer by adding a second rate tier. *Two-rate* means there is an upper bandwidth limit and associated burst size as well as a *peak information rate* (PIR) and a *peak burst rate* (PBS).

There are two types of token bucket algorithms that can be used, depending on the type of policer that is applied to network traffic. Single-rate two-color policers use the *single token bucket algorithm* to measure traffic flow conformance to a two-color policer rate limit. Single-rate three-color policers and two-rate three-color policers both use the *dual token bucket algorithm* to measure traffic flow conformance to a three-color policer rate. The main difference between these two token bucket algorithms is that the single token bucket algorithm allows bursts of traffic for short periods, whereas the dual token bucket algorithm allows more sustained bursts of traffic. (The remainder of this topic discusses the single token bucket algorithm.)

To configure a policer, you need to set two parameters:

- Bandwidth limit configured in bps (using the **bandwidth-limit** statement)
- Burst size configured in bytes (using the **burst-size-limit** statement)

NOTE: For single-rate two-color policers only, you can also specify the bandwidth limit as a percentage of either the physical interface port speed or the configured logical interface shaping rate by using the **bandwidth-percent** *percentage* statement. You cannot configure a policer to use bandwidth percentage for aggregate, tunnel, or software interfaces.

Use the following command to set the policer conditions:

```
user@router# set firewall policer <policer name> if-exceeding ?
Possible completions:
  <[Enter]>            Execute this command
+ apply-groups         Groups from which to inherit configuration data
+ apply-groups-except  Don't inherit configuration data from these groups
  bandwidth-limit    Bandwidth limit (8000..100000000000 bits per second)
  bandwidth-percent    Bandwidth limit in percentage (1..100 percent)
  burst-size-limit   Burst size limit (1500..100000000000 bytes)
  |                    Pipe through a command
```

The bandwidth limit parameter is used to determine the average rate limit applied to the traffic, while the burst-size parameter is used to allow for short periods of traffic bursting (back-to-back traffic at average rates that exceed the configured bandwidth limit). Once you apply a set of policer configuration settings (bandwidth limit and burst size), the configured values are adjusted to hardware programmable values. The conversion adjustment introduced is normally less than 1 percent of the configured bandwidth limit. This adjustment is needed because the software allows you to configure the bandwidth limit and burst size to any value within the specified ranges, but those values must be adjusted to the nearest value that can be programmed in the hardware.

The policer bandwidth limit configuration in the hardware is represented by two values: the *credit update frequency* and the *credit size*. The credit update frequency is used by the hardware to determine how frequently tokens (bits of unused bandwidth) are added to the token bucket. The credit size is based on the number of tokens that can fit in the token bucket. The MX Series, M120, M320 routers, and EX Series switches contain a set of credit update frequencies instead of having a single credit update frequency to minimize the adjustment difference from the configured bandwidth limit and to support a wide range of policer bandwidth rates (from 40 Kbps to 40 Gbps). One of the frequencies is used to program the policer (bandwidth limit and burst size) in the hardware.

The burst size is based on the overall traffic load and allows bursts of traffic to exceed the configured bandwidth limit. A policer with a large burst size effectively disables the configured bandwidth limit function, so the burst size must be relative to the configured bandwidth limit. You need to consider the traffic patterns in your network before determining the burst size. For more information about determining burst size, see "Determining Proper Burst Size for Traffic Policers" on page 1002.

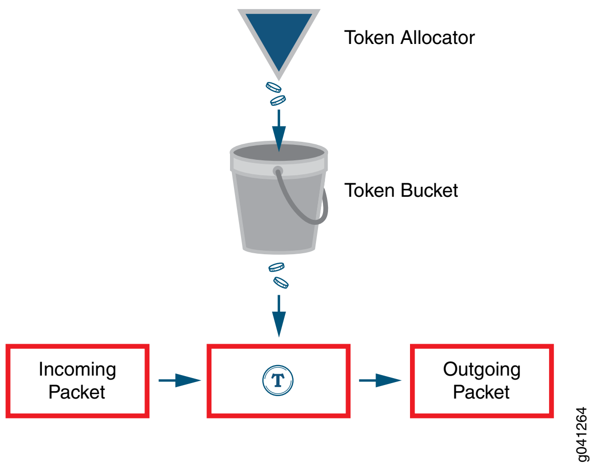
The configured burst size is adjusted in the hardware to a value that is based on the configured bandwidth limit. The burst size extends the configured bandwidth limit for bursty traffic that exceeds the configured bandwidth limit.

When a policer is applied to the traffic at an interface, the initial capacity for traffic bursting is equal to the number of bytes specified in the **burst-size-limit** statement.

Figure 63 on page 999 represents how a policer is implemented using the token bucket algorithm. The token allocator allocates tokens to the policer based on the configured bandwidth limit, which is the token size multiplied by the token arrival rate.

token size x token arrival rate = policer rate (configured bandwidth limit)

Figure 63: Token Bucket Algorithm

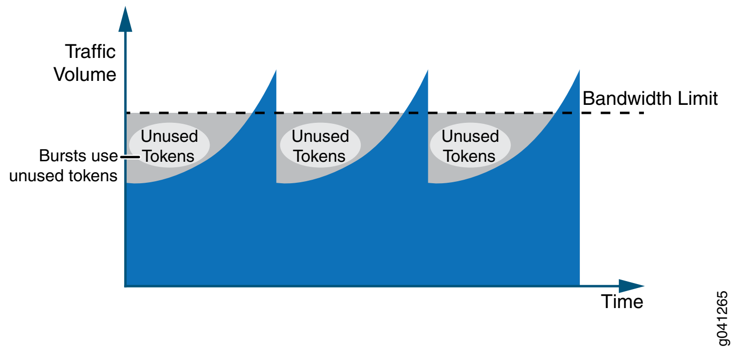


When a packet arrives at an interface configured with a policer, tokens that represent the number of bits that correspond to the length of the packet are used (or “cashed in”) from the token bucket. If the token arrival rate is higher than the rate of traffic so that there are tokens not being used, the token bucket is filled to capacity, and arriving tokens “overflow” the bucket and are lost. The token bucket depth represents the single user-configured burst size for the policer.

If there are tokens in the token bucket and the incoming traffic rate is higher than the token rate (the configured policer rate, bandwidth limit), the traffic can use the tokens until the bucket is empty. The token consumption rate can be as high as the incoming traffic rate, which creates the burst of traffic shown in Figure 64 on page 1000.

By using the token bucket algorithm, the average bandwidth rate being allowed is close to the configured bandwidth limit while simultaneously supporting bursty traffic, as shown in Figure 64 on page 1000.

Figure 64: Traffic Behavior Using Policer and Burst Size



NOTE: The measured length of a packet changes according to the family type that the policer applies to. If the policer is applied under the **family inet** hierarchy, the policer considers only the IPv4 packet length. If the policer is applied under the **family vpls** hierarchy, the entire Ethernet frame (including the Ethernet MAC header) is included in the packet length.

The major factor that affects the policer shaping result is not the conversion adjustment, but the traffic pattern since most network traffic is not consistent and is not sent at a constant rate. Due to the fluctuation of the incoming traffic rate, some of the allocated tokens are not used. As a result, the shaped traffic rate is lower than you might expect, and the TCP connection behavior discussed in “Understanding the Benefits of Policers and Token Bucket Algorithms” on page 1000 is a typical example of this. To alleviate this effect of the lower shaped traffic rate, a proper burst size configuration is required.

Related Documentation

- Understanding the Benefits of Policers and Token Bucket Algorithms on page 1000
- Determining Proper Burst Size for Traffic Policers on page 1002

Understanding the Benefits of Policers and Token Bucket Algorithms

This topic describes some scenarios that demonstrate how difficult it is to control traffic that comes into your network without the help of policers and the token bucket algorithm. These scenarios assume that traffic is coming from a TCP-based connection. Depending on the number of TCP connections, policers can have different affects on rate limits.

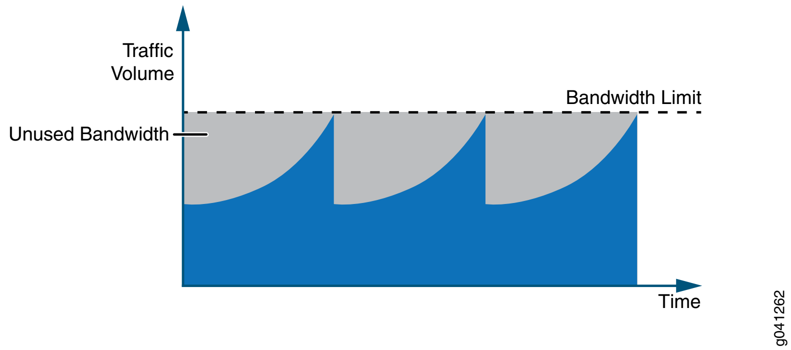
This topic presents the following scenarios:

- Scenario 1: Single TCP Connection on page 1000
- Scenario 2: Multiple TCP Connections on page 1001

Scenario 1: Single TCP Connection

Figure 65 on page 1001 shows the traffic loading on an interface with a policer configured. When the traffic rate reaches the configured bandwidth limit (which results in a packet drop), a TCP slow-start mechanism reduces the traffic rate down to half of what it was. When the traffic rate rises again, the same cycle repeats.

Figure 65: Policer Behavior with a Single TCP Connection

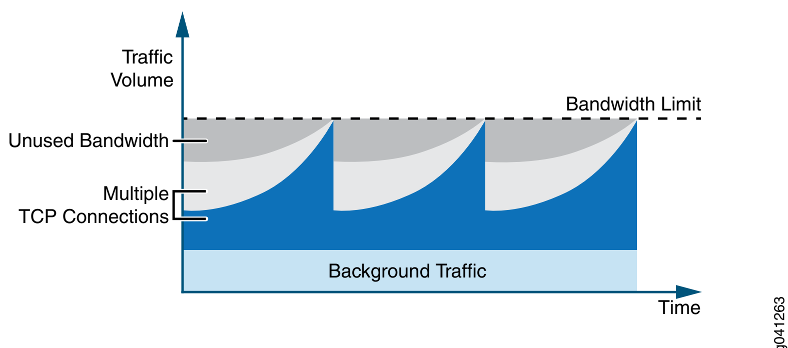


The problem presented in this scenario is that some bandwidth is available, but it is not being used by the traffic. The unused bandwidth shown in Figure 65 on page 1001 is the result of an overall data throughput that is lower than the configured bandwidth value. This example is an extreme case because there is only a single TCP connection.

Scenario 2: Multiple TCP Connections

With multiple TCP connections or some background non-TCP-based traffic, there is less unused bandwidth, as depicted in Figure 66 on page 1001. However, the same issue of unused bandwidth still exists if all the TCP connections experience a drop when the aggregated traffic rate exceeds the configured bandwidth limit.

Figure 66: Policer Behavior with Background Traffic (Multiple TCP Connections)



To reduce the problem of unused bandwidth in your network, you can configure a burst size.

Related Documentation

- Policer Implementation Overview on page 997
- Determining Proper Burst Size for Traffic Policers on page 1002

Determining Proper Burst Size for Traffic Policers

This topic covers the following information:

Policer Burst Size Limit Overview

A policer burst-size limit controls the number of bytes of traffic that can pass unrestricted through a policed interface when a burst of traffic pushes the average transmit or receive rate above the configured bandwidth limit. The actual number of bytes of bursty traffic allowed to pass through a policed interface can vary from zero to the configured burst-size limit, depending on the overall traffic load.

By configuring a proper burst size, the effect of a lower shaped rate is alleviated. Use the **burst-size-limit** statement to configure the burst size.

NOTE: **If you set the burst-size limit too low, too many packets will be subjected to rate limiting. If you set the burst-size limit too high, too few packets will be rate-limited.**

Consider these two main factors when determining the burst size to use:

- The allowed duration of a blast of traffic on the line.
- The burst size is large enough to handle the maximum transmission unit (MTU) size of the packets.

The following general guidelines apply to choosing a policer burst-size limit:

- A burst-size limit should not be set lower than 10 times the MTU of the traffic on the interface to be policed.
- The amount of time to allow a burst of traffic at the full line rate of a policed interface should not be lower than 5 ms.
- The minimum and maximum values you can specify for a policer burst-size limit depends on the policer type (two-color or three-color).

BEST PRACTICE: **The preferred method for choosing a burst-size limit is based on the line rate of the interface on which you apply the policer and the amount of time you want to allow a burst of traffic at the full line rate.**

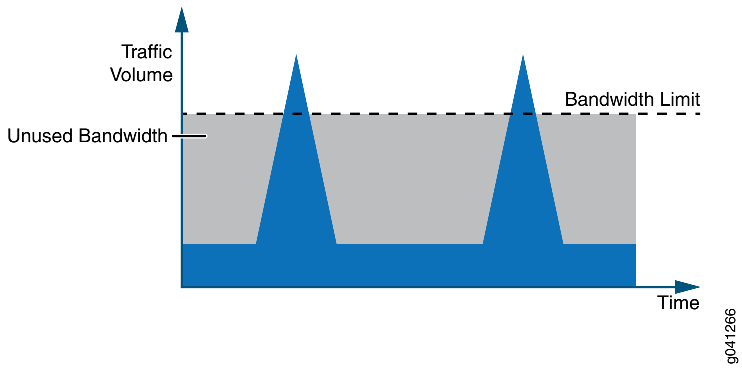
Effect of Burst-Size Limit

Bursty traffic requires a relatively large burst size so that extra tokens can be allocated into the token bucket for upcoming traffic to use.

Bursty Traffic Policed Without a Burst-Size Limit

Figure 67 on page 1003 shows an extreme case of bursty traffic where the opportunity to allocate tokens is missed, and the bandwidth goes unused because a large burst size is not configured.

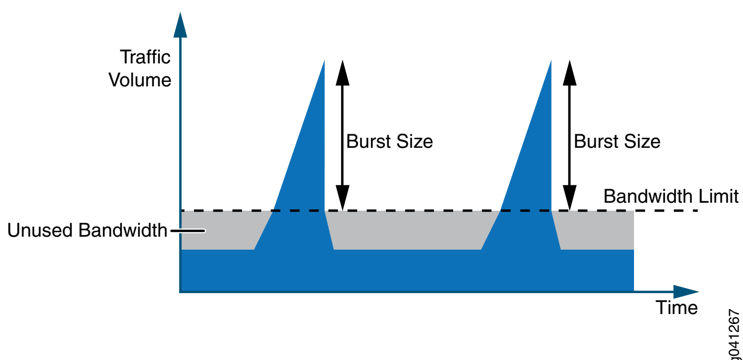
Figure 67: Bursty Traffic Without Configured Burst Size (Excessive Unused Bandwidth)



Burst-Size Limit Configured to Match Bandwidth Limit and Flow Burstiness

Figure 68 on page 1003 depicts how bandwidth usage changes when a large burst size is configured to handle bursty traffic. The large burst size minimizes the amount of unused bandwidth because tokens are being allocated in between the bursts of traffic that can be used during traffic peaks. The burst size determines the depth of the token bucket.

Figure 68: Bursty Traffic with Configured Burst Size (Less Unused Bandwidth)



Burst-Size Limit That Depletes All Accumulated Tokens

Configuring a large burst size for the unused tokens creates another issue. If the burst size is set to a very large value, the burst of traffic can be transmitted from the interface at line rate until all the accumulated tokens in the token bucket are used up. This means that configuring a large burst size can allow too many packets to avoid rate limiting, which can lead to a traffic rate that exceeds the bandwidth limit for an extended period of time.

If the average rate is considered within 1 second, the rate is still below the configured bandwidth limit. However, the downstream device might not be able to handle bursty traffic, so some packets might be dropped.

Two Methods for Calculating Burst-Size Limit

For policers configured on MX Series, M120, and M320 routers, and EX Series switches, configurable burst-size limit values range from 1 ms through 600 ms of traffic at the policer rate (the configured bandwidth limit).

Because one burst size is not suitable for every traffic pattern, select the best burst size for an interface by performing experimental configurations. For your first test configuration, select the burst-size limit by using one of the calculation methods described in the next two sections.

Calculation Based on Interface Bandwidth and Allowable Burst Time

If the bandwidth of the policed interface is known, the preferred method for calculating the policer burst-size limit is based on the following values:

- **bandwidth**—Line rate of the policed interface (in bps units)
- **burst-period**—Allowable traffic-burst time (5 ms or longer)

To calculate policer bandwidth in bytes:

```
bandwidth X burst-period / 8
```

Calculation Based on Interface Traffic MTU

If the bandwidth of the policed interface is unknown, calculate the policer burst-size limit based on the following value:

- **interface MTU**—Maximum transmission unit (in bytes) for the policed interface.

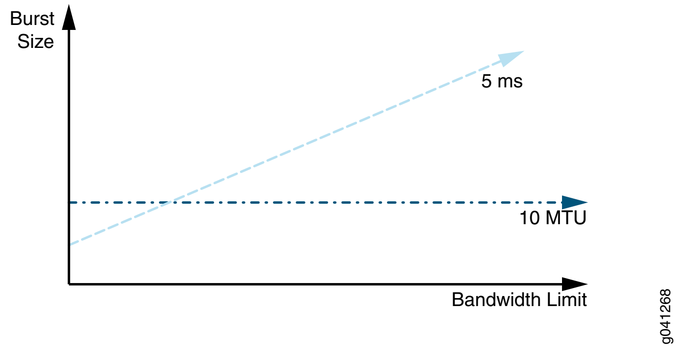
To calculate policer bandwidth in bytes:

```
interface MTU X 10
```

Comparison of the Two Methods

Figure 69 on page 1005 illustrates the relationship between the policer rate (the configured bandwidth limit) and the effective burst-size limit for the two methods of calculating the best policer burst-size limit. For the method based on interface bandwidth and allowable burst time, the correlation is labeled **5 ms**. For the method based on MTU size, the correlation is labeled **10 MTU**.

Figure 69: Comparing Burst Size Calculation Methods



For a policer burst-size limit calculated using the **5 ms** method, the effective burst-size limit is proportional to the configured bandwidth limit. With a very low bandwidth limit, the effective burst-size limit might be so small that the policer rate-limits traffic more aggressively than desired. For example, a traffic “burst” consisting of two MTU-sized packets might be rate-limited. In this scenario, a policer burst-size limit calculated using the **10 MTU** method appears to be a better choice.

10 x MTU Method for Selecting Initial Burst Size for Gigabit Ethernet with 100 Kbps Bandwidth

The following sequence illustrates the use of the 10 x MTU method for selecting an initial burst size for test configurations for a Gigabit Ethernet interface configured with a 100 Kbps bandwidth limit:

1. If you configure a 100 ms burst-size limit, the maximum amount of traffic allowed to pass through the interface unrestricted is 1250 bytes, calculated as follows:

```
                          100,000 bps x 0.1 s
100 Kbps x 100 ms   =   ---------------------   =  1250 bytes
                            8 bits per byte
```

2. In theory, a 10 x MTU burst size would allow up to 15,000 bytes to pass unrestricted. However, the maximum configurable burst-size limit for MX Series, M120, and M320 routers is 600 ms of the bandwidth limit. If you configure the maximum burst-size limit of 600 ms of the bandwidth limit, the maximum amount of traffic allowed to pass through the interface unrestricted is 7500 bytes, calculated as follows:

```
                          100,000 bps x 0.6 s
100 Kbps x 600 ms   =   ---------------------   =  7500 bytes
                            8 bits per byte
```

   On a Gigabit Ethernet interface, a configured burst-size limit of 600 ms creates a burst duration of 60 μs at Gigabit Ethernet line rate, calculated as follows:

```
 7500 bytes          60,000 bits
------------  =  -------------------  =  0.00006 s  =  60 μs
   1 Gbps         1,000,000,000 bps
```

3. If the downstream device is unable to handle the amount of bursty traffic allowed using the initial burst size configuration, reduce the burst-size limit until you achieve acceptable results.

5 ms Method for Selecting Initial Burst Size for Gigabit Ethernet Interface with 200 Mbps Bandwidth

The following sequence illustrates the use of the 5 ms method for selecting an initial burst size for test configurations for a Gigabit Ethernet interface configured with a 200 Mbps bandwidth limit. This example calculation shows how a larger burst-size limit can affect the measured bandwidth rate.

1. If you configure a 5 ms burst-size limit, the maximum amount of traffic allowed to pass through the interface unrestricted is 125,000 bytes (approximately 83 1500-byte packets), calculated as follows:

```
                     200,000,000 bps x 0.005 s
200 Mbps x 5 ms  =  --------------------------  =  125,000 bytes
                        8 bits per byte
```

   On a Gigabit Ethernet interface, a configured burst-size limit of 5 ms creates a burst duration of 1 ms at Gigabit Ethernet line rate, calculated as follows:

```
 125,000 bytes         1,000,000 bits
---------------  =  -------------------  =  0.001 s  =  10 ms
      1 Gbps          1,000,000,000 bps
```

   The average bandwidth rate in 1 second becomes 200 Mbps + 1 Mbps = 201 Mbps, which is a minimal increase over the configured bandwidth limit at 200 Mbps.

2. If you configure a 600 ms burst-size limit, the maximum amount of traffic allowed to pass through the interface unrestricted is 15 Mbytes (approximately 10,000 1500-byte packets), calculated as follows:

```
                       200,000,000 bps x 0.6 s
200 Mbps x 600 ms  =  -------------------------  =  15,000,000 bytes
                        8 bits per byte
```

   On a Gigabit Ethernet interface, a configured burst-size limit of 600 ms creates a burst duration of 120 ms at Gigabit Ethernet line rate, calculated as follows:

```
 15,000 bytes          120,000 bits
--------------  =  -------------------  =  0.012 s  =  120 ms
     1 Gbps          1,000,000,000 bps
```

   The average bandwidth rate in 1 second becomes 200 Mbps + 120 Mbps = 320 Mbps, which is much higher than the configured bandwidth limit at 200 Mbps.

200 Mbps Bandwidth Limit, 5 ms Burst Duration

If a 200 Mbps bandwidth limit is configured with a 5 ms burst size, the calculation becomes **200 Mbps x 5 ms = 125 Kbytes**, which is approximately 83 1500-byte packets. If the 200 Mbps bandwidth limit is configured on a Gigabit Ethernet interface, the burst duration is **125000 bytes / 1 Gbps = 1 ms** at the Gigabit Ethernet line rate.

200 Mbps Bandwidth Limit, 600 ms Burst Duration

If a large burst size is configured at 600 ms with the bandwidth limit configured at 200 Mbps, the calculation becomes **200 Mbps x 600 ms = 15 Mbytes**. This creates a burst duration of 120 ms at the Gigabit Ethernet line rate. The average bandwidth rate in 1 second becomes **200 Mbps + 15 Mbytes = 320 Mbps**, which is much higher than the configured bandwidth limit at 200 Mbps. This example shows that a larger burst size can affect the measured bandwidth rate.

Related Documentation

- Policer Implementation Overview on page 997
- Understanding the Benefits of Policers and Token Bucket Algorithms on page 1000

CHAPTER 29

Configuring Layer 2 Policers

Hierarchical Policers

Hierarchical Policer Overview

You can use a hierarchical policer to rate-limit ingress Layer 2 traffic at a physical or logical interface and apply different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority.

Hierarchical policing is supported on M40e, M120, and M320 edge routers with incoming Flexible PIC Concentrators (FPCs) as SFPC and outgoing FPCs as FFPC, and on MX Series, T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs.

You can apply hierarchical policing to a logical interface.

A hierarchical policer configuration defines two policers—one for EF traffic only and another for non-EF traffic—that function in a hierarchical manner:

- Premium policer—You configure the premium policer with traffic limits for high-priority EF traffic only: a guaranteed bandwidth and a corresponding burst-size limit. EF traffic is categorized as nonconforming when its average arrival rate exceeds the guaranteed bandwidth and its average packet size exceeds the premium burst-size limit. For a premium policer, the only configurable action for nonconforming traffic is to discard the packets.
- Aggregate policer—You configure the aggregate policer with an aggregate bandwidth (to accommodate both high-priority EF traffic up to the guaranteed bandwidth and normal-priority non-EF traffic) and a burst-size limit for non-EF traffic only. Non-EF traffic is categorized as nonconforming when its average arrival rate exceeds the amount of aggregate bandwidth not currently consumed by EF traffic and its average packet size exceeds the burst-size limit defined in the aggregate policer. For an aggregate policer, the configurable actions for nonconforming traffic are to discard the packets, assign a forwarding class, or assign a packet loss priority (PLP) level.

NOTE: You must configure the bandwidth limit of the premium policer at or below the bandwidth limit of the aggregate policer. If the two bandwidth limits are equal, then non-EF traffic passes through the interface unrestricted only while no EF traffic arrives at the interface.

EF traffic is guaranteed the bandwidth specified as the premium bandwidth limit, while non-EF traffic is rate-limited to the amount of aggregate bandwidth not currently consumed by the EF traffic. Non-EF traffic is rate-limited to the entire aggregate bandwidth only while no EF traffic is present.

For example, suppose that you configure a hierarchical policer with the following components:

- Premium policer with bandwidth limit set to 2 Mbps, burst-size limit set to 3000 bytes, and nonconforming action set to discard packets.
- Aggregate policer with bandwidth limit set to 10 Mbps, burst-size limit set to 3000 bytes, and nonconforming action set to discard packets.

EF traffic is guaranteed a bandwidth of 2 Mbps. Bursts of EF traffic—EF traffic that arrives at the interface at rates above 2 Mbps—can also pass through the interface provided sufficient tokens are available in the 3000-byte bucket. When no tokens are available for a burst of non-EF traffic, packets are rate-limited using policing actions for the premium policer.

Non-EF traffic is metered to a bandwidth limit that ranges between 8 Mbps and 10 Mbps, depending on the average arrival rate of the EF traffic. Bursts of non-EF traffic—non-EF traffic that arrives at the interface at rates above the current limit for non-EF traffic—also pass through the interface provided sufficient tokens are available in the 3000-byte bucket. When non-EF traffic exceeds the currently allowed bandwidth or when no tokens are available for a burst of non-EF traffic, packets are rate-limited using policing actions for the aggregate policer.

Example: Configuring a Hierarchical Policer

This example shows how to configure a hierarchical policer and apply the policer to ingress Layer 2 traffic at a logical interface on a supported platform.

- Requirements on page 1011
- Overview on page 1011
- Configuration on page 1011
- Verification on page 1015

Requirements

Before you begin, be sure that your environment meets the following requirements:

- The interface on which you apply the hierarchical policer is a SONET interface hosted on one of the following routing platforms:
  - M40e, M120, or M320 edge router with incoming FPCs as SFPC and outgoing FPCs as FFPC.
  - MX Series, T320, T640, or T1600 core router with Enhanced Intelligent Queuing (IQE) PICs.
- No other policer is applied to the input of the interface on which you apply the hierarchical policer.
- You are aware that, if you apply the hierarchical policer to logical interface on which an input filter is also applied, the policer is executed first.

Overview

In this example, you configure a hierarchical policer and apply the policer to ingress Layer 2 traffic at a logical interface.

Topology

You apply the policer to the SONET logical interface **so-1/0/0.0**, which you configure for IPv4 and VPLS traffic. When you apply the hierarchical policer to that logical interface, both IPv4 and VPLS traffic is hierarchically rate-limited.

You also configure the logical interface **so-1/0/0.1** for MPLS traffic. If you choose to apply the hierarchical policer to physical interface **so-1/0/0**, hierarchical policing would apply to IPv4 and VPLS traffic at **so-1/0/0.0** and to MPLS traffic at **so-1/0/0.1**.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Defining the Interfaces on page 1012
- Defining the Forwarding Classes on page 1013
- Configuring the Hierarchical Policer on page 1013
- Applying the Hierarchical Policer to Layer 2 Ingress Traffic at a Physical or Logical Interface on page 1014

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces so-1/0/0 unit 0 family inet address 192.168.1.1/24
```

```
set interfaces so-1/0/0 unit 0 family vpls
set interfaces so-1/0/0 unit 1 family mpls
set class-of-service forwarding-classes class fc0 queue-num 0 priority high
  policing-priority premium
set class-of-service forwarding-classes class fc1 queue-num 1 priority low policing-priority
  normal
set class-of-service forwarding-classes class fc2 queue-num 2 priority low policing-priority
  normal
set class-of-service forwarding-classes class fc3 queue-num 3 priority low policing-priority
  normal
set firewall hierarchical-policer policer1 aggregate if-exceeding bandwidth-limit 300m
  burst-size-limit 30k
set firewall hierarchical-policer policer1 aggregate then forwarding-class fc1
set firewall hierarchical-policer policer1 premium if-exceeding bandwidth-limit 100m
  burst-size-limit 50k
set firewall hierarchical-policer policer1 premium then discard
set interfaces so-1/0/0 unit 0 layer2-policer input-hierarchical-policer policer1
```

Defining the Interfaces

Step-by-Step Procedure

To define the interfaces:

1. Enable configuration of the physical interface.

   ```
   [edit]
   user@host# edit interfaces so-1/0/0
   ```

2. Configure logical unit 0.

   ```
   [edit interfaces so-1/0/0]
   user@host# set unit 0 family inet address 192.168.1.1/24
   user@host# set unit 0 family vpls
   ```

   If you apply a Layer 2 policer to this logical interface, you must configure at least one protocol family.

3. Configure logical unit 1.

   ```
   [edit interfaces so-1/0/0]
   user@host# set unit 1 family mpls
   ```

Results

Confirm the configuration of the interfaces by entering the **show interfaces** configuration command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
so-1/0/0 {
  unit 0 {
    family inet {
      address 192.168.1.1/24;
    }
    family vpls;
```

```
    }
    unit 1 {
      family mpls;
    }
  }
```

Defining the Forwarding Classes

Step-by-Step Procedure

To define the forwarding classes referenced as aggregate policer actions:

1. Enable configuration of the forwarding classes.

   ```
   [edit]
   user@host# edit class-of-service forwarding-classes
   ```

2. Define the forwarding classes.

   ```
   [edit class-of-service forwarding-classes]
   user@host# set class fc0 queue-num 0 priority high policing-priority premium
   user@host# set class fc1 queue-num 1 priority low policing-priority normal
   user@host# set class fc2 queue-num 2 priority low policing-priority normal
   user@host# set class fc3 queue-num 3 priority low policing-priority normal
   ```

Results

Confirm the configuration of the forwarding classes referenced as aggregate policer actions by entering the **show class-of-service** configuration command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show class-of-service
forwarding-classes {
  class fc0 queue-num 0 priority high policing-priority premium;
  class fc1 queue-num 1 priority low policing-priority normal;
  class fc2 queue-num 2 priority low policing-priority normal;
  class fc3 queue-num 3 priority low policing-priority normal;
}
```

Configuring the Hierarchical Policer

Step-by-Step Procedure

To configure a hierarchical policer:

1. Enable configuration of the hierarchical policer.

   ```
   [edit]
   user@host# edit firewall hierarchical-policer policer1
   ```

2. Configure the aggregate policer.

   ```
   [edit firewall hierarchical-policer policer1]
   user@host# set aggregate if-exceeding bandwidth-limit 300m burst-size-limit 30k
   user@host# set aggregate then forwarding-class fc1
   ```

For the aggregate policer, the configurable actions for a packet in a nonconforming flow are to discard the packet, change the loss priority, or change the forwarding class.

3. Configure the premium policer.

```
[edit firewall hierarchical-policer policer1]
user@host# set premium if-exceeding bandwidth-limit 100m burst-size-limit 50k
user@host# set premium then discard
```

The bandwidth limit for the premium policer must not be greater than that of the aggregate policer.

For the premium policer, the only configurable action for a packet in a nonconforming traffic flow is to discard the packet.

Results Confirm the configuration of the hierarchical policer by entering the **show firewall** configuration command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
hierarchical-policer policer1 {
  aggregate {
    if-exceeding {
      bandwidth-limit 300m;
      burst-size-limit 30k;
    }
    then {
      forwarding-class fc1;
    }
  }
  premium {
    if-exceeding {
      bandwidth-limit 100m;
      burst-size-limit 50k;
    }
    then {
      discard;
    }
  }
}
```

Applying the Hierarchical Policer to Layer 2 Ingress Traffic at a Physical or Logical Interface

Step-by-Step Procedure To hierarchically rate-limit Layer 2 ingress traffic for IPv4 and VPLS traffic only on logical interface **so-1/0/0.0**, reference the policer from the logical interface configuration:

1. Enable configuration of the logical interface.

```
[edit]
user@host# edit interfaces so-1/0/0 unit 0
```


When you apply a policer to Layer 2 traffic at a logical interface, you must define at least one protocol family for the logical interface.

2. Apply the policer to the logical interface.

```
[edit]
user@host# set layer2-policer input-hierarchical-policer policer1
```

Alternatively, to hierarchically rate-limit Layer 2 ingress traffic for all protocol families and for *all logical interfaces* configured on physical interface **so-1/0/0**, you could reference the policer from the physical interface configuration.

Results Confirm the configuration of the hierarchical policer by entering the **show interfaces** configuration command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
so-1/0/0 {
  unit 0 {
    layer2-policer {
      input-hierarchical-policer policer1;
    }
    family inet {
      address 192.168.1.1/24;
    }
    family vpls;
  }
  unit 1 {
    family mpls;
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying Traffic Statistics and Policers for the Logical Interface on page 1015
- Displaying Statistics for the Policer on page 1016

Displaying Traffic Statistics and Policers for the Logical Interface

Purpose Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

Action Use the **show interfaces** operational mode command for logical interface **so-1/0/0.0**, and include the **detail** or **extensive** option. The command output section for **Traffic statistics** lists the number of bytes and packets received and transmitted on the logical interface, and the **Protocol inet** section contains a **Policer** field that would list the policer **policer1** as an input or output policer as follows:

- **Input: policer1-so-1/0/0.0-inet-i**
- **Output: policer1-so-1/0/0.0-inet-o**

In this example, the policer is applied to logical interface traffic in the input direction only.

Displaying Statistics for the Policer

Purpose Verify the number of packets evaluated by the policer.

Action Use the **show policer** operational mode command and optionally specify the name of the policer. The command output displays the number of packets evaluated by each configured policer (or the specified policer), in each direction. For the policer **policer1**, the input and output policer names are displayed as follows:

- **policer1-so-1/0/0.0-inet-i**
- **policer1-so-1/0/0.0-inet-o**

The **-inet-i** suffix denotes a policer applied to IPv4 input traffic, while the **-inet-o** suffix denotes a policer applied to IPv4 output traffic. In this example, the policer is applied to input traffic only.

Related Documentation

- Hierarchical Policer Configuration Overview on page 981
- Guidelines for Applying Traffic Policers on page 983

Two-Color and Three-Color Policers at Layer 2

Two-Color Policing at Layer 2 Overview

This topic covers the following information:

Guidelines for Configuring Two-Color Policing of Layer 2 Traffic

The following guidelines apply to two-color policing of Layer 2 traffic:

- You can apply a two-color policer to ingress or egress Layer 2 traffic at a logical interface hosted on a Gigabit Ethernet interface (**ge-**) or a 10-Gigabit Ethernet interface (**xe-**) only.
- A single logical interface supports Layer 2 policing in both directions.
- You can apply a two-color policer to Layer 2 traffic as a logical interface policer only. You cannot apply a two-color policer to Layer 2 traffic as a stateless firewall filter action.
- You can apply a two-color policer to Layer 2 traffic by referencing the policer in the interface configuration at the logical unit level, and not at the protocol level.

For information about configuring three-color policing of Layer 2 traffic, see "Three-Color Policing at Layer 2 Overview" on page 1018.

Statement Hierarchy for Configuring a Two-Color Policer for Layer 2 Traffic

To enable a single-rate two-color policer to rate-limit Layer 2 traffic, include the **logical-interface-policer** statement in the **policer** configuration.

```
firewall {
  policer policer-name {
    logical-interface-policer;
    if-exceeding {
      (bandwidth-limit bps | bandwidth-percent percentage);
      burst-size-limit bytes;
    }
    then {
      discard;
      forwarding-class class-name;
      loss-priority (high | low | medium-high | medium-low);
    }
  }
}
```

You can include the configuration at the following hierarchy levels:

- **[edit]**
- **[edit logical-systems *logical-system-name*]**

Statement Hierarchy for Applying a Two-Color Policer to Layer 2 Traffic

To apply a logical interface policer to Layer 2 traffic, include the **layer2-policer input-policer *policer-name*** statement or the **layer2-policer output-policer *policer-name*** statement to a supported logical interface. Use the **input-policer** or **output-policer** statements to apply a two-color policer at Layer 2.

```
interfaces {
  (ge-fpc/pic/port | xe-fpc/pic/port) {
    unit unit-number {
```



```
            layer2-policer {
               input-policer policer-name;
               output-policer policer-name;
            }
         }
      }
   }
```

You can include the configuration at the following hierarchy levels:

- **[edit]**
- **[edit logical-systems *logical-system-name*]**

Three-Color Policing at Layer 2 Overview

This topic covers the following information:

Guidelines for Configuring Three-Color Policing of Layer 2 Traffic

The following guidelines apply to three-color policing of Layer 2 traffic:

- You can apply a three-color policer to Layer 2 traffic at a logical interface hosted on a Gigabit Ethernet interface (**ge-**) or a 10-Gigabit Ethernet interface (**xe-**) only.
- A single logical interface supports Layer 2 policing in both directions.
- You can apply a three-color policer to Layer 2 traffic as a logical interface policer only. You cannot apply a two-color policer to Layer 2 traffic as a stateless firewall filter action.
- You can apply a three-color policer to Layer 2 traffic by referencing the policer in the interface configuration at the logical unit level, and not at the protocol level.
- You can apply a color-aware three-color policer to Layer 2 traffic in the egress direction only, but you apply a color-blind three-color policer to Layer 2 traffic in either direction.

For information about configuring two-color policing of Layer 2 traffic, see "Two-Color Policing at Layer 2 Overview" on page 1016.

Statement Hierarchy for Configuring a Three-Color Policer for Layer 2 Traffic

To enable a single-rate or two-rate three-color policer to rate-limit Layer 2 traffic, include the **logical-interface-policer** statement in the **three-color-policer** configuration.

```
firewall {
   three-color-policer policer-name {
      action {
         loss-priority high then discard;
      }
      logical-interface-policer;
      single-rate {
```



```
            (color-aware | color-blind);
            committed-burst-size bytes;
            committed-information-rate bps;
            excess-burst-size bytes;
        }
        two-rate {
            (color-aware | color-blind);
            committed-burst-size bytes;
            committed-information-rate bps;
            peak-burst-size bytes;
            peak-information-rate bps;
        }
    }
}
```

You can include the configuration at the following hierarchy levels:

- **[edit]**
- **[edit logical-systems *logical-system-name*]**

Statement Hierarchy for Applying a Three-Color Policer to Layer 2 Traffic

To apply a logical interface policer to Layer 2 traffic, include the **layer2-policer** statement for a supported logical interface at the logical unit level. Use the **input-three-color** ***policer-name*** statement or **output-three-color** ***policer-name*** statement to specify the direction of the traffic to be policed.

```
interfaces {
    (ge-fpc/pic/port | xe-fpc/pic/port) {
        unit unit-number {
            layer2-policer {
                input-three-color policer-name;
                output-three-color policer-name;
            }
        }
    }
}
```

You can include the configuration at the following hierarchy levels:

- **[edit]**
- **[edit logical-systems *logical-system-name*]**

Example: Configuring a Three-Color Logical Interface (Aggregate) Policer

This example shows how to configure a two-rate three-color color-blind policer as a logical interface (aggregate) policer and apply the policer directly to Layer 2 input traffic at a supported logical interface.

Requirements

Before you begin, make sure that the logical interface to which you apply the three-color logical interface policer is hosted on a Gigabit Ethernet interface (**ge-**) or a 10-Gigabit Ethernet interface (**xe-**) on an MX Series router.

Overview

A two-rate three-color policer meters a traffic flow against a bandwidth limit and burst-size limit for guaranteed traffic, plus a second set of bandwidth and burst-size limits for peak traffic. Traffic that conforms to the limits for guaranteed traffic is categorized as green, and nonconforming traffic falls into one of two categories:

- Nonconforming traffic that does not exceed the bandwidth and burst-size limits for peak traffic is categorized as yellow.
- Nonconforming traffic that exceeds the bandwidth and burst-size limits for peak traffic is categorized as red.

A logical interface policer defines traffic rate-limiting rules that you can apply to multiple protocol families on the same logical interface without creating multiple instances of the policer.

NOTE: **You apply a logical interface policer directly to a logical interface at the logical unit level, and not by referencing the policer in a stateless firewall filter and then applying the filter to the logical interface at the protocol family level.**

Topology

In this example, you configure the two-rate three-color policer **trTCM2-cb** as a color-blind logical interface policer and apply the policer to incoming Layer 2 traffic on logical interface **ge-1/3/1.0**.

NOTE: **When using a three-color policer to rate-limit Layer 2 traffic, color-aware policing can be applied to egress traffic only.**

The policer defines guaranteed traffic rate limits such that traffic that conforms to the bandwidth limit of 40 Mbps with a 100 KB allowance for traffic bursting (based on the token-bucket formula) is categorized as green. As with any policed traffic, the packets in a green flow are implicitly set to a **low** loss priority and then transmitted.

Nonconforming traffic that falls within the peak traffic limits of a 60 Mbps bandwidth limit and a 200 KB allowance for traffic bursting (based on the token-bucket formula) is categorized as yellow. The packets in a yellow traffic flow are implicitly set to a **medium-high** loss priority and then transmitted.

Nonconforming traffic that exceeds the peak traffic limits are categorized as red. The packets in a red traffic flow are implicitly set to a **high** loss priority. In this example, the optional policer action for red traffic (**loss-priority high then discard**) is configured, so packets in a red traffic flow are discarded instead of transmitted.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring the Logical Interfaces on page 1021
- Configuring the Two-Rate Three-Color Policer as a Logical Interface Policer on page 1022
- Applying the Three-Color Policer to the Layer 2 Input at the Logical Interface on page 1024

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces ge-1/3/1 vlan-tagging
set interfaces ge-1/3/1 unit 0 vlan-id 100
set interfaces ge-1/3/1 unit 0 family inet address 10.10.10.1/30
set interfaces ge-1/3/1 unit 1 vlan-id 101
set interfaces ge-1/3/1 unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
   00:00:11:22:33:44
set firewall three-color-policer trTCM2-cb logical-interface-policer
set firewall three-color-policer trTCM2-cb two-rate color-blind
set firewall three-color-policer trTCM2-cb two-rate committed-information-rate 40m
set firewall three-color-policer trTCM2-cb two-rate committed-burst-size 100k
set firewall three-color-policer trTCM2-cb two-rate peak-information-rate 60m
set firewall three-color-policer trTCM2-cb two-rate peak-burst-size 200k
set firewall three-color-policer trTCM2-cb action loss-priority high then discard
set interfaces ge-1/3/1 unit 0 layer2-policer input-three-color trTCM2-cb
```

Configuring the Logical Interfaces

Step-by-Step Procedure

To configure the logical interfaces:

1. Enable configuration of the interface.

   ```
   [edit]
   user@host# edit interfaces ge-1/3/1
   ```

2. Configure single tagging.

   ```
   [edit interfaces ge-1/3/1]
   user@host# set vlan-tagging
   ```

3. Configure logical interface **ge-1/3/1.0**.

   ```
   [edit interfaces ge-1/3/1]
   user@host# set unit 0 vlan-id 100
   user@host# set unit 0 family inet address 10.10.10.1/30
   ```

4. Configure logical interface **ge-1/3/1.0**.

```
[edit interfaces ge-1/3/1]
user@host# set unit 1 vlan-id 101
user@host# set unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
   00:00:11:22:33:44
```

Results Confirm the configuration of the logical interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/1 {
   vlan-tagging;
   unit 0 {
      vlan-id 100;
      family inet {
         address 10.10.10.1/30;
      }
   }
   unit 1 {
      vlan-id 101;
      family inet {
         address 20.20.20.1/30 {
            arp 20.20.20.2 mac 00:00:11:22:33:44;
         }
      }
   }
}
```

Configuring the Two-Rate Three-Color Policer as a Logical Interface Policer

Step-by-Step Procedure To configure the two-rate three-color policer as a logical interface policer:

1. Enable configuration of a three-color policer.

```
[edit]
user@host# edit firewall three-color-policer trTCM2-cb
```

2. Specify that the policer is a logical interface (aggregate) policer.

```
[edit firewall three-color-policer trTCM2-cb]
user@host# set logical-interface-policer
```

   A logical interface policer rate-limits traffic based on a percentage of the media rate of the physical interface underlying the logical interface to which the policer is applied, and the policer is applied directly to the interface rather than referenced by a firewall filter.

3. Specify that the policer is two-rate and color-blind.

```
[edit firewall three-color-policer trTCM2-cb]
```



```
user@host# set two-rate color-blind
```

A color-aware three-color policer takes into account any coloring markings that might have been set for a packet by another traffic policer configured at a previous network node, and any preexisting color markings are used in determining the appropriate policing action for the packet.

Because you are applying this three-color policer applied to input at Layer 2, you must configure the policer to be color-blind.

4. Specify the policer traffic limits used to classify a green traffic flow.

```
[edit firewall three-color-policer trTCM2-cb]
user@host# set two-rate committed-information-rate 40m
user@host# set two-rate committed-burst-size 100k
```

5. Specify the additional policer traffic limits used to classify a yellow or red traffic flow.

```
[edit firewall three-color-policer trTCM2-cb]
user@host# set two-rate peak-information-rate 60m
user@host# set two-rate peak-burst-size 200k
```

6. (Optional) Specify the configured policer action for packets in a red traffic flow.

```
[edit firewall three-color-policer trTCM2-cb]
user@host# set action loss-priority high then discard
```

In color-aware mode, the three-color policer configured action can increase the packet loss priority (PLP) level of a packet, but never decrease it. For example, if a color-aware three-color policer meters a packet with a medium PLP marking, it can raise the PLP level to high, but cannot reduce the PLP level to low.

Results Confirm the configuration of the three-color policer by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
three-color-policer trTCM2-cb {
  logical-interface-policer;
  action {
    loss-priority high then discard;
  }
  two-rate {
    color-blind;
    committed-information-rate 40m;
    committed-burst-size 100k;
    peak-information-rate 60m;
    peak-burst-size 200k;
  }
}
```


Applying the Three-Color Policer to the Layer 2 Input at the Logical Interface

Step-by-Step Procedure

To apply the three-color policer to the Layer 2 input at the logical interface:

1. Enable application of Layer 2 logical interface policers.

```
[edit]
user@host# edit interfaces ge-1/3/1 unit 0
```

2. Apply the three-color logical interface policer to a logical interface input.

```
[edit interfaces ge-1/3/1 unit 0]
user@host# set layer2-policerinput-three-color trTCM2-cb
```

Results

Confirm the configuration of the logical interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/1 {
  vlan-tagging;
  unit 0 {
    vlan-id 100;
    layer2-policer {
      input-three-color trTCM2-cb;
    }
    family inet {
      address 10.10.10.1/30;
    }
  }
  unit 1 {
    vlan-id 101;
    family inet {
      address 20.20.20.1/30 {
        arp 20.20.20.2 mac 00:00:11:22:33:44;
      }
    }
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying Traffic Statistics and Policers for the Logical Interface on page 1025
- Displaying Statistics for the Policer on page 1025

Displaying Traffic Statistics and Policers for the Logical Interface

Purpose Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

Action Use the **show interfaces** operational mode command for logical interface **ge-1/3/1.0**, and include the **detail** or **extensive** option. The command output section for **Traffic statistics** lists the number of bytes and packets received and transmitted on the logical interface, and the **Protocol inet** section contains a **Policer** field that would list the policer **trTCM2-cb** as an input or output policer as follows:

- **Input: trTCM2-cb-ge-1/3/1.0-log_int-i**
- **Output: trTCM2-cb-ge-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to in the input direction only.

Displaying Statistics for the Policer

Purpose Verify the number of packets evaluated by the policer.

Action Use the show policer operational mode command and optionally specify the name of the policer. The command output displays the number of packets evaluated by each configured policer (or the specified policer), in each direction. For the policer **trTCM2-cb**, the input and output policer names are displayed as follows:

- **trTCM2-cb-ge-1/3/1.0-log_int-i**
- **trTCM2-cb-e-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to input traffic only.

Related Documentation

- Guidelines for Applying Traffic Policers on page 983
- layer2-policer on page 1305
- logical-interface-policer on page 1309
- policer (Configuring) on page 1322
- three-color-policer (Configuring) on page 1333

CHAPTER 30

Configuring Two-Color Traffic Policers at Layer 3

Two-Color Policer Configuration Overview

Table 68 on page 1027 describes the hierarchy levels at which you can configure and apply single-rate two-color policers to Layer 3 traffic. For information about applying single-rate two-color policers to Layer 2 traffic, see "Two-Color Policing at Layer 2 Overview" on page 1016.

Table 68: Two-Color Policer Configuration and Application Overview

Policer Configuration	Layer 3 Application	Key Points
Single-Rate Two-Color Policer *Defines traffic rate limiting that you can apply to Layer 3 protocol-specific traffic at a logical interface. Can be applied as an interface policer or as a firewall filter policer.*		

Basic policer configuration:
 	
[edit firewall]
policer *policer-name* {
if-exceeding {
bandwidth-limit *bps*;
burst-size-limit *bytes*;
}
then {
discard;
forwarding-class *class-name*;
loss-priority *supported-value*;
}
	Method A—Apply as an interface policer at the protocol family level:
	
	[edit interfaces]
	interface-name {
	unit *unit-number* {
	family *family-name* {
	policer {
	input *policer-name*;
	output *policer-name*;
	}
	}
	}
		Policer configuration:
		• Use **bandwidth-limit** ***bps*** to specify an absolute value.
		Firewall filter configuration (*)
		• If applying to multiple interfaces, include the **interface-specific** statement to create unique policers and counters for each interface.

Table 68: Two-Color Policer Configuration and Application Overview *(continued)*

Policer Configuration	Layer 3 Application	Key Points

}	}
	
	Method B—Apply as a firewall filter policer at the protocol family level:
	
	[edit firewall]
	family *family-name* {
	  filter *filter-name* {
	    interface-specific; # (*)
	    from {
	      ... *match-conditions* ...
	    }
	    then {
	      policer *policer-name*;
	    }
	  }
	}
	
	[edit interfaces]
	interface-name {
	  unit *unit-number* {
	    family *family-name* {
	      filter {
	        input *filter-name*;
	        output *filter-name*;
	      }
	      ... *protocol-configuration* ...
	    }
	  }
	}
		Interface policer verification:
		
		• Use the **show interfaces (detail | extensive)** operational mode command.
		• Use the **show policer** operational mode command.
		
		Firewall filter policer verification:
		
		• Use the **show interfaces (detail | extensive)** operational mode command.
		• Use the **show firewall filter *filter-name*** operational mode command.

Table 68: Two-Color Policer Configuration and Application Overview *(continued)*

Policer Configuration	Layer 3 Application	Key Points
Bandwidth Policer *Defines traffic rate limiting that you can apply to Layer 3 protocol-specific traffic at a logical interface, but the bandwidth limit is specified as a percentage value. Bandwidth can be based on physical interface line rate (the default) or the logical interface shaping rate. Can be applied as an interface policer or as a firewall filter policer where the filter is either interface-specific or a physical interface filter.*		

Bandwidth policer configuration:
 	
[edit firewall]
policer *policer-name* {
  logical-bandwidth-policer;
  if-exceeding {
    bandwidth-percent (1..100);
    burst-size-limit *bytes*;
  }
  then {
    discard;
    forwarding-class *class-name*;
    loss-priority *supported-value*;
  }
}
	Method A—Apply as an interface policer at the protocol family level:
	
	[edit interfaces]
	interface-name {
	  unit *unit-number* {
	    family *family-name* {
	      policer {
	        input *policer-name*;
	        output *policer-name*;
	      }
	    }
	  }
	}
	
	Method B—Apply as a firewall filter policer at the protocol family level:
	
	[edit firewall]
	family *family-name* {
	  filter *filter-name* {
	    interface-specific;
	    from {
	      ... *match-conditions* ...
	    }
	    then {
	      policer *policer-name*;
	    }
	  }
	}
	
	[edit interfaces]
	interface-name {
	  unit *unit-number* {
	    family *family-name* {
	      filter {
	        input *filter-name*;
	        output *filter-name*;
	      }
	      ... *protocol-configuration* ...
	    }
	  }
	}
		Policer configuration:
		
		• Use the **bandwidth-percent *percentage*** statement instead of the **bandwidth-limit *bps*** statement.
		By default, bandwidth policing rate-limits traffic based on a percentage of the physical interface media rate.
		• To rate-limit traffic based on a percentage of the logical interface configured shaping rate, also include the **logical-bandwidth-policer** statement.
		
		Firewall filter configuration:
		
		• Percentage bandwidth policers can only be referenced by filters configured with the **interface-specific** statement.
		
		Interface policer verification:
		
		• Use the **show interfaces (detail | extensive)** operational mode command.
		• Use the **show policer** operational mode command.
		
		Firewall filter policer verification:
		
		• Use the **show interfaces (detail | extensive)** operational mode command.
		• Use the **show firewall filter *filter-name*** operational mode command.

Table 68: Two-Color Policer Configuration and Application Overview *(continued)*

Policer Configuration	Layer 3 Application	Key Points
Logical Interface (Aggregate) Policer *Defines traffic rate limiting that you can apply to multiple protocol families on the same logical interface without creating multiple instances of the policer. Can be applied directly to a logical interface configuration only.*		

Logical interface policer configuration:
 	
[edit firewall]
policer *policer-name* {
logical-interface-policer;
if-exceeding {
bandwidth-limit *bps*;
burst-size-limit *bytes*;
}
then {
discard;
forwarding-class *class-name*;
loss-priority *supported-value*;
}
}
	Apply as an interface policer only:
	
	[edit interfaces]
	interface-name {
	unit *unit-number* {
	policer { # All protocols
	input *policer-name*;
	output *policer-name*;
	}
	family *family-name* {
	policer { # One protocol
	input *policer-name*;
	output *policer-name*;
	}
	}
	}
	}
		Policer configuration:
		
		• Include the **logical-interface-policer** statement.
		
		Two options for interface policer application:
		
		• To rate-limit all traffic types, regardless of the protocol family, apply the logical interface policer at the logical unit level.
		• To rate-limit traffic of a specific protocol family, apply the logical interface policer at the protocol family level.
		
		Interface policer verification:
		
		• Use the **show interfaces (detail | extensive)** operational mode command.
		• Use the **show policer** operational mode command.

Table 68: Two-Color Policer Configuration and Application Overview *(continued)*

<table>
<tr><th>Policer Configuration</th><th>Layer 3 Application</th><th>Key Points</th></tr>
<tr><td colspan="3">Physical Interface Policer
Defines traffic rate limiting that applies to all logical interfaces and protocol families configured on a physical interface, even if the interfaces belong to different routing instances. Can be applied as a firewall filter policer referenced from a physical interface filter only.</td></tr>
<tr><td>Physical interface policer configuration:<pre>
[edit firewall]
policer policer-name {
  physical-interface-policer;
  if-exceeding {
    bandwidth-limit bps;
    burst-size-limit bytes;
  }
  then {
    discard;
    forwarding-class class-name;
    loss-priority supported-value;
  }
}
</pre></td><td>Apply as a firewall filter policer referenced from a physical interface filter that you apply at the protocol family level:<pre>
[edit firewall]
family family-name {
  filter filter-name {
    physical-interface-filter;
    from {
      ... match-conditions ...
    }
    then {
      policer policer-name;
    }
  }
}

[edit interfaces]
interface-name {
  unit number {
    family family-name {
      filter {
        input filter-name;
        output filter-name;
      }
      ... protocol-configuration ...
    }
  }
}
</pre></td><td>Policer configuration: Include the physical-interface-policer statement. Firewall filter configuration: Include the physical-interface-filter statement. Application: Apply the filter to the input or output of a logical interface at the protocol family level. Firewall filter policer verification: Use the show interfaces (detail | extensive) operational mode command. Use the show firewall filter filter-name operational mode command.</td></tr>
</table>

Related Documentation

- Basic Single-Rate Two-Color Policers on page 1032
- Bandwidth Policers on page 1051
- Filter-Specific Counters and Policers on page 1060
- Prefix-Specific Counting and Policing Actions on page 1072
- Multifield Classification on page 1088
- Policer Overhead to Account for Rate Shaping in the Traffic Manager on page 1106
- Two-Color and Three-Color Physical Interface Policers on page 1154

Basic Single-Rate Two-Color Policers

Single-Rate Two-Color Policer Overview

Single-rate two color policing enforces a configured rate of traffic flow for a particular service level by applying implicit or configured actions to traffic that does not conform to the limits. When you apply a single-rate two-color policer to the input or output traffic at an interface, the policer meters the traffic flow to the rate limit defined by the following components:

- Bandwidth limit—The average number of bits per second permitted for packets received or transmitted at the interface. You can specify the bandwidth limit as an absolute number of bits per second or as a percentage value from 1 through 100. If a percentage value is specified, the effective bandwidth limit is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.
- Packets per second (pps) limit (MX Series with MPC only)–The average number of packets per second permitted for packets received or transmitted at the interface. You specify the pps limit as an absolute number of packets per second.
- Burst-size limit—The maximum size permitted for bursts of data.
- Packet burst limit–

For a traffic flow that conforms to the configured limits (categorized as green traffic), packets are implicitly marked with a packet loss priority (PLP) level of **low** and are allowed to pass through the interface unrestricted.

For a traffic flow that exceeds the configured limits (categorized as red traffic), packets are handled according to the traffic-policing actions configured for the policer. The action might be to discard the packet, or the action might be to re-mark the packet with a specified forwarding class, a specified PLP, or both, and then transmit the packet.

To rate-limit Layer 3 traffic, you can apply a two-color policer in the following ways:

- Directly to a logical interface, at a specific protocol level.
- As the action of a standard stateless firewall filter that is applied to a logical interface, at a specific protocol level.

To rate-limit Layer 2 traffic, you can apply a two-color policer as a *logical interface policer* only. You cannot apply a two-color policer to Layer 2 traffic through a firewall filter.

Example: Limiting Inbound Traffic at Your Network Border by Configuring an Ingress Single-Rate Two-Color Policer

This example shows you how to configure an ingress single-rate two-color policer to filter incoming traffic. The policer enforces the class-of-service (CoS) strategy for in-contract and out-of-contract traffic. You can apply a single-rate two-color policer to incoming packets, outgoing packets, or both. This example applies the policer as an input (ingress) policer. The goal of this topic is to provide you with an introduction to policing by using a example that shows traffic policing in action.

Policers use a concept known as a token bucket to allocate system resources based on the parameters defined for the policer. A thorough explanation of the token bucket concept and its underlying algorithms is beyond the scope of this document. For more information about traffic policing, and CoS in general, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at www.juniper.net/books.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos OS. The example shown here was tested and verified on MX Series routers running Junos OS Release 10.4.

Overview

Single-rate two-color policing enforces a configured rate of traffic flow for a particular service level by applying implicit or configured actions to traffic that does not conform to the limits. When you apply a single-rate two-color policer to the input or output traffic at an interface, the policer meters the traffic flow to the rate limit defined by the following components:

- Bandwidth limit—The average number of bits per second permitted for packets received or transmitted at the interface. You can specify the bandwidth limit as an absolute number of bits per second or as a percentage value from 1 through 100. If a percentage value is specified, the effective bandwidth limit is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.
- Burst-size limit—The maximum size permitted for bursts of data. Burst sizes are measured in bytes. We recommend two formulas for calculating burst size:

  Burst size = bandwidth x allowable time for burst traffic / 8

  Or

Burst size = interface mtu x 10

For information about configuring the burst size, see “Determining Proper Burst Size for Traffic Policers” on page 1002.

NOTE: There is a finite buffer space for an interface. In general, the estimated total buffer depth for an interface is about 125 ms.

For a traffic flow that conforms to the configured limits (categorized as green traffic), packets are implicitly marked with a packet loss priority (PLP) level of low and are allowed to pass through the interface unrestricted.

For a traffic flow that exceeds the configured limits (categorized as red traffic), packets are handled according to the traffic-policing actions configured for the policer. This example discards packets that burst over the 15 KBps limit.

To rate-limit Layer 3 traffic, you can apply a two-color policer in the following ways:

- Directly to a logical interface, at a specific protocol level.
- As the action of a standard stateless firewall filter that is applied to a logical interface, at a specific protocol level. This is the technique used in this example.

To rate-limit Layer 2 traffic, you can apply a two-color policer as a logical interface policer only. You cannot apply a two-color policer to Layer 2 traffic through a firewall filter.

CAUTION: You can choose either bandwidth-limit or bandwidth percent within the policer, as they are mutually exclusive. You cannot configure a policer to use bandwidth percent for aggregate, tunnel, and software interfaces.

In this example, the host is a traffic generator emulating a webserver. Devices R1 and R2 are owned by a service provider. The webserver is accessed by users on Device Host2. Device Host1 will be sending traffic with a source TCP HTTP port of 80 to the users. A single-rate two-color policer is configured and applied to the interface on Device R1 that connects to Device Host1. The policer enforces the contractual bandwidth availability made between the owner of the webserver and the service provider that owns Device R1 for the web traffic that flows over the link that connects Device Host1 to Device R1.

In accordance with the contractual bandwidth availability made between the owner of the webserver and the service provider that owns Devices R1 and R2, the policer will limit the HTTP port 80 traffic originating from Device Host1 to using 700 Mbps (70 percent) of the available bandwidth with an allowable burst rate of 10 x the MTU size of the gigabit Ethernet interface between the host Device Host1 and Device R1.

NOTE: In a real-world scenario you would probably also rate limit traffic for a variety of other ports such as FTP, SFTP, SSH, TELNET, SMTP, IMAP, and POP3 because they are often included as additional services with web hosting services.

NOTE: You need to leave some additional bandwidth available that is not rate limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

Topology

This example uses the topology in Figure 70 on page 1035.

Figure 70: Single-Rate Two-Color Policer Scenario

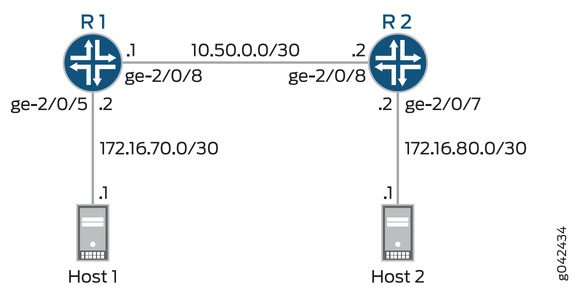
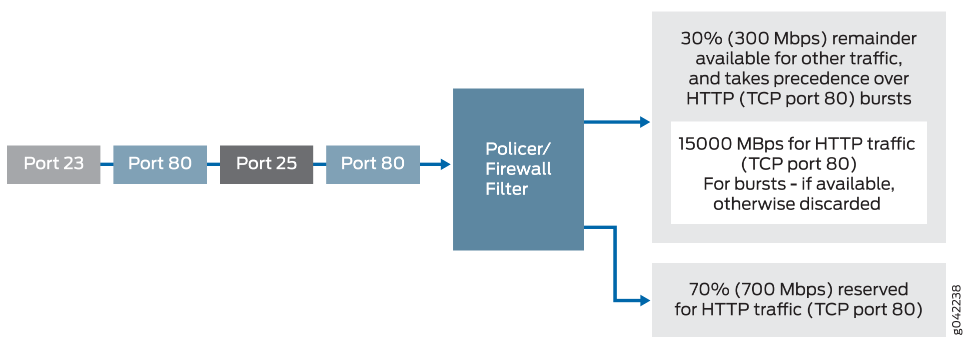


Figure 71 on page 1035 shows the policing behavior.

Figure 71: Traffic Limiting in a Single-Rate Two-Color Policer Scenario



Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, and then copy and paste the commands into the CLI at the **[edit]** hierarchy level.

Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set firewall policer discard if-exceeding bandwidth-limit 700m
set firewall policer discard if-exceeding burst-size-limit 15k
set firewall policer discard then discard
set firewall family inet filter mf-classifier term t1 from protocol tcp
set firewall family inet filter mf-classifier term t1 from port 80
set firewall family inet filter mf-classifier term t1 then policer discard
set firewall family inet filter mf-classifier term t2 then accept
set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Device R2

```
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces ge-2/0/7 description to-Host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.14.1/32
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set ge-2/0/5 description to-Host
user@R1# set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set ge-2/0/8 description to-R2
user@R1# set ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set lo0 unit 0 description looback-interface
user@R1# set lo0 unit 0 family inet address 192.168.13.1/32
```

2. Apply the firewall filter to interface ge-2/0/5 as an input filter.

```
[edit interfaces ge-2/0/5 unit 0 family inet]
user@R1# set filter input mf-classifier
```

3. Configure the policer to rate-limit to a bandwidth of 700 Mbps and a burst size of 15000 KBps for HTTP traffic (TCP port 80).

```
[edit firewall policer discard]
user@R1# set if-exceeding bandwidth-limit 700m
```



```
user@R1# set if-exceeding burst-size-limit 15k
```

4. Configure the policer to discard packets in the red traffic flow.

```
[edit firewall policer discard]
user@R1# set then discard
```

5. Configure the two conditions of the firewall to accept all TCP traffic to port HTTP (port 80).

```
[edit firewall family inet filter mf-classifier]
user@R1# set term t1 from protocol tcp
user@R1# set term t1 from port 80
```

6. Configure the firewall action to rate-limit HTTP TCP traffic using the policer.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term t1 then policer discard
```

7. At the end of the firewall filter, configure a default action that accepts all other traffic.

   Otherwise, all traffic that arrives on the interface and is not explicitly accepted by the firewall is discarded.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term t2 then accept
```

8. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set ge-2/0/8 description to-R1
user@R1# set ge-2/0/7 description to-Host
user@R1# set lo0 unit 0 description looback-interface
user@R1# set ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@R1# set ge-2/0/7 unit 0 family inet address 172.16.80.2/30
user@R1# set lo0 unit 0 family inet address 192.168.14.1/32
```

2. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/7.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```


Results From configuration mode, confirm your configuration by entering the **show interfaces** , **show firewall**, and **show protocols ospf** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
ge-2/0/5 {
  description to-Host;
  unit 0 {
    family inet {
      filter {
        input mf-classifier;
      }
      address 172.16.70.2/30;
    }
  }
}
ge-2/0/8 {
  description to-R2;
  unit 0 {
    family inet {
      address 10.50.0.1/30;
    }
  }
}
lo0 {
  unit 0 {
    description looback-interface;
    family inet {
      address 192.168.13.1/32;
    }
  }
}
```

```
user@R1# show firewall
family inet {
  filter mf-classifier {
    term t1 {
      from {
        protocol tcp;
        port 80;
      }
      then policer discard;
    }
    term t2 {
      then accept;
    }
  }
}
policer discard {
  if-exceeding {
    bandwidth-limit 700m;
    burst-size-limit 15k;
  }
  then discard;
}
```



```
user@R1# show protocols ospf
area 0.0.0.0 {
  interface ge-2/0/5.0 {
    passive;
  }
  interface lo0.0 {
    passive;
  }
  interface ge-2/0/8.0;
}
```

If you are done configuring Device R1, enter **commit** from configuration mode.

```
user@R2# show interfaces
ge-2/0/7 {
  description to-Host;
  unit 0 {
    family inet {
      address 172.16.80.2/30;
    }
  }
}
ge-2/0/8 {
  description to-R1;
  unit 0 {
    family inet {
      address 10.50.0.2/30;
    }
  }
}
lo0 {
  unit 0 {
    description looback-interface;
    family inet {
      address 192.168.14.1/32;
    }
  }
}
```

```
user@R2# show protocols ospf
area 0.0.0.0 {
  interface ge-2/0/7.0 {
    passive;
  }
  interface lo0.0 {
    passive;
  }
  interface ge-2/0/8.0;
}
```

If you are done configuring Device R2, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Clearing the Counters on page 1040
- Sending TCP Traffic into the Network and Monitoring the Discards on page 1040

Clearing the Counters

Purpose Confirm that the firewall counters are cleared.

Action On Device R1, run the **clear firewall all** command to reset the firewall counters to 0.

```
user@R1> clear firewall all
```

Sending TCP Traffic into the Network and Monitoring the Discards

Purpose Make sure that the traffic of interest that is sent is rate-limited on the input interface (ge-2/0/5).

Action

1. Use a traffic generator to send 10 TCP packets with a source port of 80.

   The -s flag sets the source port. The -k flag causes the source port to remain steady at 80 instead of incrementing. The -c flag sets the number of packets to 10. The -d flag sets the packet size.

   The destination IP address of 172.16.80.1 belongs to Device Host 2 that is connected to Device R2. The user on Device Host 2 has requested a webpage from Device Host 1 (the webserver emulated by the traffic generator on Device Host 1). The packets that being rate-limited are sent from Device Host 1 in response to the request from Device Host 2.

   NOTE: In this example the policer numbers are reduced to a bandwidth limit of 8 Kbps and a burst size limit of 1500 KBps to ensure that some packets are dropped during this test.

   ```
   [root@host]# hping 172.16.80.1 -c 10 -s 80 -k -d 300
   ```

   ```
   [User@Host]#  hping 172.16.80.1 -c 10 -s 80 -k -d 350
   HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 350 data
    bytes
   len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.5 ms
   .
   .
   .
   --- 172.16.80.1 hping statistic ---
   10 packets transmitted, 6 packets received, 40% packet loss
   round-trip min/avg/max = 0.5/3000.8/7001.3 ms
   ```


2. On Device R1, check the firewall counters by using the **show firewall** command.

```
user@R1> show firewall


User@R1# run show firewall

Filter: __default_bpdu_filter__

Filter: mf-classifier
Policers:
Name                                                 Bytes              Packets
discard-t1                                            1560                4
```

Meaning In Steps 1 and 2 the output from both devices shows that 4 packets were discarded This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 KBps burst option for red out-of-contract HTTP port 80 traffic was exceeded.

Example: Configuring Interface and Firewall Filter Policers at the Same Interface

This example shows how to configure three single-rate two-color policers and apply the policers to the IPv4 input traffic at the same single-tag virtual LAN (VLAN) logical interface.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you configure three single-rate two-color policers and apply the policers to the IPv4 input traffic at the same single-tag VLAN logical interface. Two policers are applied to the interface through a firewall filter, and one policer is applied directly to the interface.

You configure one policer, named **p-all-1m-5k-discard**, to rate-limit traffic to 1 Mbps with a burst size of 5000 bytes. You apply this policer directly to IPv4 input traffic at the logical interface. When you apply a policer directly to protocol-specific traffic at a logical interface, the policer is said to be applied as an *interface policer*.

You configure the other two policers to allow burst sizes of 500 KB, and you apply these policers to IPv4 input traffic at the logical interface by using an IPv4 standard stateless firewall filter. When you apply a policer to protocol-specific traffic at a logical interface through a firewall filter action, the policer is said to be applied as a *firewall-filter policer*.

- You configure the policer named **p-icmp-500k-500k-discard** to rate-limit traffic to 500 Kbps with a burst size of 500 K bytes by discarding packets that do not conform to these limits. You configure one of the firewall filter terms to apply this policer to Internet Control Message Protocol (ICMP) packets.
- You configure the policer named **p-ftp-10p-500k-discard** to rate-limit traffic to a 10 percent bandwidth with a burst size of 500 KB by discarding packets that do not conform to these limits. You configure another firewall-filter term to apply this policer to File Transfer Protocol (FTP) packets.

A policer that you configure with a bandwidth limit expressed as a percentage value (rather than as an absolute bandwidth value) is called a *bandwidth policer*. Only single-rate two-color policers can be configured with a percentage bandwidth specification. By default, a bandwidth policer rate-limits traffic to the specified percentage of the line rate of the physical interface underlying the target logical interface.

Topology

You configure the target logical interface as a single-tag VLAN logical interface on a Fast Ethernet interface operating at 100 Mbps. This means that the policer you configure with the 10-percent bandwidth-limit (the policer that you apply to FTP packets) rate-limits the FTP traffic on this interface to 10 Mbps.

NOTE: In this example, you do not configure the bandwidth policer as a *logical-bandwidth policer*. Therefore, the percentage is based on the physical media rate rather than on the configured shaping rate of the logical interface.

The firewall filter that you configure to reference two of the policers must be configured as an *interface-specific filter*. Because the policer that is used to rate-limit FTP packets specifies the bandwidth limit as a percentage value, the firewall filter that references this policer must be configured as an interface-specific filter. Thus, if this firewall filter were to be applied to multiple interfaces instead of just the Fast Ethernet interface in this example, unique policers and counters would be created for each interface to which the filter is applied.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring the Single-Tag VLAN Logical Interface on page 1043
- Configuring the Three Policers on page 1044

- Configuring the IPv4 Firewall Filter on page 1046
- Applying the Interface Policer and Firewall Filter Policers to the Logical Interface on page 1048

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces fe-0/1/1 vlan-tagging
set interfaces fe-0/1/1 unit 0 vlan-id 100
set interfaces fe-0/1/1 unit 0 family inet address 10.20.15.1/24
set interfaces fe-0/1/1 unit 1 vlan-id 101
set interfaces fe-0/1/1 unit 1 family inet address 10.20.240.1/24
set firewall policer p-all-1m-5k-discard if-exceeding bandwidth-limit 1m
set firewall policer p-all-1m-5k-discard if-exceeding burst-size-limit 5k
set firewall policer p-all-1m-5k-discard then discard
set firewall policer p-ftp-10p-500k-discard if-exceeding bandwidth-percent 10
set firewall policer p-ftp-10p-500k-discard if-exceeding burst-size-limit 500k
set firewall policer p-ftp-10p-500k-discard then discard
set firewall policer p-icmp-500k-500k-discard if-exceeding bandwidth-limit 500k
set firewall policer p-icmp-500k-500k-discard if-exceeding burst-size-limit 500k
set firewall policer p-icmp-500k-500k-discard then discard
set firewall family inet filter filter-ipv4-with-limits interface-specific
set firewall family inet filter filter-ipv4-with-limits term t-ftp from protocol tcp
set firewall family inet filter filter-ipv4-with-limits term t-ftp from port ftp
set firewall family inet filter filter-ipv4-with-limits term t-ftp from port ftp-data
set firewall family inet filter filter-ipv4-with-limits term t-ftp then policer
  p-ftp-10p-500k-discard
set firewall family inet filter filter-ipv4-with-limits term t-icmp from protocol icmp
set firewall family inet filter filter-ipv4-with-limits term t-icmp then policer
  p-icmp-500k-500k-discard
set firewall family inet filter filter-ipv4-with-limits term catch-all then accept
set interfaces fe-0/1/1 unit 1 family inet filter input filter-ipv4-with-limits
set interfaces fe-0/1/1 unit 1 family inet policer input p-all-1m-5k-discard
```

Configuring the Single-Tag VLAN Logical Interface

Step-by-Step Procedure

To configure the single-tag VLAN logical interface:

1. Enable configuration of the Fast Ethernet interface.

   ```
   [edit]
   user@host# edit interfaces fe-0/1/1
   ```

2. Enable single-tag VLAN framing.

   ```
   [edit interfaces fe-0/1/1]
   user@host# set vlan-tagging
   ```

3. Bind VLAN IDs to the logical interfaces.

   ```
   [edit interfaces fe-0/1/1]
   user@host# set unit 0 vlan-id 100
   user@host# set unit 1 vlan-id 101
   ```


4. Configure IPv4 on the single-tag VLAN logical interfaces.

```
[edit interfaces fe-0/1/1]
user@host# set unit 0 family inet address 10.20.15.1/24
user@host# set unit 1 family inet address 10.20.240.1/24
```

Results Confirm the configuration of the VLAN by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
fe-0/1/1 {
  vlan-tagging;
  unit 0 {
    vlan-id 100;
    family inet {
      address 10.20.15.1/24;
    }
  }
  unit 1 {
    vlan-id 101;
    family inet {
      address 10.20.240.1/24;
    }
  }
}
```

Configuring the Three Policers

Step-by-Step Procedure To configure the three policers:

1. Enable configuration of a two-color policer that discards packets that do not conform to a bandwidth of 1 Mbps and a burst size of 5000 bytes.

   NOTE: You apply this policer directly to all IPv4 input traffic at the single-tag VLAN logical interface, so the packets will not be filtered before being subjected to rate limiting.

```
[edit]
user@host# edit firewall policer p-all-1m-5k-discard
```

2. Configure the first policer.

```
[edit firewall policer p-all-1m-5k-discard]
user@host# set if-exceeding bandwidth-limit 1m
user@host# set if-exceeding burst-size-limit 5k
user@host# set then discard
```

3. Enable configuration of a two-color policer that discards packets that do not conform to a bandwidth specified as “10 percent” and a burst size of 500,000 bytes.

   You apply this policer only to the FTP traffic at the single-tag VLAN logical interface.

   You apply this policer as the action of an IPv4 firewall filter term that matches FTP packets from TCP.

   ```
   [edit firewall policer p-all-1m-5k-discard]
   user@host# up

   [edit]
   user@host# edit firewall policer p-ftp-10p-500k-discard
   ```

4. Configure policing limits and actions.

   ```
   [edit firewall policer p-ftp-10p-500k-discard]
   user@host# set if-exceeding bandwidth-percent 10
   user@host# set if-exceeding burst-size-limit 500k
   user@host# set then discard
   ```

   Because the bandwidth limit is specified as a percentage, the firewall filter that references this policer must be configured as an interface-specific filter.

   NOTE: If you wanted this policer to rate-limit to 10 percent of the logical interface configured shaping rate (rather than to 10 percent of the physical interface media rate), you would need to include the logical-bandwidth-policer statement at the **[edit firewall policer p-all-1m-5k-discard]** hierarchy level. This type of policer is called a *logical-bandwidth policer*.

5. Enable configuration of the IPv4 firewall filter policer for ICMP packets.

   ```
   [edit firewall policer p-ftp-10p-500k-discard]
   user@host# up

   [edit]
   user@host# edit firewall policer p-icmp-500k-500k-discard
   ```

6. Configure policing limits and actions.

   ```
   [edit firewall policer p-icmp-500k-500k-discard]
   user@host# set if-exceeding bandwidth-limit 500k
   user@host# set if-exceeding burst-size-limit 500k
   user@host# set then discard
   ```

Results

Confirm the configuration of the policers by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer p-all-1m-5k-discard {
  if-exceeding {
    bandwidth-limit 1m;
    burst-size-limit 5k;
  }
  then discard;
}
policer p-ftp-10p-500k-discard {
  if-exceeding {
    bandwidth-percent 10;
    burst-size-limit 500k;
  }
  then discard;
}
policer p-icmp-500k-500k-discard {
  if-exceeding {
    bandwidth-limit 500k;
    burst-size-limit 500k;
  }
  then discard;
}
```

Configuring the IPv4 Firewall Filter

Step-by-Step Procedure

To configure the IPv4 firewall filter:

1. Enable configuration of the IPv4 firewall filter.

```
[edit]
user@host# edit firewall family inet filter filter-ipv4-with-limits
```

2. Configure the firewall filter as interface-specific.

```
[edit firewall family inet filter filter-ipv4-with-limits]
user@host# set interface-specific
```

The firewall filter must be interface-specific because one of the policers referenced is configured with a bandwidth limit expressed as a percentage value.

3. Enable configuration of a filter term to rate-limit FTP packets.

```
[edit firewall family inet filter filter-ipv4-with-limits]
user@host# edit term t-ftp

[edit firewall family inet filter filter-ipv4-with-limits term t-ftp]
user@host# set from protocol tcp
user@host# set from port [ ftp ftp-data ]
```

FTP messages are sent over TCP port 20 (**ftp**) and received over TCP port 21 (**ftp-data**).

4. Configure the filter term to match FTP packets.

```
[edit firewall family inet filter filter-ipv4-with-limits term t-ftp]
user@host# set then policer p-ftp-10p-500k-discard
```

5. Enable configuration of a filter term to rate-limit ICMP packets.

```
[edit firewall family inet filter filter-ipv4-with-limits term t-ftp]
user@host# up

[edit firewall family inet filter filter-ipv4-with-limits]
user@host# edit term t-icmp
```

6. Configure the filter term for ICMP packets

```
[edit firewall family inet filter filter-ipv4-with-limits term t-icmp]
user@host# set from protocol icmp
user@host# set then policer p-icmp-500k-500k-discard
```

7. Configure a filter term to accept all other packets without policing.

```
[edit firewall family inet filter filter-ipv4-with-limits term t-icmp]
user@host# up

[edit firewall family inet filter filter-ipv4-with-limits]
user@host# set term catch-all then accept
```

Results Confirm the configuration of the firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter filter-ipv4-with-limits {
    interface-specific;
    term t-ftp {
      from {
        protocol tcp;
        port [ ftp ftp-data ];
      }
      then policer p-ftp-10p-500k-discard;
    }
    term t-icmp {
      from {
        protocol icmp;
      }
      then policer p-icmp-500k-500k-discard;
    }
```



```
        term catch-all {
            then accept;
        }
    }
}
policer p-all-1m-5k-discard {
    if-exceeding {
        bandwidth-limit 1m;
        burst-size-limit 5k;
    }
    then discard;
}
policer p-ftp-10p-500k-discard {
    if-exceeding {
        bandwidth-percent 10;
        burst-size-limit 500k;
    }
    then discard;
}
policer p-icmp-500k-500k-discard {
    if-exceeding {
        bandwidth-limit 500k;
        burst-size-limit 500k;
    }
    then discard;
}
```

Applying the Interface Policer and Firewall Filter Policers to the Logical Interface

Step-by-Step Procedure

To apply the three policers to the VLAN:

1. Enable configuration of IPv4 on the logical interface.

   ```
   [edit]
   user@host# edit interfaces fe-0/1/1 unit 1 family inet
   ```

2. Apply the firewall filter policers to the interface.

   ```
   [edit interfaces fe-0/1/1 unit 1 family inet]
   user@host# set filter input filter-ipv4-with-limits
   ```

3. Apply the interface policer to the interface.

   ```
   [edit interfaces fe-0/1/1 unit 1 family inet]
   user@host# set policer input p-all-1m-5k-discard
   ```

   Input packets at **fe-0/1/1.0** are evaluated against the interface policer before they are evaluated against the firewall filter policers. For more information, see "Order of Policer and Firewall Filter Operations" on page 979.

Results Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
fe-0/1/1 {
    vlan-tagging;
    unit 0 {
        vlan-id 100;
        family inet {
            address 10.20.15.1/24;
        }
    }
    unit 1 {
        vlan-id 101;
        family inet {
            filter {
                input filter-ipv4-with-limits;
            }
            policer {
                input p-all-1m-5k-discard;
            }
            address 10.20.240.1/24;
        }
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying Policers Applied Directly to the Logical Interface on page 1049
- Displaying Statistics for the Policer Applied Directly to the Logical Interface on page 1050
- Displaying the Policers and Firewall Filters Applied to an Interface on page 1050
- Displaying Statistics for the Firewall Filter Policers on page 1051

Displaying Policers Applied Directly to the Logical Interface

Purpose Verify that the interface policer is evaluated when packets are received on the logical interface.

Action Use the show interfaces policers operational mode command for logical interface **fe-0/1/1.1**. The command output section for the **Proto** column and **Input Policer** column shows that the policer **p-all-1m-5k-discard** is evaluated when packets are received on the logical interface.

```
user@host> show interfaces policers fe-0/1/1.1
Interface       Admin Link Proto Input Policer         Output Policer
fe-0/1/1.1      up    up
```

```
inet  p-all-1m-5k-discard-fe-0/1/1.1-inet-i
```

In this example, the interface policer is applied to logical interface traffic in the input direction only.

Displaying Statistics for the Policer Applied Directly to the Logical Interface

Purpose Verify the number of packets evaluated by the interface policer.

Action Use the show policer operational mode command and optionally specify the name of the policer. The command output displays the number of packets evaluated by each configured policer (or the specified policer), in each direction.

```
user@host> show policer p-all-1m-5k-discard-fe-0/1/1.1-inet-i
Policers:
Name                                                 Bytes              Packets
p-all-1m-5k-discard-fe-0/1/1.1-inet-i                  200                    5
```

Displaying the Policers and Firewall Filters Applied to an Interface

Purpose Verify that the firewall filter **filter-ipv4-with-limits** is applied to the IPv4 input traffic at logical interface **fe-0/1/1.1**.

Action Use the show interfaces statistics operational mode command for logical interface **fe-0/1/1.1**, and include the **detail** option. Under the **Protocol inet** section of the command output section, the **Input Filters** and **Policer** lines display the names of filter and policer applied to the logical interface in the input direction.

```
user@host> show interfaces statistics fe-0/1/1.1 detail
  Logical interface fe-0/1/1.1 (Index 83) (SNMP ifIndex 545) (Generation 153)
    Flags: SNMP-Traps 0x4000 VLAN-Tag [ 0x8100.100 ]  Encapsulation: ENET2
    Traffic statistics:
     Input  bytes  :                   0
     Output bytes  :                  46
     Input  packets:                   0
     Output packets:                   1
    Local statistics:
     Input  bytes  :                   0
     Output bytes  :                  46
     Input  packets:                   0
     Output packets:                   1
    Transit statistics:
     Input  bytes  :                   0                    0 bps
     Output bytes  :                   0                    0 bps
     Input  packets:                   0                    0 pps
     Output packets:                   0                    0 pps
    Protocol inet, MTU: 1500, Generation: 176, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Input Filters: filter-ipv4-with-limits-fe-0/1/1.1-i
      Policer: Input: p-all-1m-5k-discard-fe-0/1/1.1-inet-i
      Addresses, Flags: Is-Preferred Is-Primary
```

```
Destination: 10.20.130/24, Local: 10.20.130.1, Broadcast: 10.20.130.255,

Generation: 169
```

In this example, the two firewall filter policers are applied to logical interface traffic in the input direction only.

Displaying Statistics for the Firewall Filter Policers

Purpose Verify the number of packets evaluated by the firewall filter policers.

Action Use the **show firewall** operational mode command for the filter you applied to the logical interface.

```
[edit]
user@host> show firewall filter filter-ipv4-with-limits-fe-0/1/1.1-i

Filter: filter-ipv4-with-limits-fe-0/1/1.1-i
Policers:
Name                                              Bytes             Packets
p-ftp-10p-500k-discard-t-ftp-fe-0/1/1.1-i             0                   0
p-icmp-500k-500k-discard-t-icmp-fe-0/1/1.1-i          0                   0
```

The command output displays the names of the policers (**p-ftp-10p-500k-discard** and **p-icmp-500k-500k-discard**), combined with the names of the filter terms (**t-ftp** and **t-icmp**, respectively) under which the policer action is specified. The policer-specific output lines display the number of packets that matched the filter term. This is only the number of out-of-specification (out-of-spec) packet counts, not all packets policed by the policer.

Related Documentation

- Order of Policer and Firewall Filter Operations on page 979
- Two-Color Policer Configuration Overview on page 1027
- Guidelines for Applying Traffic Policers on page 983
- Determining Proper Burst Size for Traffic Policers on page 1002

Bandwidth Policers

- Bandwidth Policer Overview on page 1051
- Example: Configuring a Logical Bandwidth Policer on page 1053

Bandwidth Policer Overview

For a single-rate two-color policer only, you can specify the bandwidth limit as a percentage value from 1 through 100 instead of as an absolute number of bits per second. This type of two-color policer, called a *bandwidth policer*, rate-limits traffic to a bandwidth limit that is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.

Guidelines for Configuring a Bandwidth Policer

The following guidelines apply to configuring a bandwidth policer:

- To specify a percentage bandwidth limit, you include the bandwidth-percent ***percentage*** statement in place of the bandwidth-limit ***bps*** statement.
- By default, a bandwidth policer calculates the percentage bandwidth limit based on the physical interface port speed. To configure a bandwidth policer to calculate the percentage bandwidth limit based on the configured logical interface shaping rate instead, include the logical-bandwidth-policer statement at the **[edit firewall policer *policer-name*]** hierarchy level. This type of bandwidth policer is called a *logical bandwidth policer*.

  You can configure a logical interface shaping rate by including the **shaping-rate *bps*** statement at the **[edit class-of-service interfaces interface *interface-name* unit *logical-unit-number*]** hierarchy level. A logical interface shaping rate causes the specified amount of bandwidth to be allocated to the logical interface.

  NOTE: If you configure a logical-bandwidth policer and then apply the policer to a logical interface that is not configured with a shaping rate, then the policer rate-limits traffic on that logical interface to calculate the percentage bandwidth limit based on the physical interface port speed, even if you include the logical-bandwidth-policer statement in the bandwidth policer configuration.

- If you reference a bandwidth policer from a stateless firewall filter term, you must include the interface-specific statement in the firewall filter configuration.

Guidelines for Applying a Bandwidth Policer

The following guidelines pertain to applying a bandwidth policer to traffic:

- You can use a bandwidth policer to rate-limit protocol-specific traffic (not **family any**) at the input or output of a logical interface.
- You can apply a bandwidth policer directly to protocol-specific input or output traffic at a logical interface.
- To send only selected packets to a bandwidth policer, you can reference the bandwidth policer from a stateless firewall filter term and then apply the filter to logical interface traffic for a specific protocol family.
  - To reference a *logical bandwidth policer* from a firewall filter, you must include the interface-specific statement in the firewall filter configuration.

  - You cannot use a bandwidth policer for forwarding-table filters.

- You cannot apply a bandwidth policer to an aggregate interface, a tunnel interface, or a software interface.

Example: Configuring a Logical Bandwidth Policer

This example shows how to configure a logical bandwidth policer.

Requirements

Before you begin, make sure that you have two logical units available on a Gigabit Ethernet interface.

Overview

In this example, you configure a single-rate two-color policer that specifies the bandwidth limit as a percentage value rather than as an absolute number of bits per second. This type of policer is called a *bandwidth policer*. By default, a bandwidth policer enforces a bandwidth limit based on the line rate of the underlying physical interface. As an option, you can configure a bandwidth policer to enforce a bandwidth limit based on the configured shaping rate of the logical interface. To configure this type of bandwidth policer, called a *logical bandwidth policer*, you include the logical-bandwidth-policer statement in the policer configuration.

To configure a logical interface shaping rate, include the **shaping-rate *bps*** statement at the **[edit class-of-service interfaces interface *interface-name* unit *logical-unit-number*]** hierarchy level. This class-of-service (CoS) configuration statement causes the specified amount of bandwidth to be allocated to the logical interface.

NOTE: **If you configure a policer bandwidth limit as a percentage but a shaping rate is not configured for the target logical interface, the policer bandwidth limit is calculated as a percentage of the physical interface media rate, even if you enable the logical-bandwidth policing feature.**

To apply a logical bandwidth policer to a logical interface, you can apply the policer directly to the logical interface at the protocol family level or (if you only need to rate-limit filtered packets) you can reference the policer from a stateless firewall filter configured to operate in *interface-specific* mode.

Topology

In this example, you configure two logical interfaces on a single Gigabit Ethernet interface and configure a shaping rate on each logical interface. On logical interface **ge-1/3/0.0**, you allocate 4 Mbps of bandwidth. On logical interface **ge-1/3/0.1**, you allocate 2 Mbps of bandwidth.

You also configure a logical bandwidth policer with a bandwidth limit of 50 percent and a maximum burst size of 125,000 bytes, and then you apply the policer to input and output traffic at the logical units configured on **ge-1/3/0.0**. For logical interface **ge-1/3/0.0**, the policer rate-limits to a bandwidth limit of 2 Mbps (50 percent of the 4 Mbps shaping rate configured for the logical interface). For logical interface **ge-1/3/0.1**, the policer rate-limits traffic to a bandwidth limit of 1 Mbps (50 percent of the 2 Mbps shaping rate configured for the logical interface).

If no shaping rate is configured for a target logical interface, the policer rate-limits to a bandwidth limit calculated as 50 percent of the physical interface media rate. For example, if you apply a 50 percent bandwidth policer to input or output traffic at a Gigabit Ethernet logical interface without rate shaping, the policer applies a bandwidth limit of 500 Mbps (50 percent of 1000 Mbps).

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring the Logical Interfaces on page 1055
- Configuring Traffic Rate-Shaping by Specifying the Amount of Bandwidth to be Allocated to the Logical Interface on page 1056
- Configuring the Logical Bandwidth Policer on page 1056
- Applying the Logical Bandwidth Policers to the Logical Interfaces on page 1057

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces ge-1/3/0 per-unit-scheduler
set interfaces ge-1/3/0 vlan-tagging
set interfaces ge-1/3/0 unit 0 vlan-id 100
set interfaces ge-1/3/0 unit 0 family inet address 172.16.1.1/30
set interfaces ge-1/3/0 unit 1 vlan-id 200
set interfaces ge-1/3/0 unit 1 family inet address 172.16.1.1/30
set class-of-service interfaces ge-1/3/0 unit 0 shaping-rate 4m
set class-of-service interfaces ge-1/3/0 unit 1 shaping-rate 2m
set firewall policer LB-policer logical-bandwidth-policer
set firewall policer LB-policer if-exceeding bandwidth-percent 50
set firewall policer LB-policer if-exceeding burst-size-limit 125k
set firewall policer LB-policer then discard
set interfaces ge-1/3/0 unit 0 family inet policer input LB-policer
set interfaces ge-1/3/0 unit 0 family inet policer output LB-policer
set interfaces ge-1/3/0 unit 1 family inet policer input LB-policer
set interfaces ge-1/3/0 unit 1 family inet policer output LB-policer
```


Configuring the Logical Interfaces

Step-by-Step Procedure

To configure the logical interfaces:

1. Enable configuration of the physical interface.

```
[edit]
user@host# edit interfaces ge-1/3/0

[edit interfaces ge-1/3/0]
user@host# set per-unit-scheduler
user@host# set vlan-tagging
```

2. Configure the first logical interface.

```
[edit interfaces ge-1/3/0]
user@host# set unit 0 vlan-id 100
user@host# set unit 0 family inet address 172.16.1.1/30
```

3. Configure the second logical interface.

```
[edit interfaces ge-1/3/0]
user@host# set unit 1 vlan-id 200
user@host# set unit 1 family inet address 172.16.1.1/30
```

Results

Confirm the configuration of the interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/0 {
  per-unit-scheduler;
  vlan-tagging;
  unit 0 {
    vlan-id 100;
    family inet {
      address 172.16.1.1/30;
    }
  }
  unit 1 {
    vlan-id 200;
    family inet {
      address 172.2.16.1/30;
    }
  }
}
```

Configuring Traffic Rate-Shaping by Specifying the Amount of Bandwidth to be Allocated to the Logical Interface

Step-by-Step Procedure

To configure rate shaping by specifying the bandwidth to be allocated to the logical interface:

1. Enable CoS configuration on the physical interface.

   ```
   [edit]
   user@host# edit class-of-service interfaces ge-1/3/0
   ```

2. Configure rate shaping for the logical interfaces.

   ```
   [edit class-of-service interfaces ge-1/3/0]
   user@host# set unit 0 shaping-rate 4m
   user@host# set unit 1 shaping-rate 2m
   ```

   These statements allocate 4 Mbps of bandwidth to logical unit **ge-1/3/0.0** and 2 Mbps of bandwidth to logical unit **ge-1/3/0.1**.

Results

Confirm the configuration of the rate shaping by entering the **show class-of-service** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show class-of-service
interfaces {
  ge-1/3/0 {
    unit 0 {
      shaping-rate 4m;
    }
    unit 1 {
      shaping-rate 2m;
    }
  }
}
```

Configuring the Logical Bandwidth Policer

Step-by-Step Procedure

To configure the logical bandwidth policer:

1. Enable configuration of a single-rate two-color policer.

   ```
   [edit]
   user@host# edit firewall policer LB-policer
   ```

2. Configure the policer as a logical-bandwidth policer.

   ```
   [edit firewall policer LB-policer]
   user@host# set logical-bandwidth-policer
   ```

   This applies the rate-limiting to logical interfaces.

3. Configure the policer traffic limits and actions.

```
[edit firewall policer LB-policer]
user@host# set if-exceeding bandwidth-percent 50
user@host# set if-exceeding burst-size-limit 125k
user@host# set then discard
```

Results Confirm the configuration of the policer by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer LB-policer {
  logical-bandwidth-policer;
  if-exceeding {
    bandwidth-percent 50;
    burst-size-limit 125k;
  }
  then discard;
}
```

Applying the Logical Bandwidth Policers to the Logical Interfaces

Step-by-Step Procedure To configure the logical bandwidth policers to the logical interfaces:

1. Enable configuration of the interface.

```
[edit]
user@host# edit interfaces ge-1/3/0
```

2. Apply the logical bandwidth policer to the first logical interface.

```
[edit interfaces ge-1/3/0]
user@host# set unit 0 family inet policer input LB-policer
user@host# set unit 0 family inet policer output LB-policer
```

3. Apply the policing to the second logical interface.

```
[edit interfaces ge-1/3/0]
user@host# set unit 1 family inet policer input LB-policer
user@host# set unit 1 family inet policer output LB-policer
```

Results Confirm the configuration of the interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/0 {
  per-unit-scheduler;
  vlan-tagging;
```



```
    unit 0 {
      vlan-id 100;
      family inet {
        policer {
          input LB-policer;
          output LB-policer;
        }
        address 172.16.1.1/30;
      }
    }
    unit 1 {
      vlan-id 200;
      family inet {
        policer {
          input LB-policer;
          output LB-policer;
        }
        address 172.16.1.1/30;
      }
    }
  }
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying Traffic Statistics and Policers for the Logical Interface on page 1058
- Displaying Statistics for the Policer on page 1059

Displaying Traffic Statistics and Policers for the Logical Interface

Purpose Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

Action Use the **show interfaces** operational mode command for logical interfaces **ge-1/3/0.0** and **ge-1/3/0.1**, and include the **detail** or **extensive** option. The command output section for **Traffic statistics** lists the number of bytes and packets received and transmitted on the logical interface, and the **Protocol inet** section contains a **Policer** field that lists the policer **LB-policer** as an input or output policer as follows:

- **Input: LB-policer-ge-1/3/0.0-inet-i**
- **Output: LB-policer-ge-1/3/0.0-inet-o**

In this example, the policer is applied to logical interface traffic in both the input and output directions.

```
user@host> show interfaces ge-1/3/0.0 detail
  Logical interface ge-1/3/0.0 (Index 80) (SNMP ifIndex 154) (Generation 150)
    Flags: SNMP-Traps 0x4000 VLAN-Tag [ 0x8100.100 ]  Encapsulation: ENET2
    Traffic statistics:
     Input  bytes  :                    0
```



```
     Output bytes  :                   46
     Input  packets:                    0
     Output packets:                    1
    Local statistics:
     Input  bytes  :                    0
     Output bytes  :                   46
     Input  packets:                    0
     Output packets:                    1
    Transit statistics:
     Input  bytes  :                    0                    0 bps
     Output bytes  :                    0                    0 bps
     Input  packets:                    0                    0 pps
     Output packets:                    0                    0 pps
    Protocol inet, MTU: 1500, Generation: 174, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Policer: Input: LB-policer-ge-1/3/0.0-inet-i, Output:
LB-policer-ge-1/3/0.0-inet-o
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 172.16.1.0/30, Local: 172.16.1.1, Broadcast: 172.16.1.3,
Generation: 165

user@host> show interfaces ge-1/3/0.1 detail
  Logical interface ge-1/3/0.1 (Index 81) (SNMP ifIndex 543) (Generation 151)
    Flags: SNMP-Traps 0x4000 VLAN-Tag [ 0x8100.200 ]  Encapsulation: ENET2
    Traffic statistics:
     Input  bytes  :                    0
     Output bytes  :                   46
     Input  packets:                    0
     Output packets:                    1
    Local statistics:
     Input  bytes  :                    0
     Output bytes  :                   46
     Input  packets:                    0
     Output packets:                    1
    Transit statistics:
     Input  bytes  :                    0                    0 bps
     Output bytes  :                    0                    0 bps
     Input  packets:                    0                    0 pps
     Output packets:                    0                    0 pps
    Protocol inet, MTU: 1500, Generation: 175, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Policer: Input: LB-policer-ge-1/3/0.1-inet-i, Output:
LB-policer-ge-1/3/0.1-inet-o
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 172.17.1.0/30, Local: 172.17.1.1, Broadcast: 172.17.1.3,
Generation: 167
```

Displaying Statistics for the Policer

Purpose Verify the number of packets evaluated by the policer.

Action

Use the show policer operational mode command and optionally specify the name of the policer. The command output displays the number of packets evaluated by each configured policer (or the specified policer), in each direction. For the policer **LB-policer**, the input and output policer names are displayed as follows:

- **LB-policer-ge-1/3/0.0-inet-i**
- **LB-policer-ge-1/3/0.0-inet-o**
- **LB-policer-ge-1/3/0.1-inet-i**
- **LB-policer-ge-1/3/0.1-inet-o**

The **-inet-i** suffix denotes a policer applied to logical interface input traffic, while the **-inet-o** suffix denotes a policer applied to logical interface output traffic. In this example, the policer is applied to both input and output traffic on logical interface **ge-1/3/0.0** and logical interface **ge-1/3/0.1**.

```
user@host> show policer
Policers:
Name                                              Packets
__default_arp_policer__                                 0
LB-policer-ge-1/3/0.0-inet-i                            0
LB-policer-ge-1/3/0.0-inet-o                            0
LB-policer-ge-1/3/0.1-inet-i                            0
LB-policer-ge-1/3/0.1-inet-o                            0
```

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Guidelines for Applying Traffic Policers on page 983
- **bandwidth-percent on page 1278**
- **interface-specific (Firewall Filters) on page 1261**
- **logical-bandwidth-policer on page 1308**

- *shaping-rate (Applying to an Interface)*

Filter-Specific Counters and Policers

Filter-Specific Policer Overview

By default, a policer operates in *term-specific* mode so that, for a given firewall filter, the Junos OS creates a separate policer instance for every filter term that references the policer. As an option, you can configure a policer to operate in *filter-specific* mode so that a single policer instance is used by all terms (within the same firewall filter) that reference the policer.

For an IPv4 firewall filter with multiple terms that reference the same policer, configuring the policer to operate in filter-specific mode enables you to count and monitor the activity of the policer at the firewall filter level.

NOTE: Term-specific mode and filter-specific mode also apply to prefix-specific policer sets.

To enable a single-rate two-color policer to operate in filter-specific mode, you can include the filter-specific statement at the following hierarchy levels:

- **[edit firewall policer *policer-name*]**
- **[edit logical-systems *logical-system-name* firewall policer *policer-name*]**

You can reference filter-specific policers from IPv4 (**family inet**) firewall filters only.

Example: Configuring a Stateless Firewall Filter to Protect Against TCP and ICMP Floods

This example shows how to create a stateless firewall filter that protects against TCP and ICMP denial-of-service attacks.

- Requirements on page 1061
- Overview on page 1061
- Configuration on page 1063
- Verification on page 1067

Requirements

No special configuration beyond device initialization is required before configuring stateless firewall filters.

Overview

In this example we create a stateless firewall filter called **protect-RE** to police TCP and ICMP packets. It uses the policers described here:

- **tcp-connection-policer**—This policer limits TCP traffic to 1,000,000 bits per second (bps) with a maximum burst size of 15,000 bytes. Traffic exceeding either limit is discarded.
- **icmp-policer**—This policer limits ICMP traffic to 1,000,000 bps with a maximum burst size of 15,000 bytes. Traffic exceeding either limit is discarded.

When specifying limits, the bandwidth limit can be from 32,000 bps to 32,000,000,000 bps and the burst-size limit can be from 1,500 bytes through 100,000,000 bytes. Use the following abbreviations when specifying limits: k (1,000), m (1,000,000), and g (1,000,000,000).

Each policer is incorporated into the action of a filter term. This example includes the following terms:

- **tcp-connection-term**—Polices certain TCP packets with a source address of 192.168.0.0/24 or 10.0.0.0/24. These addresses are defined in the **trusted-addresses** prefix list.

  Filtered packets include **tcp-established** packets The **tcp-established** match condition is an alias for the bit-field match condition **tcp-flags "(ack | rst)"**, which indicates an established TCP session, but not the first packet of a TCP connection.

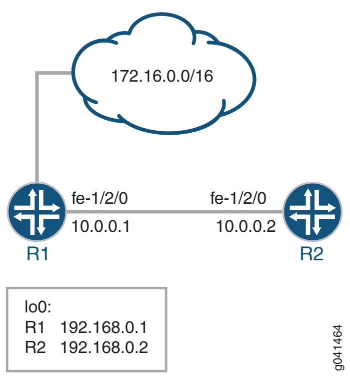
- **icmp-term**—Polices ICMP packets. All ICMP packets are counted in the **icmp-counter** counter.

NOTE: You can move terms within the firewall filter by using the insert command. See *insert* in the *CLI User Guide*.

You can apply a stateless firewall to the input or output sides, or both, of an interface. To filter packets transiting the device, apply the firewall filter to any non-Routing Engine interface. To filter packets originating from, or destined for, the Routing Engine, apply the firewall filter to the loopback (lo0) interface.

Figure 49 on page 720 shows the sample network.

Figure 72: Firewall Filter to Protect Against TCP and ICMP Floods



Because this firewall filter limits Routing Engine traffic to TCP packets, routing protocols that use other transport protocols for Layer 4 cannot successfully establish sessions when this filter is active. To demonstrate, this example sets up OSPF between Device R1 and Device R2.

"CLI Quick Configuration" on page 720 shows the configuration for all of the devices in Figure 49 on page 720.

The section "Step-by-Step Procedure" on page 721 describes the steps on Device R2.

Configuration

CLI Quick Configuration

To quickly configure the stateless firewall filter, copy the following commands to a text file, remove any line breaks, and then paste the commands into the CLI.

Device R1

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32 primary
set interfaces lo0 unit 0 family inet address 172.16.0.1/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext peer-as 200
set protocols bgp group ext neighbor 10.0.0.2
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 100
```

Device R2

```
set interfaces fe-1/2/0 unit 0 family inet address 10.0.0.2/30
set interfaces lo0 unit 0 family inet filter input protect-RE
set interfaces lo0 unit 0 family inet address 192.168.0.2/32 primary
set interfaces lo0 unit 0 family inet address 172.16.0.2/32
set protocols bgp group ext type external
set protocols bgp group ext export send-direct
set protocols bgp group ext neighbor 10.0.0.1 peer-as 100
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface fe-1/2/0.0
set policy-options prefix-list trusted-addresses 10.0.0.0/24
set policy-options prefix-list trusted-addresses 192.168.0.0/24
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 200
set firewall family inet filter protect-RE term tcp-connection-term from source-prefix-list
  trusted-addresses
set firewall family inet filter protect-RE term tcp-connection-term from protocol tcp
set firewall family inet filter protect-RE term tcp-connection-term from tcp-established
set firewall family inet filter protect-RE term tcp-connection-term then policer
  tcp-connection-policer
set firewall family inet filter protect-RE term tcp-connection-term then accept
set firewall family inet filter protect-RE term icmp-term from source-prefix-list
  trusted-addresses
set firewall family inet filter protect-RE term icmp-term from protocol icmp
set firewall family inet filter protect-RE term icmp-term then policer icmp-policer
set firewall family inet filter protect-RE term icmp-term then count icmp-counter
set firewall family inet filter protect-RE term icmp-term then accept
set firewall policer tcp-connection-policer filter-specific
set firewall policer tcp-connection-policer if-exceeding bandwidth-limit 1m
set firewall policer tcp-connection-policer if-exceeding burst-size-limit 15k
set firewall policer tcp-connection-policer then discard
set firewall policer icmp-policer filter-specific
set firewall policer icmp-policer if-exceeding bandwidth-limit 1m
set firewall policer icmp-policer if-exceeding burst-size-limit 15k
```



```
set firewall policer icmp-policer then discard
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure stateless firewall filter to discard :

1. Configure the device interfaces.

```
[edit interfaces fe-1/2/0 unit 0 family inet ]
user@R2# set address 10.0.0.2/30

[edit interfaces lo0 unit 0 family inet]
user@R2# set address 192.168.0.2/32 primary
user@R2# set address 172.16.0.2/32
```

2. Configure the BGP peering session.

```
[edit protocols bgp group ext]
user@R2# set type external
user@R2# set export send-direct
user@R2# set neighbor 10.0.0.1 peer-as 100
```

3. Configure the autonomous system (AS) number and router ID.

```
[edit routing-options]
user@R2# set autonomous-system 200
user@R2# set router-id 192.168.0.2
```

4. Configure OSPF.

```
[edit protocols ospf area 0.0.0.0]
user@R2# set interface lo0.0 passive
user@R2# set interface fe-1/2/0.0
```

5. Define the list of trusted addresses.

```
[edit policy-options prefix-list trusted-addresses]
user@R2# set 10.0.0.0/24
user@R2# set 192.168.0.0/24
```

6. Configure a policy to advertise direct routes.

```
[edit policy-options policy-statement send-direct term 1]
user@R2# set from protocol direct
user@R2# set then accept
```

7. Configure the TCP policer.

```
[edit firewall policer tcp-connection-policer]
user@R2# set filter-specific
user@R2# set if-exceeding bandwidth-limit 1m
```

```
user@R2# set if-exceeding burst-size-limit 15k
user@R2# set then discard
```

8. Create the ICMP policer.

```
[edit firewall policer icmp-policer]
user@R2# set filter-specific
user@R2# set if-exceeding bandwidth-limit 1m
user@R2# set if-exceeding burst-size-limit 15k
user@R2# set then discard
```

9. Configure the TCP filter rules.

```
[edit firewall family inet filter protect-RE term tcp-connection-term]
user@R2# set from source-prefix-list trusted-addresses
user@R2# set from protocol tcp
user@R2# set from tcp-established
user@R2# set then policer tcp-connection-policer
user@R2# set then accept
```

10. Configure the ICMP filter rules.

```
[edit firewall family inet filter protect-RE term icmp-term]
user@R2# set from source-prefix-list trusted-addresses
user@R2# set from protocol icmp
user@R2# set then policer icmp-policer
user@R2# set then count icmp-counter
user@R2# set then accept
```

11. Apply the filter to the loopback interface.

```
[edit interfaces lo0 unit 0]
user@R2# set family inet filter input protect-RE
```

Results Confirm your configuration by entering the **show interfaces**, **show protocols**, **show policy-options**, **show routing-options**, and **show firewall** commands from configuration mode. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
fe-1/2/0 {
    unit 0 {
        family inet {
            address 10.0.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            filter {
                input protect-RE;
```

```
            }
            address 192.168.0.2/32 {
                primary;
            }
            address 172.16.0.2/32;
        }
    }
}
```

user@R2# **show protocols**

```
bgp {
    group ext {
        type external;
        export send-direct;
        neighbor 10.0.0.1 {
            peer-as 100;
        }
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface fe-1/2/0.0;
    }
}
```

user@R2# **show policy-options**

```
prefix-list trusted-addresses {
    10.0.0.0/24;
    192.168.0.0/24;
}
policy-statement send-direct {
    term 1 {
        from protocol direct;
        then accept;
    }
}
```

user@R2# **show routing-options**

```
router-id 192.168.0.2;
autonomous-system 200;
```

user@R2# **show firewall**

```
family inet {
    filter protect-RE {
        term tcp-connection-term {
            from {
                source-prefix-list {
                    trusted-addresses;
                }
                protocol tcp;
                tcp-established;
            }
            then {
                policer tcp-connection-policer;
```

```
            accept;
          }
        }
        term icmp-term {
          from {
            source-prefix-list {
              trusted-addresses;
            }
            protocol icmp;
          }
          then {
            policer icmp-policer;
            count icmp-counter;
            accept;
          }
        }
      }
    }
    policer tcp-connection-policer {
      filter-specific;
      if-exceeding {
        bandwidth-limit 1m;
        burst-size-limit 15k;
      }
      then discard;
    }
    policer icmp-policer {
      filter-specific;
      if-exceeding {
        bandwidth-limit 1m;
        burst-size-limit 15k;
      }
      then discard;
    }
    }
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

NOTE: To verify the TCP policer, you can use a packet generation tool. This task is not shown here.

- Displaying Stateless Firewall Filter That Are in Effect on page 1068
- Using telnet to Verify the tcp-established Condition in the TCP Firewall Filter on page 1068
- Using telnet to Verify the Trusted Prefixes Condition in the TCP Firewall Filter on page 1069
- Using OSPF to Verify the TCP Firewall Filter on page 1070
- Verifying the ICMP Firewall Filter on page 1071

Displaying Stateless Firewall Filter That Are in Effect

Purpose Verify the configuration of the firewall filter.

Action From operational mode, enter the **show firewall** command.

```
user@R2> show firewall
Filter: protect-RE
Counters:
Name                                                  Bytes              Packets
icmp-counter                                              0                    0
Policers:
Name                                                  Bytes              Packets
icmp-policer                                                               0
tcp-connection-policer                                                     0
```

Meaning The output shows the filter, the counter, and the policers that are in effect on Device R2.

Using telnet to Verify the tcp-established Condition in the TCP Firewall Filter

Purpose Make sure that telnet traffic works as expected.

Action Verify that the device can establish only TCP sessions with hosts that meet the **from tcp-established** condition.

1. From Device R2, make sure that the BGP session with Device R1 is established.

```
user@R2> show bgp summary | match down
Groups: 1 Peers: 1 Down peers: 0
```

2. From Device R2, telnet to Device R1.

```
user@R2> telnet 192.168.0.1
Trying 192.168.0.1...
Connected to R1.example.net.
Escape character is '^]'.

R1 (ttyp4)

login:
```

3. From Device R1, telnet to Device R2.

```
user@R1> telnet 192.168.0.2
Trying 192.168.0.2...
telnet: connect to address 192.168.0.2: Operation timed out
telnet: Unable to connect to remote host
```

4. On Device R2, deactivate the **from tcp-established** match condition.

   [edit firewall family inet filter protect-RE term tcp-connection-term]
   user@R2# **deactivate from tcp-established**

user@R2# **commit**

5. From Device R1, try again to telnet to Device R2.

```
user@R1> telnet 192.168.0.1
Trying 192.168.0.2...
Connected to R2.example.net.
Escape character is '^]'.

R2 (ttyp4)

login:
```

Meaning Verify the following information:

- As expected , the BGP session is established. The **from tcp-established** match condition is not expected to block BGP session establishment.
- From Device R2, you can telnet to Device R1. Device R1 has no firewall filter configured, so this is the expected behavior.
- From Device R1, you cannot telnet to Device R2. Telnet uses TCP as the transport protocol, so this result might be surprising. The cause for the lack of telnet connectivity is the **from tcp-established** match condition. This match condition limits the type of TCP traffic that is accepted of Device R2. After this match condition is deactivated, the telnet session is successful.

Using telnet to Verify the Trusted Prefixes Condition in the TCP Firewall Filter

Purpose Make sure that telnet traffic works as expected.

Action Verify that the device can establish only telnet sessions with a host at an IP address that matches one of the trusted source addresses. For example, log in to the device with the **telnet** command from another host with one of the trusted address prefixes. Also, verify that telnet sessions with untrusted source addresses are blocked.

1. From Device R1, telnet to Device R2 from an untrusted source address.

```
user@R1> telnet 172.16.0.2 source 172.16.0.1
Trying 172.16.0.2...
^C
```

2. From Device R2, add 172.16/16 to the list of trusted prefixes.

   [edit policy-options prefix-list trusted-addresses]
   user@R2# **set 172.16.0.0/16**
   user@R2# **commit**

3. From Device R1, try again to telnet to Device R2.

```
user@R1> telnet 172.16.0.2 source 172.16.0.1
Trying 172.16.0.2...
Connected to R2.example.net.
Escape character is '^]'.
```

```
R2 (ttyp4)

login:
```

Meaning Verify the following information:

- From Device R1, you cannot telnet to Device R2 with an unstrusted source address. After the 172.16/16 prefix is added to the list of trusted prefixes, the telnet request from source address 172.16.0.1 is accepted.
- OSPF session establishment is blocked. OSPF does not use TCP as its transport protocol. After the **from protocol tcp** match condition is deactivated, OSPF session establishment is not blocked.

Using OSPF to Verify the TCP Firewall Filter

Purpose Make sure that OSPF traffic works as expected.

Action Verify that the device cannot establish OSPF connectivity.

1. From Device R1, check the OSPF sessions.

```
user@R1> show ospf neighbor
Address          Interface              State     ID               Pri  Dead
10.0.0.2         fe-1/2/0.0             Init      192.168.0.2      128    34
```

2. From Device R2, check the OSPF sessions.

```
user@R2> show ospf neighbor
```

3. From Device R2, remove the **from protocol tcp** match condition.

```
[edit firewall family inet filter protect-RE term tcp-connection-term]
user@R2# deactivate from protocol
user@R2# commit
```

4. From Device R1, recheck the OSPF sessions.

```
user@R1> show ospf neighbor
Address          Interface              State     ID               Pri  Dead
10.0.0.2         fe-1/2/0.0             Full      192.168.0.2      128    36
```

5. From Device R2, recheck the OSPF sessions.

```
user@R2> show ospf neighbor
Address          Interface              State     ID               Pri  Dead
10.0.0.1         fe-1/2/0.0             Full      192.168.0.1      128    39
```

Meaning Verify the following information:

- OSPF session establishment is blocked. OSPF does not use TCP as its transport protocol. After the **from protocol tcp** match condition is deactivated, OSPF session establishment is successful.

Verifying the ICMP Firewall Filter

Purpose Verify that ICMP packets are being policed and counted. Also make sure that ping requests are discarded when the requests originate from an untrusted source address.

Action

1. Undo the configuration changes made in previous verification steps.

   Reactivate the TCP firewall settings, and delete the 172.16/16 trusted source address.

   ```
   [edit firewall family inet filter protect-RE term tcp-connection-term]
   user@R2# activate from protocol
   user@R2# activate from tcp-established

   [edit policy-options prefix-list trusted-addresses]
   user@R2# delete 172.16.0.0/16

   user@R2# commit
   ```

2. From Device R1, ping the loopback interface on Device R2.

   ```
   user@R1> ping 192.168.0.2 rapid count 600 size 2000
   PING 192.168.0.2 (192.168.0.2): 2000 data bytes
   !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
   --- 192.168.0.2 ping statistics ---
   600 packets transmitted, 536 packets received, 10% packet loss
   pinground-trip min/avg/max/stddev = 2.976/3.405/42.380/2.293 ms
   ```

3. From Device R2, check the firewall statistics.

   ```
   user@R2> show firewall

   Filter: protect-RE
   Counters:
   Name                                                Bytes              Packets
   icmp-counter                                      1180804                 1135
   Policers:
   Name                                                Bytes              Packets
   icmp-policer                                                            66
   tcp-connection-policer                                                   0
   ```

4. From an untrusted source address on Device R1, send a ping request to Device R2's loopback interface.

   ```
   user@R1> ping 172.16.0.2 source 172.16.0.1

   PING 172.16.0.2 (172.16.0.2): 56 data bytes
   ^C
   --- 172.16.0.2 ping statistics ---
   14 packets transmitted, 0 packets received, 100% packet loss
   ```


Meaning Verify the following information:

- The ping output shows that 10% packet loss is occurring.
- The ICMP packet counter is incrementing, and the icmp-policer is incrementing.
- Device R2 does not send ICMP responses to the **ping 172.16.0.2 source 172.16.0.1** command.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Guidelines for Applying Traffic Policers on page 983
- Prefix-Specific Counting and Policing Actions on page 1072

Prefix-Specific Counting and Policing Actions

Prefix-Specific Counting and Policing Overview

This topic covers the following information:

Separate Counting and Policing for Each IPv4 Address Range

Prefix-specific counting and policing enables you to configure an IPv4 firewall filter term that matches on a source or destination address, applies a single-rate two-color policer as the term action, but associates the matched packet with a specific counter and policer instance based on the source or destination in the packet header. You can implicitly create a separate counter or policer instance for a single address or for a group of addresses.

Prefix-specific counting and policing uses a *prefix-specific action* configuration that specifies the name of the policer you want to apply, whether prefix-specific counting is to be enabled, and a source or destination address prefix range.

The prefix range specifies between 1 and 16 sequential set bits of an IPv4 address mask. The length of the prefix range determines the size of the counter and policer *set*, which consists of as few as 2 or as many as 65,536 counter and policer instances. The position of the bits of the prefix range determines the indexing of filter-matched packets into the set of instances.

NOTE: A prefix-specific action is specific to a source or destination *prefix range*, but it is not specific to a particular source or destination *address range*, and it is not specific to a particular interface.

To apply a prefix-specific action to the traffic at an interface, you configure a firewall filter term that matches on source or destination addresses, and then you apply the firewall filter to the interface. The flow of filtered traffic is rate-limited using prefix-specific counter and policer instances that are selected per packet based on the source or destination address in the header of the filtered packet.

Prefix-Specific Action Configuration

To configure a prefix-specific action, you specify the following information:

- Prefix-specific action name—Name that can be referenced as the action of an IPv4 standard firewall filter term that matches packets on source or destination addresses.
- Policer name—Name of a single-rate two-color policer for which you want to implicitly create prefix-specific instances.

  NOTE: For aggregated Ethernet interfaces, you can configure a prefix-specific action that references a logical interface policer (also called an aggregate policer). You can reference this type of prefix-specific action from an IPv4 standard firewall filter and then apply the filter at the aggregate level of the interface.

- Counting option—Option to include if you want to enable prefix-specific counters.
- Filter-specific option—Option to include if you want a single counter and policer set to be shared across all terms in the firewall filter. A prefix-specific action that operates in this way is said to operate in *filter-specific* mode. If you do not enable this option, the prefix-specific action operates in *term-specific* mode, meaning that a separate counter and policer set is created for each filter term that references the prefix-specific action.
- Source address prefix length—Length of the address prefix, from 0 through 32, to be used with a packet matched on the source address.
- Destination address prefix length—Length of the address prefix, from 0 through 32, to be used with a packet matched on the destination address.
- Subnet prefix length—Length of the subnet prefix, from 0 through 32, to be used with a packet matched on either the source or destination address.

You must configure source and destination address prefix lengths to be from 1 to 16 bits longer than the subnet prefix length. If you configure source or destination address prefix lengths to be more than 16 bits beyond the configured subnet prefix length, an error occurs when you try to commit the configuration.

Counter and Policer Set Size and Indexing

The number of prefix-specific actions (counters or policers) implicitly created for a prefix-specific action is determined by the length of the address prefix and the length of the subnet prefix:

Size of Counter and Policer Set = 2 ^ (*source-or-destination-prefix-length* - *subnet-prefix-length*)

Table 69 on page 1074 shows examples of counter and policer set size and indexing.

Table 69: Examples of Counter and Policer Set Size and Indexing

Example Prefix Lengths Specified in the Prefix-Specific Action	Calculation of Counter or Policer Set Size	Indexing of Instances	
source-prefix-length = 32 *subnet-prefix-length* = 16	Size = 2 ^ (32 - 16) = 2 ^16 = 65,536 instances NOTE: This calculation shows the largest counter or policer set size supported.	Instance 0:	*x.x*.0.0
		Instance 1:	*x.x*.0.1
		Instance 65535:	*x.x*.255.255
source-prefix-length = 32 *subnet-prefix-length* = 24	Size = 2 ^ (32 - 24) = 2 ^ 8 = 256 instances	Instance 0:	*x.x.x*.0
		Instance 1:	*x.x.x*.1
		Instance 255:	*x.x.x*.255
source-prefix-length = 32 *subnet-prefix-length* = 25	Size = 2 ^ (32 - 25) = 2 ^ 7 = 128 instances	Instance 0:	*x.x.x*.0
		Instance 1:	*x.x.x*.1
		Instance 127:	*x.x.x*.127
source-prefix-length = 24 *subnet-prefix-length* = 20	Size = 2 ^ (24 - 20) = 2 ^ 4 = 16 instances	Instance 0:	*x.x*.0.*x*
		Instance 1:	*x.x*.1.*x*
		Instance 15:	*x.x*.15.*x*

Filter-Specific Counter and Policer Set Overview

By default, a prefix-specific policer set operates in *term-specific* mode so that, for a given firewall filter, the Junos OS creates a separate counter and policer set for every filter term that references the prefix-specific action. As an option, you can configure a prefix-specific policer set to operate in *filter-specific* mode so that a single prefix-specific policer set is used by all terms (within the same firewall filter) that reference the policer.

For an IPv4 firewall filter with multiple terms that reference the same prefix-specific policer set, configuring the policer set to operate in filter-specific mode enables you to count and monitor the activity of the policer set at the firewall filter level.

NOTE: Term-specific mode and filter-specific mode also apply to policers. See "Filter-Specific Policer Overview" on page 1060.

To enable a prefix-specific policer set to operate in filter-specific mode, you can include the filter-specific statement at the following the hierarchy levels:

- **[edit firewall family inet prefix-action *prefix-action-name*]**
- **[edit logical-systems *logical-system-name* firewall family inet prefix-action *prefix-action-name*]**

You can reference filter-specific, prefix-specific policer sets from IPv4 (**family inet**) firewall filters only.

Example: Configuring Prefix-Specific Counting and Policing

This example shows how to configure prefix-specific counting and policing.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

In this example, you configure prefix-specific counting and policing based on the last octet of the source address field in packets matched by an IPv4 firewall filter.

The single-rate two-color policer named **1Mbps-policer** rate-limits traffic to a bandwidth of 1,000,000 bps and a burst-size limit of 63,000 bytes, discarding any packets in a traffic flow that exceeds the traffic limits.

Independent of the IPv4 addresses contained in any packets passed from a firewall filter, the prefix-specific action named **psa-1Mbps-per-source-24-32-256** specifies a set of 256 counters and policers, numbered from 0 through 255. For each packet, the last octet of the source address field is used to index into the associated prefix-specific counter and policer in the set:

- Packets with a source address ending with the octet 0x0000 00000 index the first counter and policer in the set.

- Packets with a source address ending with the octet 0x0000 0001 index the second counter and policer in the set.
- Packets with a source address ending with the octet 0x1111 1111 index the last counter and policer in the set.

The **limit-source-one-24** firewall filter contains a single term that matches all packets from the **/24** subnet of source address **10.10.10.0**, passing these packets to the prefix-specific action **psa-1Mbps-per-source-24-32-256**.

Topology

In this example, because the filter term matches the **/24** subnet of a single source address, each counting and policing instance in the prefix-specific set is used for only one source address.

- Packets with a source address **10.10.10.0** index the first counter and policer in the set.
- Packets with a source address **10.10.10.1** index the second counter and policer in the set.
- Packets with a source address **10.10.10.255** index the last counter and policer in the set.

This example shows the simplest case of prefix-specific actions, in which the filter term matches on one address with a prefix length that is the same as the prefix length specified in the prefix-specific action for indexing into the set of prefix-specific counters and policers.

For descriptions of other configurations for prefix-specific counting and policing, see "Prefix-Specific Counting and Policing Configuration Scenarios" on page 1082.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall policer 1Mbps-policer if-exceeding bandwidth-limit 1m
set firewall policer 1Mbps-policer if-exceeding burst-size-limit 63k
set firewall policer 1Mbps-policer then discard
set firewall family inet prefix-action psa-1Mbps-per-source-24-32-256 policer
  1Mbps-policer
set firewall family inet prefix-action psa-1Mbps-per-source-24-32-256 count
```

```
set firewall family inet prefix-action psa-1Mbps-per-source-24-32-256
  subnet-prefix-length 24
set firewall family inet prefix-action psa-1Mbps-per-source-24-32-256 source-prefix-length
  32
set firewall family inet filter limit-source-one-24 term one from source-address
  10.10.10.0/24
set firewall family inet filter limit-source-one-24 term one then prefix-action
  psa-1Mbps-per-source-24-32-256
set interfaces so-0/0/2 unit 0 family inet filter input limit-source-one-24
set interfaces so-0/0/2 unit 0 family inet address 10.39.1.1/16
```

Configuring a Policer for Prefix-Specific Counting and Policing

Step-by-Step Procedure

To configure a policer to be used for prefix-specific counting and policing:

1. Enable configuration of a single-rate two-color policer.

```
[edit]
user@host# edit firewall policer 1Mbps-policer
```

2. Define the traffic limit.

```
[edit firewall policer 1Mbps-policer]
user@host# set if-exceeding bandwidth-limit 1m
user@host# set if-exceeding burst-size-limit 63k
```

   Packets in a traffic flow that conforms to this limit are passed with the PLP set to **low**.

3. Define the actions for nonconforming traffic.

```
[edit firewall policer 1Mbps-policer]
user@host# set then discard
```

   Packets in a traffic flow that exceeds this limit are discarded. Other configurable actions for a single-rate two-color policer are to set the forwarding class and to set the PLP level.

Results

Confirm the configuration of the policer by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer 1Mbps-policer {
  if-exceeding {
    bandwidth-limit 1m;
    burst-size-limit 63k;
  }
  then discard;
}
```

Configuring a Prefix-Specific Action Based on the Policer

Step-by-Step Procedure

To configure a prefix-specific action that references the policer and specifies a portion of a source address prefix:

1. Enable configuration of a prefix-specific action.

   ```
   [edit]
   user@host# edit firewall family inet prefix-action psa-1Mbps-per-source-24-32-256
   ```

   Prefix-specific counting and policing can be defined for IPv4 traffic only.

2. Reference the policer for which a prefix-specific set is to be created.

   ```
   [edit firewall family inet prefix-action psa-1Mbps-per-source-24-32-256]
   user@host# set policer 1Mbps-policer
   user@host# set count
   ```

   NOTE: For aggregated Ethernet interfaces, you can configure a prefix-specific action that references a logical interface policer (also called an aggregate policer). You can reference this type of prefix-specific action from an IPv4 standard firewall filter and then apply the filter at the aggregate level of the interface.

3. Specify the prefix range on which IPv4 addresses are to be indexed to the counter and policer set.

   ```
   [edit firewall family inet prefix-action psa-1Mbps-per-source-24-32-256]
   user@host# set source-prefix-length 32
   user@host# set subnet-prefix-length 24
   ```

Results

Confirm the configuration of the prefix-specific action by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer 1Mbps-policer {
  if-exceeding {
    bandwidth-limit 1m;
    burst-size-limit 63k;
  }
  then discard;
}
family inet {
  prefix-action psa-1Mbps-per-source-24-32-256 {
    policer 1Mbps-policer;
    subnet-prefix-length 24;
    source-prefix-length 32;
```

```
    }
  }
```

Configuring an IPv4 Filter That References the Prefix-Specific Action

Step-by-Step Procedure

To configure an IPv4 standard firewall filter that references the prefix-specific action:

1. Enable configuration of the IPv4 standard firewall filter.

   ```
   [edit]
   user@host# edit firewall family inet filter limit-source-one-24
   ```

   Prefix-specific counting and policing can be defined for IPv4 traffic only.

2. Configure the filter term to match on the packet source address or destination address.

   ```
   [edit firewall family inet filter limit-source-one-24]
   user@host# set term one from source-address 10.10.10.0/24
   ```

3. Configure the filter term to reference the prefix-specific action.

   ```
   [edit firewall family inet filter limit-source-one-24]
   user@host# set term one then prefix-action psa-1Mbps-per-source-24-32-256
   ```

   You could also use the **next term** action to configure all Hypertext Transfer Protocol (HTTP) traffic to each host to transmit at 500 Kbps and have the total HTTP traffic limited to 1 Mbps.

Results

Confirm the configuration of the prefix-specific action by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer 1Mbps-policer {
  if-exceeding {
    bandwidth-limit 1m;
    burst-size-limit 63k;
  }
  then discard;
}
family inet {
  prefix-action psa-1Mbps-per-source-24-32-256 {
    policer 1Mbps-policer;
    subnet-prefix-length 24;
    source-prefix-length 32;
  }
  filter limit-source-one-24 {
    term one {
      from {
        source-address {
          10.10.10.0/24;
        }
```

```
            }
            then prefix-action psa-1Mbps-per-source-24-32-256;
         }
      }
   }
```

Applying the Firewall Filter to IPv4 Input Traffic at a Logical Interface

Step-by-Step Procedure

To apply the firewall filter to IPv4 input traffic at a logical interface:

1. Enable configuration of IPv4 on the logical interface.

```
[edit]
user@host# edit interfaces so-0/0/2 unit 0 family inet
```

2. Configure an IP address.

```
[edit interfaces so-0/0/2 unit 0 family inet]
user@host# set address 10.39.1.1/16
```

3. Apply the IPv4 standard stateless firewall filter.

```
[edit interfaces so-0/0/2 unit 0 family inet]
user@host# set filter input limit-source-one-24
```

Results

Confirm the configuration of the prefix-specific action by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
so-0/0/2 {
   unit 0 {
      family inet {
         filter {
            input limit-source-one-24;
         }
         address 10.39.1.1/16;
      }
   }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying the Firewall Filters Applied to an Interface on page 1081
- Displaying Prefix-Specific Actions Statistics for the Firewall Filter on page 1081

Displaying the Firewall Filters Applied to an Interface

Purpose Verify that the firewall filter **limit-source-one-24** is applied to the IPv4 input traffic at logical interface **so-0/0/2.0**.

Action Use the show interfaces statistics operational mode command for logical interface **so-0/0/2.0**, and include the **detail** option. In the command output section for **Protocol inet**, the **Input Filters** field displays **limit-source-one-24**, indicating that the filter is applied to IPv4 traffic in the input direction:

```
user@host> show interfaces statistics so-0/0/2.0 detail
  Logical interface so-0/0/2.0 (Index 79) (SNMP ifIndex 510) (Generation 149)
    Flags: Hardware-Down Point-To-Point SNMP-Traps 0x4000 Encapsulation: PPP
    Protocol inet, MTU: 4470, Generation: 173, Route table: 0
      Flags: Sendbcast-pkt-to-re, Protocol-Down
      Input Filters: limit-source-one-24
      Addresses, Flags: Dest-route-down Is-Preferred Is-Primary
        Destination: 10.39/16, Local: 10.39.1.1, Broadcast: 10.39.255.255,
Generation: 163
```

Displaying Prefix-Specific Actions Statistics for the Firewall Filter

Purpose Verify the number of packets evaluated by the policer.

Action Use the show firewall prefix-action-stats **filter** ***filter-name*** **prefix-action** ***name*** operational mode command to display statistics about a prefix-specific action configured on a firewall filter.

As an option, you can use the **from *set-index* to *set-index*** command option to specify the starting and ending counter or policer to be displayed. A policer set is indexed from 0 through 65535.

The command output displays the specified filter name followed by a listing of the number of bytes and packets processed by each policer in the policer set.

For a term-specific policer, each policer in the set is identified as follows:

prefix-specific-action-name-term-name-set-index

For a filter-specific policer, each policer is identified in the command output as follows:

prefix-specific-action-name-set-index

Because the example prefix-specific action **psa-1Mbps-per-source-24-32-256** is referenced by only one term of the example filter **limit-source-one-24**, the example policer **1Mbps-policer** is configured as term-specific. In the **show firewall prefix-action-stats** command output, the policer statistics are displayed as **psa-1Mbps-per-source-24-32-256-one-0**, **psa-1Mbps-per-source-24-32-256-one-1**, and so on through **psa-1Mbps-per-source-24-32-256-one-255**.

```
user@host> show firewall prefix-action-stats filter limit-source-one-24 prefix-action
psa-1Mbps-per-source-24-32-256 from 0 to 9
```

```
Filter: limit-source-one-24
Counters:
Name                                                Bytes    Packets
psa-1Mbps-per-source-24-32-256-one-0                    0          0
psa-1Mbps-per-source-24-32-256-one-1                    0          0
psa-1Mbps-per-source-24-32-256-one-2                    0          0
psa-1Mbps-per-source-24-32-256-one-3                    0          0
psa-1Mbps-per-source-24-32-256-one-4                    0          0
psa-1Mbps-per-source-24-32-256-one-5                    0          0
psa-1Mbps-per-source-24-32-256-one-6                    0          0
psa-1Mbps-per-source-24-32-256-one-7                    0          0
psa-1Mbps-per-source-24-32-256-one-8                    0          0
psa-1Mbps-per-source-24-32-256-one-9                    0          0
```

Prefix-Specific Counting and Policing Configuration Scenarios

This topic covers the following information:

Prefix Length of the Action and Prefix Length of Addresses in Filtered Packets

Table 70 on page 1082 describes the relationship between the prefix length specified in the prefix-specific action and the prefix length of the addresses matched by the firewall filter term that references the prefix-specific action.

Table 70: Summary of Prefix-Specific Action Scenarios

Counter and Policer Set	Packet-Filtering Criteria	Indexing of Instances	
Prefix-specific action scenario: “Example: Configuring Prefix-Specific Counting and Policing” on page 1075			
source-prefix-length = 32 *subnet-prefix-length* = 24 Set size: 2^8 = 256 Instance numbers: 0 - 255	*source-address* = 10.10.10.0/24	Instance 0	10.10.10.0
		Instance 1:	10.10.10.1
		...	...
		Instance 255:	10.10.10.255

Table 70: Summary of Prefix-Specific Action Scenarios *(continued)*

Counter and Policer Set	Packet-Filtering Criteria	Indexing of Instances	
Prefix-specific action scenario: "Scenario 1: Firewall Filter Term Matches on Multiple Addresses" on page 1083			
source-prefix-length = 32 *subnet-prefix-length* = 24 Set size: 2^8 = 256 Instance numbers: 0 - 255	*source-address* = 10.10.10.0/24 *source-address* = 10.11.0.0/16	Instance 0	10.10.10.0, 10.11.*x*.0
		Instance 1:	10.10.10.1, 10.11.*x*.1
		...	...
		Instance 255:	10.10.10.255, 10.11.*x*.255
	For addresses in the /16 subnet, *x* ranges from 0 through 255.		
Prefix-specific action scenario: "Scenario 2: Subnet Prefix Is Longer Than the Prefix in the Filter Match Condition" on page 1085			
source-prefix-length = 32 *subnet-prefix-length* = 25 Set size: 2^7 = 128 Instance numbers: 0 - 127	*source-address* = 10.10.10.0/24	Instance 0	10.10.10.0, 10.10.10.128
		Instance 1:	10.10.10.1, 10.10.10.120
		...	...
		Instance 127:	10.10.10.255, 10.10.10.127
Prefix-specific action scenario: "Scenario 3: Subnet Prefix Is Shorter Than the Prefix in the Firewall Filter Match Condition" on page 1086			
source-prefix-length = 32 *subnet-prefix-length* = 24 Set size: 2^8 = 256 Instance numbers: 0 - 255	*source-address* = 10.10.10.0/25 NOTE: Only packets with source addresses ranging from **10.10.10.0** through **10.10.10.127** are passed to the prefix-specific action.	Instance 0	10.10.10.0
		Instance 1:	10.10.10.1
		...	...
		Instance 127:	10.10.10.127
		Instances 128 – 255: unused	

Scenario 1: Firewall Filter Term Matches on Multiple Addresses

The complete example, "Example: Configuring Prefix-Specific Counting and Policing" on page 1075, shows the simplest case of prefix-specific actions, in which a single-term firewall filter matches on one address with a prefix length that is the same as the subnet prefix length specified in the prefix-specific action. Unlike the example, this scenario describes a configuration in which a single-term firewall filter matches on two IPv4 source addresses. In addition, the additional condition matches on a source address with a prefix length that is different from the subnet prefix length defined in the prefix-specific action. In this case, the additional condition matches on the **/16** subnet of the source address **10.11.0.0**.

NOTE: **Unlike packets that match the source address 10.10.10.0/24, packets that match the source address 10.11.0.0/16 are in a many-to-one correspondence with the instances in the counter and policer set.**

The filter-matched packets that are passed to the prefix-specific action index into the counter and policer set in such a way that the counting and policing instances are shared by packets that contain source addresses across the **10.10.10.0/24** and **10.11.0.0/16** subnets as follows:

- The first counter and policer in the set are indexed by packets with source addresses **10.10.10.0** and **10.11.*x*.0**, where *x* ranges from **0** through **255**.
- The second counter and policer in the set are indexed by packets with source addresses **10.10.10.1** and **10.11.*x*.1**, where *x* ranges from **0** through **255**.
- The 256th (last) counter and policer in the set are indexed by packets with source addresses **10.10.10.255** and **10.11.*x*.255**, where *x* ranges from **0** through **255**.

The following configuration shows the statements for configuring the single-rate two-color policer, the prefix-specific action that references the policer, and the IPv4 standard stateless firewall filter that references the prefix-specific action:

```
[edit]
firewall {
  policer 1Mbps-policer {
    if-exceeding {
      bandwidth-limit 1m;
      burst-size-limit 63k;
    }
    then discard;
  }
  family inet {
    prefix-action psa-1Mbps-per-source-24-32-256 {
      policer 1Mbps-policer;
      subnet-prefix-length 24;
      source-prefix-length 32;
    }
    filter limit-source-two-24-16 {
      term one {
        from {
          source-address {
            10.10.10.0/24;
            10.11.0.0/16;
          }
        }
```



```
            then prefix-action psa-1Mbps-per-source-24-32-256;
          }
        }
      }
    }
    interfaces {
      so-0/0/2 {
        unit 0 {
          family inet {
            filter {
              input limit-source-two-24-16;
            }
            address 10.39.1.1/16;
          }
        }
      }
    }
```

Scenario 2: Subnet Prefix Is Longer Than the Prefix in the Filter Match Condition

The complete example, "Example: Configuring Prefix-Specific Counting and Policing" on page 1075, shows the simplest case of prefix-specific actions, in which the single-term firewall filter matches on one address with a prefix length that is the same as the subnet prefix length specified in the prefix-specific action. Unlike the example, this scenario describes a configuration in which the prefix-specific action defines a subnet prefix length that is longer than the prefix of the source address matched by the firewall filter. In this case, the prefix-specific action defines a subnet-prefix value of **25**, while the firewall filter matches on a source address in the **/24** subnet.

NOTE: **The firewall filter passes the prefix-specific action packets with source addresses that range from 10.10.10.0 through 10.10.10.255, while the prefix-specific action specifies a set of only 128 counters and policers, numbered from 0 through 127.**

The filter-matched packets that are passed to the prefix-specific action index into the counter and policer set in such a way that the counting and policing instances are shared by packets that contain either of two source addresses within the **10.10.10.0/24** subnet:

- The first counter and policer in the set are indexed by packets with source addresses **10.10.10.0** and **10.10.10.128**.
- The second counter and policer in the set are indexed by packets with source addresses **10.10.10.1** and **10.10.10.129**.
- The 128th (last) counter and policer in the set are indexed by packets with source addresses **10.10.10.127** and **10.10.10.255**.

The following configuration shows the statements for configuring the single-rate two-color policer, the prefix-specific action that references the policer, and the IPv4 standard stateless firewall filter that references the prefix-specific action:

```
[edit]
```

```
firewall {
  policer 1Mbps-policer {
    if-exceeding {
      bandwidth-limit 1m;
      burst-size-limit 63k;
    }
    then discard;
  }
  family inet {
    prefix-action psa-1Mbps-per-source-25-32-128 {
      policer 1Mbps-policer;
      subnet-prefix-length 25;
      source-prefix-length 32;
    }
    filter limit-source-one-24 {
      term one {
        from {
          source-address {
            10.10.10.0/24;
          }
        }
        then prefix-action psa-1Mbps-per-source-25-32-128;
      }
    }
  }
}
interfaces {
  so-0/0/2 {
    unit 0 {
      family inet {
        filter {
          input limit-source-one-24;
        }
        address 10.39.1.1/16;
      }
    }
  }
}
```

Scenario 3: Subnet Prefix Is Shorter Than the Prefix in the Firewall Filter Match Condition

The complete example, "Example: Configuring Prefix-Specific Counting and Policing" on page 1075, shows the simplest case of prefix-specific actions, in which the single-term firewall filter matches on one address with a prefix length that is the same as the subnet prefix length specified in the prefix-specific action. Unlike the example, this scenario describes a configuration in which the prefix-specific action defines a subnet prefix length that is shorter than the prefix of the source address matched by the firewall filter. In this case, the filter term matches on the **/25** subnet of the source address **10.10.10.0**.

NOTE: **The firewall filter passes the prefix-specific action only packets with source addresses that range from 10.10.10.0 through 10.10.10.127, while the prefix-specific action specifies a set of 256 counters and policers, numbered from 0 through 255.**

The matched packets that are passed to the prefix-specific action index into the lower half of the counter and policer set only:

- The first counter and policer in the set are indexed by packets with source address **10.10.10.0**.
- The second counter and policer in the set are indexed by packets with source address **10.10.10.1** and **10.10.10.129**.
- The 128th counter and policer in the set are indexed by packets with source address **10.10.10.127**.
- The upper half of the set (instances numbered from 128 through 255) are not indexed by packets passed to the prefix-specific action from this particular firewall filter.

The following configuration shows the statements for configuring the single-rate two-color policer, the prefix-specific action that references the policer, and the IPv4 standard stateless firewall filter that references the prefix-specific action:

```
[edit]
firewall {
  policer 1Mbps-policer {
    if-exceeding {
      bandwidth-limit 1m;
      burst-size-limit 63k;
    }
    then discard;
  }
  family inet {
    prefix-action psa-1Mbps-per-source-24-32-256 {
      policer 1Mbps-policer;
      subnet-prefix-length 24;
      source-prefix-length 32;
    }
    filter limit-source-one-25 {
      term one {
        from {
          source-address {
            10.10.10.0/25;
          }
        }
        then prefix-action psa-1Mbps-per-source-24-32-256;
      }
    }
  }
}
interfaces {
  so-0/0/2 {
```



```
            unit 0 {
                family inet {
                    filter {
                        input limit-source-one-25;
                    }
                    address 10.39.1.1/16;
                }
            }
        }
    }
```

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Guidelines for Applying Traffic Policers on page 983

Multifield Classification

Multifield Classification Overview

This topic covers the following information:

Forwarding Classes and PLP Levels

You can configure the Junos OS class of service (CoS) features to classify incoming traffic by associating each packet with a forwarding class, a packet loss priority (PLP) level, or both:

- Based on the associated forwarding class, each packet is assigned to an output queue, and the router services the output queues according to the associated scheduling you configure.
- Based on the associated PLP, each packet carries a lower or higher likelihood of being dropped if congestion occurs. The CoS random early detection (RED) process uses the drop probability configuration, output queue fullness percentage, and packet PLP to drop packet as needed to control congestion at the output stage.

Multifield Classification and BA Classification

The Junos OS supports two general types of packet classification: behavior aggregate (BA) classification and multifield classification:

- BA classification, or CoS value traffic classification, refers to a method of packet classification that uses a CoS configuration to set the forwarding class or PLP of a packet based on the *CoS value* in the IP packet header. The CoS value examined for BA classification purposes can be the Differentiated Services code point (DSCP) value, DSCP IPv6 value, IP precedence value, MPLS EXP bits, and IEEE 802.1p value. The default classifier is based on the IP precedence value.
- Multifield classification refers to a method of packet classification that uses a standard stateless firewall filter configuration to set the forwarding class or PLP for each packet entering or exiting the interface based on *multiple fields* in the IP packet header, including the DSCP value (for IPv4 only), the IP precedence value, the MPLS EXP bits, and the IEEE 802.1p bits. Multifield classification commonly matches on IP address fields, the IP protocol type field, or the port number in the UDP or TCP pseudoheader field. Multifield classification is used instead of BA classification when you need to classify packets based on information in the packet information other than the CoS values only.

  With multifield classification, a firewall filter term can specify the packet classification actions for matching packets though the use of the **forwarding-class *class-name*** or **loss-priority (high | medium-high | medium-low | low)** nonterminating actions in the term's **then** clause.

NOTE: BA classification of a packet can be overridden by the stateless firewall filter actions **forwarding-class** and **loss-priority**.

Multifield Classification Used In Conjunction with Policers

To configure multifield classification in conjunction with rate limiting, a firewall filter term can specify the packet classification actions for matching packets through the use of a **policer** nonterminating action that references a single-rate two-color policer.

When multifield classification is configured to perform classification through a policer, the filter-matched packets in the traffic flow are rate-limited to the policer-specified traffic limits. Packets in a conforming flow of filter-matched packets are implicitly set to a **low** PLP. Packets in a nonconforming traffic flow can be discarded, or the packets can be set to a specified forwarding class, set to a specified PLP level, or both, depending on the type of policer and how the policer is configured to handle nonconforming traffic.

NOTE: Before you apply a firewall filter that performs multifield classification and also a policer to the same logical interface and for the same traffic direction, make sure that you consider the order of policer and firewall filter operations.

As an example, consider the following scenario:

- You configure a firewall filter that performs multifield classification (acts on matched packets by setting the forwarding class, the PLP, or both) based on the packet's existing forwarding class or PLP. You apply the firewall filter at the input of a logical interface.
- You also configure a single-rate two-color policer that acts on a red traffic flow by re-marking (setting the forwarding class, the PLP, or both) rather than discarding those packets. You apply the policer as an interface policer at the input of the same logical interface to which you apply the firewall filter.

Because of the order of policer and firewall operations, the input policer is executed before the input firewall filter. This means that the multifield classification specified by the firewall filter is performed on input packets that have already been re-marked once by policing actions. Consequently, any input packet that matches the conditions specified in a firewall filter term is then subject to a second re-marking according to the **forwarding-class** or **loss-priority** nonterminating actions also specified in that term.

Multifield Classification Requirements and Restrictions

This topic covers the following information:

- Supported Platforms on page 1090
- CoS Tricolor Marking Requirement on page 1091
- Restrictions on page 1091

Supported Platforms

The **loss-priority** firewall filter action is supported on the following routing platforms only:

- EX Series switches
- M7i and M10i routers with the Enhanced CFEB (CFEB-E)
- M120 and M320 routers
- MX Series routers
- T Series routers with Enhanced II Flexible PIC Concentrators (FPCs)
- PTX Series routers

CoS Tricolor Marking Requirement

The **loss-priority** firewall filter action has platform-specific requirements dependencies on the CoS tricolor marking feature, as defined in RFC 2698:

- On an M320 router, you cannot commit a configuration that includes the **loss-priority** firewall filter action unless you enable the CoS tricolor marking feature.
- On all routing platforms that support the **loss-priority** firewall filter action, you cannot set the **loss-priority** firewall filter action to **medium-low** or **medium-high** unless you enable the CoS tricolor marking feature. .

To enable the CoS tricolor marking feature, include the **tri-color** statement at the **[edit class-of-service]** hierarchy level.

Restrictions

You cannot configure the **loss-priority** and **three-color-policer** nonterminating actions for the same firewall filter term. These two nonterminating actions are mutually exclusive.

NOTE: On a PTX Series router, you must configure the **policer** action in a separate rule and not combine it with the rule configuring the **forwarding-class**, and **loss-priority** actions. See "Firewall and Policing Differences Between PTX Series Packet Transport Routers and T Series Matrix Routers" on page 584.

Multifield Classification Limitations on M Series Routers

This topic covers the following information:

Problem: Output-Filter Matching on Input-Filter Classification

On M Series routers (except M120 routers), you cannot classify packets with an output filter match based on the ingress classification that is set with an input filter applied to the same IPv4 logical interface.

For example, in the following configuration, the filter called **ingress** assigns all incoming IPv4 packets to the **expedited-forwarding** class. The filter called **egress** counts all packets that were assigned to the **expedited-forwarding** class in the **ingress** filter. This configuration does not work on most M Series routers. It works on all other routing platforms, including M120 routers, MX Series routers, and T Series routers.

```
[edit]
user@host # show firewall
family inet {
  filter ingress {
    term 1 {
      then {
        forwarding-class expedited-forwarding;
```

```
            accept;
          }
        }
        term 2 {
          then accept;
        }
      }
      filter egress {
        term 1 {
          from {
            forwarding-class expedited-forwarding;
          }
          then count ef;
        }
        term 2 {
          then accept;
        }
      }
    }

    [edit]
    user@host# show interfaces
    ge-1/2/0 {
      unit 0 {
        family inet {
          filter {
            input ingress;
            output egress;
          }
        }
      }
    }
```

Workaround: Configure All Actions in the Ingress Filter

As a workaround, you can configure all of the actions in the ingress filter.

```
user@host # show firewall
family inet {
  filter ingress {
    term 1 {
      then {
        forwarding-class expedited-forwarding;
        accept;
        count ef;
      }
    }
    term 2 {
      then accept;
    }
  }
}

[edit]
user@host# show interfaces
ge-1/2/0 {
```



```
    unit 0 {
        family inet {
            filter {
                input ingress;
            }
        }
    }
}
```

Example: Configuring Multifield Classification

This example shows how to configure multifield classification of IPv4 traffic by using firewall filter actions and two firewall filter policers.

Requirements

Before you begin, make sure that your environment supports the features shown in this example:

1. The **loss-priority** firewall filter action must be supported on the router and configurable to all four values.

   a. To be able to set a **loss-priority** firewall filter action, configure this example on logical interface **ge-1/2/0.0** on one of the following routing platforms:

      - MX Series router
      - M120 or M320 router
      - M7i or M10i router with the Enhanced CFEB (CFEB-E)
      - T Series router with Enhanced II Flexible PIC Concentrator (FPC)

   b. To be able to set a **loss-priority** firewall filter action to **medium-low** or **medium-high**, make sure that the CoS tricolor marking feature is enabled. To enable the CoS tricolor marking feature, include the **tri-color** statement at the **[edit class-of-service]** hierarchy level.

2. The **expedited-forwarding** and **assured-forwarding** forwarding classes must be scheduled on the underlying physical interface **ge-1/2/0**.

   a. Make sure that the following forwarding classes are assigned to output queues:

      - **expedited-forwarding**
      - **assured-forwarding**

      Forwarding-class assignments are configured at the **[edit class-of-service forwarding-classes queue *queue-number*]** hierarchy level.

NOTE: You cannot commit a configuration that assigns the same forwarding class to two different queues.

b. Make sure that the output queues to which the forwarding classes are assigned are associated with schedulers. A scheduler defines the amount of interface bandwidth assigned to the queue, the size of the memory buffer allocated for storing packets, the priority of the queue, and the random early detection (RED) drop profiles associated with the queue.

   - You configure output queue schedulers at the **[edit class-of-service schedulers]** hierarchy level.
   - You associate output queue schedulers with forwarding classes by means of a scheduler map that you configure at the **[edit class-of-service scheduler-maps *map-name*]** hierarchy level.

c. Make sure that output-queue scheduling is applied to the physical interface **ge-1/2/0**.

   You apply a scheduler map to a physical interface at the **[edit class-of-service interfaces ge-1/2/0 scheduler-map *map-name*]** hierarchy level.

Overview

In this example, you apply multifield classification to the input IPv4 traffic at a logical interface by using stateless firewall filter actions and two firewall filter policers that are referenced from the firewall filter. Based on the source address field, packets are either set to the **low** loss priority or else policed. Neither of the policers discards nonconforming traffic. Packets in nonconforming flows are marked for a specific forwarding class (**expedited-forwarding** or **assured-forwarding**), set to a specific loss priority, and then transmitted.

NOTE: Single-rate two-color policers always transmit packets in a conforming traffic flow after implicitly setting a **low** loss priority.

Topology

In this example, you apply multifield classification to the IPv4 traffic on logical interface **ge-1/2/0.0**. The classification rules are specified in the IPv4 stateless firewall filter **mfc-filter** and two single-rate two-color policers, **ef-policer** and **af-policer**.

The IPv4 standard stateless firewall filter **mfc-filter** defines three filter terms:

- **isp1-customers**—The first filter term matches packets with the source address 10.1.1.0/24 or 10.1.2.0/24. Matched packets are assigned to the **expedited-forwarding** forwarding class and set to the **low** loss priority.
- **isp2-customers**—The second filter term matches packets with the source address 10.1.3.0/24 or 10.1.4.0/24. Matched packets are passed to **ef-policer**, a policer that

rate-limits traffic to a bandwidth limit of 300 Kbps with a burst-size limit of 50 KB. This policer specifies that packets in a nonconforming flow are marked for the **expedited-forwarding** forwarding class and set to the **high** loss priority.

- **other-customers**—The third and final filter term passes all other packets to **af-policer**, a policer that rate-limits traffic to a bandwidth limit of 300 Kbps and a burst-size limit of 50 KB (the same traffic limits as defined by **ef-policer**). This policer specifies that packets in a nonconforming flow are marked for the **assured-forwarding** forwarding class and set to the **medium-high** loss priority.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring Policers to Rate-Limit Expedited-Forwarding and Assured-Forwarding Traffic on page 1096
- Configuring a Multifield Classification Filter That Also Applies Policing on page 1097
- Applying Multifield Classification Filtering and Policing to the Logical Interface on page 1098

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall policer ef-policer if-exceeding bandwidth-limit 300k
set firewall policer ef-policer if-exceeding burst-size-limit 50k
set firewall policer ef-policer then loss-priority high
set firewall policer ef-policer then forwarding-class expedited-forwarding
set firewall policer af-policer if-exceeding bandwidth-limit 300k
set firewall policer af-policer if-exceeding burst-size-limit 50k
set firewall policer af-policer then loss-priority high
set firewall policer af-policer then forwarding-class assured-forwarding
set firewall family inet filter mfc-filter term isp1-customers from source-address 10.1.1.0/24
set firewall family inet filter mfc-filter term isp1-customers from source-address 10.1.2.0/24
set firewall family inet filter mfc-filter term isp1-customers then loss-priority low
set firewall family inet filter mfc-filter term isp1-customers then forwarding-class
  expedited-forwarding
set firewall family inet filter mfc-filter term isp2-customers from source-address
  10.1.3.0/24
set firewall family inet filter mfc-filter term isp2-customers from source-address
  10.1.4.0/24
set firewall family inet filter mfc-filter term isp2-customers then policer ef-policer
set firewall family inet filter mfc-filter term other-customers then policer af-policer
set interfaces ge-1/2/0 unit 0 family inet address 192.168.1.1/24
set interfaces ge-1/2/0 unit 0 family inet filter input mfc-filter
```

Configuring Policers to Rate-Limit Expedited-Forwarding and Assured-Forwarding Traffic

Step-by-Step Procedure

To configure policers to rate-limit expedited-forwarding and assured-forwarding traffic:

1. Define traffic limits for expedited-forwarding traffic.

```
[edit]
user@host# edit firewall policer ef-policer
[edit firewall policer ef-policer]
user@host# set if-exceeding bandwidth-limit 300k
user@host# set if-exceeding burst-size-limit 50k
user@host# set then loss-priority high
user@host# set then forwarding-class expedited-forwarding
```

2. Configure a policer for assured-forwarding traffic.

```
[edit firewall policer ef-policer]
user@host# up

[edit firewall]
user@host# edit policer af-policer

[edit firewall policer af-policer]
user@host# set if-exceeding bandwidth-limit 300k
user@host# set if-exceeding burst-size-limit 50k
user@host# set then loss-priority high
user@host# set then forwarding-class assured-forwarding
```

Results

Confirm the configuration of the policer by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer af-policer {
  if-exceeding {
    bandwidth-limit 300k;
    burst-size-limit 50k;
  }
  then {
    loss-priority high;
    forwarding-class assured-forwarding;
  }
}
policer ef-policer {
  if-exceeding {
    bandwidth-limit 300k;
    burst-size-limit 50k;
  }
  then {
    loss-priority high;
    forwarding-class expedited-forwarding;
  }
```

```
}
```

Configuring a Multifield Classification Filter That Also Applies Policing

Step-by-Step Procedure

To configure a multifield classification filter that additionally applies policing:

1. Enable configuration of a firewall filter term for IPv4 traffic.

   ```
   [edit]
   user@host# edit firewall family inet filter mfc-filter
   ```

2. Configure the first term to match on source addresses and then classify the matched packets.

   ```
   [edit firewall family inet filter mfc-filter]
   user@host# set term isp1-customers from source-address 10.1.1.0/24
   user@host# set term isp1-customers from source-address 10.1.2.0/24
   user@host# set term isp1-customers then loss-priority low
   user@host# set term isp1-customers then forwarding-class expedited-forwarding
   ```

3. Configure the second term to match on different source addresses and then police the matched packets.

   ```
   [edit firewall family inet filter mfc-filter]
   user@host# set term isp2-customers from source-address 10.1.3.0/24
   user@host# set term isp2-customers from source-address 10.1.4.0/24
   user@host# set term isp2-customers then policer ef-policer
   ```

4. Configure the third term to police all other packets to a different set of traffic limits and actions.

   ```
   [edit firewall family inet filter mfc-filter]
   user@host# set term other-customers then policer af-policer
   ```

Results

Confirm the configuration of the filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter mfc-filter {
    term isp1-customers {
      from {
        source-address 10.1.1.0/24;
        source-address 10.1.2.0/24;
      }
      then {
        loss-priority low;
        forwarding-class expedited-forwarding;
      }
    }
    term isp2-customers {
```



```
            from {
               source-address 10.1.3.0/24;
               source-address 10.1.4.0/24;
            }
            then {
               policer ef-policer;
            }
         }
         term other-customers {
            then {
               policer af-policer;
            }
         }
      }
   }
   policer af-policer {
      if-exceeding {
         bandwidth-limit 300k;
         burst-size-limit 50k;
      }
      then discard;
   }
   policer ef-policer {
      if-exceeding {
         bandwidth-limit 200k;
         burst-size-limit 50k;
      }
      then {
         loss-priority high;
         forwarding-class expedited-forwarding;
      }
   }
```

Applying Multifield Classification Filtering and Policing to the Logical Interface

Step-by-Step Procedure

To apply multifield classification filtering and policing to the logical interface:

1. Enable configuration of IPv4 on the logical interface.

   ```
   [edit]
   user@host# edit interfaces ge-1/2/0 unit 0 family inet
   ```

2. Configure an IP address for the logical interface.

   ```
   [edit interfaces ge-1/2/0 unit 0 family inet ]
   user@host# set address 192.168.1.1/24
   ```

3. Apply the firewall filter to the logical interface input.

   ```
   [edit interfaces ge-1/2/0 unit 0 family inet ]
   user@host# set filter input mfc-filter
   ```


NOTE: Because the policer is executed before the filter, if an input policer is also configured on the logical interface, it cannot use the forwarding class and PLP of a multifield classifier associated with the interface.

Results Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/2/0 {
  unit 0 {
    family inet {
      filter {
        input mfc-filter;
      }
      address 192.168.1.1/24;
    }
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Displaying the Number of Packets Processed by the Policer at the Logical Interface

Purpose Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

Action Use the **show firewall** operational mode command for the filter you applied to the logical interface.

```
user@host> show firewall filter rate-limit-in
Filter: rate-limit-in
Policers:
Name                                            Packets
ef-policer-isp2-customers                         32863
af-policer-other-customers                         3870
```

The command output lists the policers applied by the firewall filter **rate-limit-in**, and the number of packets that matched the filter term.

NOTE: The packet count includes the number of out-of-specification (out-of-spec) packet counts, not all packets policed by the policer.

The policer name is displayed concatenated with the name of the firewall filter term in which the policer is referenced as an action.

Example: Configuring and Applying a Firewall Filter for a Multifield Classifier

This example shows how to configure a firewall filter to classify traffic using a multifield classifier. The classifier detects packets of interest to class of service (CoS) as they arrive on an interface. Multifield classifiers are used when a simple behavior aggregate (BA) classifier is insufficient to classify a packet, when peering routers do not have CoS bits marked, or the peering router's marking is untrusted.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos OS. The example shown here was tested and verified on MX Series routers running Junos OS Release 10.4.

Overview

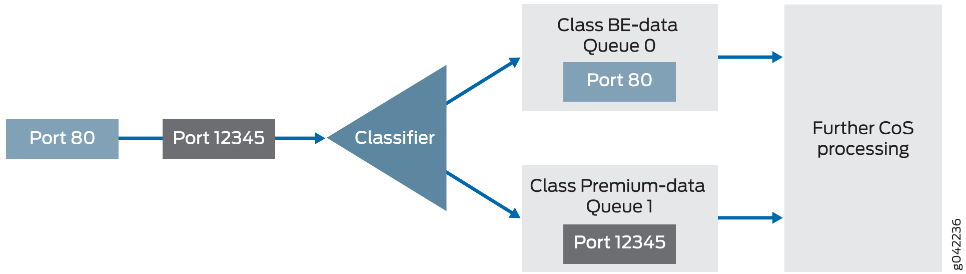
A classifier is a software operation that inspects a packet as it enters the router or switch. The packet header contents are examined, and this examination determines how the packet is treated when the network becomes too busy to handle all of the packets and you want your devices to drop packets intelligently, instead of dropping packets indiscriminately. One common way to detect packets of interest is by source port number. The TCP port numbers 80 and 12345 are used in this example, but many other matching criteria for packet detection are available to multifield classifiers, using firewall filter match conditions. The configuration in this example specifies that TCP packets with source port 80 are classified into the BE-data forwarding class and queue number 0. TCP packets with source port 12345 are classified into the Premium-data forwarding class and queue number 1.

Multifield classifiers are typically used at the network edge as packets enter an autonomous system (AS).

In this example, you configure the firewall filter mf-classifier and specify some custom forwarding classes on Device R1. In specifying the custom forwarding classes, you also associate each class with a queue.

The classifier operation is shown in Figure 73 on page 1101.

Figure 73: Multifield Classifier Based on TCP Source Ports

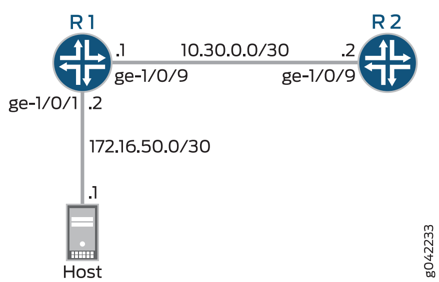


You apply the multifield classifier's firewall filter as an input filter on each customer-facing or host-facing interface that needs the filter. The incoming interface is ge-1/0/0 on Device R1. The classification and queue assignment is verified on the outgoing interface. The outgoing interface is Device R1's ge-1/0/2 interface.

Topology

Figure 74 on page 1101 shows the sample network.

Figure 74: Multifield Classifier Scenario



"CLI Quick Configuration" on page 1101 shows the configuration for all of the Juniper Networks devices in Figure 74 on page 1101.

The section "Step-by-Step Procedure" on page 1102 describes the steps on Device R1.

Classifiers are described in more detail in the following Juniper Networks Learning Byte video.

Video: Class of Service Basics, Part 2: Classification Learning Byte

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from the configuration mode.

Device R1

```
set interfaces ge-1/0/0 description to-host
set interfaces ge-1/0/0 unit 0 family inet filter input mf-classifier
```

```
set interfaces ge-1/0/0 unit 0 family inet address 172.16.50.2/30
set interfaces ge-1/0/2 description to-R2
set interfaces ge-1/0/2 unit 0 family inet address 10.30.0.1/30
set class-of-service forwarding-classes class BE-data queue-num 0
set class-of-service forwarding-classes class Premium-data queue-num 1
set class-of-service forwarding-classes class Voice queue-num 2
set class-of-service forwarding-classes class NC queue-num 3
set firewall family inet filter mf-classifier term BE-data from protocol tcp
set firewall family inet filter mf-classifier term BE-data from port 80
set firewall family inet filter mf-classifier term BE-data then forwarding-class BE-data
set firewall family inet filter mf-classifier term Premium-data from protocol tcp
set firewall family inet filter mf-classifier term Premium-data from port 12345
set firewall family inet filter mf-classifier term Premium-data then forwarding-class
  Premium-data
set firewall family inet filter mf-classifier term accept-all-else then accept
```

Device R2

```
set interfaces ge-1/0/2 description to-R1
set interfaces ge-1/0/2 unit 0 family inet address 10.30.0.2/30
```

Step-by-Step Procedure

The following example requires that you navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode* in the *CLI User Guide*.

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set ge-1/0/0 description to-host
user@R1# set ge-1/0/0 unit 0 family inet address 172.16.50.2/30

user@R1# set ge-1/0/2 description to-R2
user@R1# set ge-1/0/2 unit 0 family inet address 10.30.0.1/30
```

2. Configure the custom forwarding classes and associated queue numbers.

```
[edit class-of-service forwarding-classes]
user@R1# set BE-data queue-num 0
user@R1# set Premium-data queue-num 1
user@R1# set Voice queue-num 2
user@R1# set NC queue-num 3
```

3. Configure the firewall filter term that places TCP traffic with a source port of 80 (HTTP traffic) into the BE-data forwarding class, associated with queue 0.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term BE-data from protocol tcp
user@R1# set term BE-data from port 80
user@R1# set term BE-data then forwarding-class BE-data
```

4. Configure the firewall filter term that places TCP traffic with a source port of 12345 into the Premium-data forwarding class, associated with queue 1.


```
[edit firewall family inet filter mf-classifier]
user@R1# set term Premium-data from protocol tcp
user@R1# set term Premium-data from port 12345
user@R1# set term Premium-data then forwarding-class Premium-data
```

5. At the end of your firewall filter, configure a default term that accepts all other traffic.

   Otherwise, all traffic that arrives on the interface and is not explicitly accepted by the firewall filter is discarded.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term accept-all-else then accept
```

6. Apply the firewall filter to the ge-1/0/0 interface as an input filter.

```
[edit interfaces]
user@R1# set ge-1/0/0 unit 0 family inet filter input mf-classifier
```

Results From configuration mode, confirm your configuration by entering the **show interfaces**, **show class-of-service**, **show firewall** commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
ge-1/0/0 {
    description to-host;
    unit 0 {
        family inet {
            filter {
                input mf-classifier;
            }
            address 172.16.50.2/30;
        }
    }
}
ge-1/0/2 {
    description to-R2;
    unit 0 {
        family inet {
            address 10.30.0.1/30;
        }
    }
}
```

```
user@R1# show class-of-service
forwarding-classes {
    class BE-data queue-num 0;
    class Premium-data queue-num 1;
    class Voice queue-num 2;
    class NC queue-num 3;
}
```

```
user@R1# show firewall
family inet {
```

```
    filter mf-classifier {
        term BE-data {
            from {
                protocol tcp;
                port 80;
            }
            then forwarding-class BE-data;
        }
        term Premium-data {
            from {
                protocol tcp;
                port 12345;
            }
            then forwarding-class Premium-data;
        }
        term accept-all-else {
            then accept;
        }
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Checking the CoS Settings on page 1104
- Sending TCP Traffic into the Network and Monitoring the Queue Placement on page 1105

Checking the CoS Settings

Purpose Confirm that the forwarding classes are configured correctly.

Action From Device R1, run the **show class-of-service forwardng-classes** command.

```
user@R1> show class-of-service forwarding-class

Forwarding class                        ID      Queue  Restricted queue  Fabric
priority  Policing priority   SPU priority
  BE-data                                0       0          0               low
            normal             low
  Premium-data                           1       1          1               low
            normal             low
  Voice                                  2       2          2               low
            normal             low
  NC                                     3       3          3               low
            normal             low
```

Meaning The output shows the configured custom classifier settings.

Sending TCP Traffic into the Network and Monitoring the Queue Placement

Purpose Make sure that the traffic of interest is sent out the expected queue.

Action

1. Clear the interface statistics on Device R1's outgoing interface.

   `user@R1>` **clear interfaces statistics ge-1/0/2**

2. Use a traffic generator to send 50 TCP port 80 packets to Device R2 or to some other downstream device.
3. On Device R1, check the queue counters.

   Notice that you check the queue counters on the downstream output interface, not on the incoming interface.

   `user@R1>` **show interfaces extensive ge-1/0/2 | find "Queue counters"**

```
  Queue counters:      Queued packets  Transmitted packets     Dropped packets

    0                               50                   50
  0
    1                                0                   57
  0
    2                                0                    0
  0
    3                                0                    0
  0
```

4. Use a traffic generator to send 50 TCP port 12345 packets to Device R2 or to some other downstream device.

   `[root@host]#` **hping 172.16.60.1 -c 50 -s 12345 -k**

5. On Device R1, check the queue counters.

   `user@R1>` **show interfaces extensive ge-1/0/2 | find "Queue counters"**

```
  Queue counters:      Queued packets  Transmitted packets     Dropped packets

    0                               50                   50
  0
    1                               50                   57
  0
    2                                0                    0
  0
    3                                0                    0
  0
```

Meaning The output shows that the packets are classified correctly. When port 80 is used in the TCP packets, queue 0 is incremented. When port 12345 is used, queue 1 is incremented.

Related Documentation

- Firewall Filter Nonterminating Actions on page 673
- Order of Policer and Firewall Filter Operations on page 979
- Two-Color Policer Configuration Overview on page 1027
- Guidelines for Applying Traffic Policers on page 983

- *The Junos OS CoS Components Used to Manage Congestion and Control Service Levels*
- *Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic*
- *Understanding How Forwarding Classes Assign Classes to Output Queues*
- *Default Forwarding Classes*
- *Managing Congestion Using RED Drop Profiles and Packet Loss Priorities*
- *tri-color* statement

Policer Overhead to Account for Rate Shaping in the Traffic Manager

- Policer Overhead to Account for Rate Shaping Overview on page 1106
- Example: Configuring Policer Overhead to Account for Rate Shaping on page 1106

Policer Overhead to Account for Rate Shaping Overview

If you configure ingress or egress traffic-shaping overhead values for an interface, the traffic manager cannot apply these values to any rate-limiting also applied to the interface. To enable the router to account for the additional Ethernet frame length when policing actions are being determined, you must configure the ingress or egress overhead values for policers separately.

NOTE: When a policer overhead value is changed, the PIC or DPC goes offline and then comes back online.

For Gigabit Ethernet Intelligent Queuing 2 (IQ2) and Enhanced IQ2 (IQ2E) PICs or interfaces on Dense Port Concentrators (DPCs) in MX Series routers, you can control the rate of traffic that passes through all interfaces on the PIC or DPC by configuring a *policer overhead*. You can configure a policer ingress overhead and a policer egress overhead, each with values from 0 through 255 bytes. The policer overhead values are added to the length of the final Ethernet frame when determining ingress and egress policer actions.

Example: Configuring Policer Overhead to Account for Rate Shaping

This example shows how to configure overhead values for policers when rate-shaping overhead is configured.

- Requirements on page 1107
- Overview on page 1107

Requirements

Before you begin, make sure that interface for which you are applying ingress or egress policer overhead is hosted on one of the following:

- Gigabit Ethernet IQ2 PIC
- IQ2E PIC
- DPCs in MX Series routers

Overview

This example shows how to configure policer overhead values for all physical interfaces on a supported PIC or MPC so that the rate shaping value configured on a logical interface is accounted for in any policing on that logical interface.

Topology

The router hosts a Gigabit Ethernet IQ2 PIC, installed in PIC location 3 of the Flexible PIC Concentrator (FPC) in slot number 1. The physical interface on port 1 on that PIC is configured to receive traffic on logical interface 0 and send it back out on logical interface 1. Class-of-service scheduling includes 100 Mbps of traffic rate-shaping overhead for the output traffic. A policer egress overhead of 100 bytes is configured on the entire PIC so that, for any policers applied to the output traffic, 100 bytes are added to the final Ethernet frame length when determining ingress and egress policer actions.

NOTE:

Traffic rate-shaping and corresponding policer overhead are configured separately:

- You configure rate shaping at the **[edit class-of-service interfaces *interface-name* unit *unit-number*]** hierarchy level.
- You configure policer overhead at the **[edit chassis fpc *slot-number* pic *pic-number*]** hierarchy level.

When a policer overhead value is changed, the PIC or DPC goes offline and then comes back online.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring the Logical Interfaces on page 1108
- Configuring Traffic Rate-Shaping on the Logical Interface That Carries Output Traffic on page 1110
- Configuring Policer Overhead on the PIC or DPC That Hosts the Rate-Shaped Logical Interface on page 1111
- Applying a Policer to the Logical Interface That Carries Input Traffic on page 1112

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces ge-1/3/1 per-unit-scheduler
set interfaces ge-1/3/1 vlan-tagging
set interfaces ge-1/3/1 unit 0 vlan-id 100
set interfaces ge-1/3/1 unit 0 family inet address 10.10.10.1/30
set interfaces ge-1/3/1 unit 1 vlan-id 101
set interfaces ge-1/3/1 unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
  00:00:11:22:33:44
set class-of-service schedulers be transmit-rate percent 5
set class-of-service schedulers ef transmit-rate percent 30
set class-of-service schedulers af transmit-rate percent 30
set class-of-service schedulers nc transmit-rate percent 35
set class-of-service scheduler-maps my-map forwarding-class best-effort scheduler be
set class-of-service scheduler-maps my-map forwarding-class expedited-forwarding
  scheduler ef
set class-of-service scheduler-maps my-map forwarding-class network-control scheduler
  nc
set class-of-service scheduler-maps my-map forwarding-class assured-forwarding
  scheduler af
set class-of-service interfaces ge-1/3/1 unit 1 scheduler-map my-map
set class-of-service interfaces ge-1/3/1 unit 1 shaping-rate 100m
set firewall policer 500Kbps logical-interface-policer
set firewall policer 500Kbps if-exceeding bandwidth-limit 500k
set firewall policer 500Kbps if-exceeding burst-size-limit 625k
set firewall policer 500Kbps then discard
set chassis fpc 1 pic 3 ingress-policer-overhead 100
set chassis fpc 1 pic 3 egress-policer-overhead 100
set interfaces ge-1/3/1 unit 0 family inet policer input 500Kbps
```

Configuring the Logical Interfaces

Step-by-Step Procedure

To configure the logical interfaces:

1. Enable configuration of the interface

```
[edit]
user@host# edit interfaces ge-1/3/1
```


2. Enable multiple queues for each logical interface (so that you can associate an output scheduler with each logical interface).

   ```
   [edit interfaces ge-1/3/1]
   user@host# set per-unit scheduler
   user@host# set vlan-tagging
   ```

   NOTE: For Gigabit Ethernet IQ2 PICs only, use the **shared-scheduler** statement to enable shared schedulers and shapers on a physical interface.

3. Configure logical interface **ge-1/3/1.0**.

   ```
   [edit interfaces ge-1/3/1]
   user@host# set unit 0 vlan-id 100
   user@host# set unit 0 family inet address 10.10.10.1/30
   ```

4. Configure logical interface **ge-1/3/1.1**.

   ```
   [edit interfaces ge-1/3/1]
   user@host# set unit 1 vlan-id 101
   user@host# set unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
     00:00:11:22:33:44
   ```

Results Confirm the configuration of the interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/1 {
  per-unit-scheduler;
  vlan-tagging;
  unit 0 {
    vlan-id 100;
    family inet {
      address 10.10.10.1/30;
    }
  }
  unit 1 {
    vlan-id 101;
    family inet {
      address 20.20.20.1/30 {
        arp 20.20.20.2 mac 00:00:11:22:33:44;
      }
    }
  }
}
```

Configuring Traffic Rate-Shaping on the Logical Interface That Carries Output Traffic

Step-by-Step Procedure

To configure traffic rate-shaping on the logical interface that carries output traffic:

1. Enable configuration of class-of-service features.

   ```
   [edit]
   user@host# edit class-of-service
   ```

2. Configure packet scheduling on logical interface **ge-1/3/1.0**.

   a. Configure schedulers that specify the percentage of transmission capacity.

   ```
   [edit class-of-service]
   user@host# edit schedulers

   [edit class-of-service schedulers]
   user@host# set be transmit-rate percent 5
   user@host# set ef transmit-rate percent 30
   user@host# set af transmit-rate percent 30
   user@host# set nc transmit-rate percent 35
   ```

   A percentage of zero drops all packets in the queue. When the **rate-limit** option is specified, the transmission rate is limited to the rate-controlled amount. In contrast with the **exact** option, a scheduler with the **rate-limit** option shares unused bandwidth above the rate-controlled amount.

   b. Configure a scheduler map to associate each scheduler with a forwarding class.

   ```
   [edit class-of-service]
   user@host# edit scheduler-maps my-map

   [edit class-of-service scheduler-maps my-map]
   user@host# set forwarding-class best-effort scheduler be
   user@host# set forwarding-class expedited-forwarding scheduler ef
   user@host# set forwarding-class network-control scheduler nc
   user@host# set forwarding-class assured-forwarding scheduler af
   ```

   c. Associate the scheduler map with logical interface **ge-1/3/1.0**.

   ```
   [edit class-of-service]
   user@host# edit interfaces ge-1/3/1 unit 1

   [edit class-of-service interfaces ge-1/3/1 unit 1]
   user@host# set scheduler-map my-map
   ```

3. Configure 100 Mbps of traffic rate-shaping overhead on logical interface **ge-1/3/1.1**.

   ```
   [edit class-of-service interfaces ge-1/3/1 unit 1]
   user@host# set shaping-rate 100
   ```

   Alternatively, you can configure a shaping rate for a logical interface and oversubscribe the physical interface by including the **shaping-rate** statement at the **[edit class-of-service traffic-control-profiles]** hierarchy level. With this configuration approach, you can independently control the delay-buffer rate.

Results

Confirm the configuration of the class-of-service features (including the 100 Mbp of shaping of the egress traffic) by entering the **show class-of-service** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show class-of-service
interfaces {
  ge-1/3/1 {
    unit 1 {
      scheduler-map my-map;
      shaping-rate 100m;
    }
  }
}
scheduler-maps {
  my-map {
    forwarding-class best-effort scheduler be;
    forwarding-class expedited-forwarding scheduler ef;
    forwarding-class network-control scheduler nc;
    forwarding-class assured-forwarding scheduler af;
  }
}
schedulers {
  be {
    transmit-rate percent 5;
  }
  ef {
    transmit-rate percent 30;
  }
  af {
    transmit-rate percent 30;
  }
  nc {
    transmit-rate percent 35;
  }
}
```

Configuring Policer Overhead on the PIC or DPC That Hosts the Rate-Shaped Logical Interface

Step-by-Step Procedure

To configure policer overhead on the PIC or MPC that hosts the rate-shaped logical interface:

1. Enable configuration of the supported PIC or MPC.

```
[edit]
user@host# set chassis fpc 1 pic 3
```

2. Configure 100 bytes of policer overhead on the supported PIC or MPC.

```
[edit chassis fpc 1 pic 3]
user@host# set ingress-policer-overhead 100
user@host# set egress-policer-overhead 100
```

NOTE: These values are added to the length of the final Ethernet frame when determining ingress and egress policer actions for all physical interfaces on the PIC or MPC.

You can specify policer overhead with values from 0 through 255 bytes.

Results

Confirm the configuration of the policer overhead on the physical interface to account for rate-shaping by entering the **show chassis** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show chassis
chassis {
  fpc 1 {
    pic 3 {
      egress-policer-overhead 100;
      ingress-policer-overhead 100;
    }
  }
}
```

Applying a Policer to the Logical Interface That Carries Input Traffic

Step-by-Step Procedure

To apply a policer to the logical interface that carries input traffic:

1. Configure the logical interface (aggregate) policer.

```
[edit]
user@host# edit firewall policer 500Kbps

[edit firewall policer 500Kbps]
user@host# set logical-interface-policer
user@host# set if-exceeding bandwidth-limit 500k
user@host# set if-exceeding burst-size-limit 625k
user@host# set then discard
```

2. Apply the policer to Layer 3 input on the IPv4 logical interface.

```
[edit]
user@host# set interfaces ge-1/3/1 unit 0 family inet policer input 500Kbps
```

NOTE: The 100 Mbps policer overhead is added to the length of the final Ethernet frame when determining ingress and egress policer actions,

Results Confirm the configuration of the policer with rate-shaping overhead by entering the **show firewall** and **show interfaces** configuration mode commands. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer 500Kbps {
    logical-interface-policer;
    if-exceeding {
        bandwidth-limit 500k;
        burst-size-limit 625k;
    }
    then discard;
}
[edit]
user@host# show interfaces
ge-1/3/1 {
    per-unit-scheduler;
    vlan-tagging;
    unit 0 {
        vlan-id 100;
        layer2-policer {
            input-policer 500Kbps;
        }
        family inet {
            address 10.10.10.1/30;
        }
    }
    unit 0 {
        vlan-id 101;
        family inet {
            address 20.20.20.1/30 {
                arp 20.20.20.2 mac 00:00:11:22:33:44;
            }
        }
    }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying Traffic Statistics and Policers for the Logical Interface on page 1113
- Displaying Statistics for the Policer on page 1114

Displaying Traffic Statistics and Policers for the Logical Interface

Purpose Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

Action Use the **show interfaces** operational mode command for logical interface **ge-1/3/1.0**, and include the **detail** or **extensive** option. The command output section for **Traffic statistics** lists the number of bytes and packets received and transmitted on the logical interface, and the **Protocol inet** section contains a **Policer** field that would list the policer **500Kbps** as an input or output policer as follows:

- **Input: 500Kbps-ge-1/3/1.0-log_int-i**
- **Output: 500Kbps-ge-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to Input traffic only.

Displaying Statistics for the Policer

Purpose Verify the number of packets evaluated by the policer.

Action Use the **show policer** operational mode command and optionally specify the name of the policer. The command output displays the number of packets evaluated by each configured policer (or the specified policer), in each direction. For the policer **500Kbps**, the input and output policer names are displayed as follows:

- **500Kbps-ge-1/3/1.0-log_int-i**
- **500Kbps-ge-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to input traffic only.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Guidelines for Applying Traffic Policers on page 983
- "*Configuring a Policer Overhead*" in the CLI Explorer

CHAPTER 31

Configuring Three-Color Traffic Policers at Layer 3

Three-Color Policer Configuration Overview

Table 71 on page 1115 describes the hierarchy levels at which you can configure and apply single-rate tricolor-marking (single-rate TCM) policers and two-rate tricolor-marking (two-rate TCM) policers to Layer 3 traffic. For information about applying three-color policers to Layer 2 traffic, see "Three-Color Policing at Layer 2 Overview" on page 1018.

Table 71: Three-Color Policer Configuration and Application Overview

Policer Configuration	Layer 3 Application	Key Points
Single-Rate Three-Color Policer Defines traffic rate limiting that you can apply to Layer 3 protocol-specific traffic at a logical interface. Can be applied as a firewall filter policer only. Provides moderate allowances for short periods of traffic that exceed the committed burst size.		

Basic single-rate TCM policer configuration:
 	
[edit firewall]
three-color-policer *policer-name* {
  single-rate {
    (color-aware | color-blind);
    committed-information-rate
      bps;
    committed-burst-size *bytes*;
    excess-burst-size *bytes*;
  }
	Reference the policer from a firewall filter, and apply the filter to a protocol family on a logical interface:
	
	[edit firewall]
	family *family-name* {
	  filter *filter-name* {
	    term *term-name* {
	      from {
	        ... *match-conditions* ...
	      }
	      then {
		Policer configuration:
		
		• Include the **single-rate (color-aware | color-blind)** statement.
		
		Firewall filter configuration:
		
		• Include the **three-color-policer single-rate** ***policer-name*** action.

Table 71: Three-Color Policer Configuration and Application Overview *(continued)*

<table>
<tr><th>Policer Configuration</th><th>Layer 3 Application</th><th>Key Points</th></tr>
<tr><td><pre>
    action {
      loss-priority high then discard;
    }
  }
</pre></td><td><pre>
          three-color-policer {
            single-rate policer-name;
          }
        }
      }
    }
  }
</pre>Apply the filter to a logical interface at the protocol family level:<pre>
  [edit interfaces]
  interface-name {
    unit unit-number {
      family family-name {
        filter {
          input filter-name;
          output filter-name;
        }
      }
    }
  }
</pre></td><td>Applying the firewall filter to the logical interface: Include the filter (input | output) filter-name statement.</td></tr>
</table>

Table 71: Three-Color Policer Configuration and Application Overview *(continued)*

Policer Configuration	Layer 3 Application	Key Points
Single-Rate Three-Color Physical Interface Policer Defines traffic rate limiting that applies to all logical interfaces and protocol families configured on a physical interface, even if the interfaces belong to different routing instances. Can be applied as a firewall filter policer only.		

Physical interface single-rate TCM policer:
 	
[edit firewall]
three-color-policer *policer-name* {
  physical-interface-policer;
  single-rate {
    (color-aware | color-blind);
    committed-information-rate *bps*;
    committed-burst-size *bytes*;
    excess-burst-size *bytes*;
  }
  action {
    loss-priority high then discard;
  }
}
	Reference the policer from a physical interface filter only, and apply the filter to a protocol family on a logical interface:
	
	[edit firewall]
	family *family-name* {
	  filter *filter-name* {
	    physical-interface-filter
	    term *term-name* {
	      from {
	        ... *match-conditions* ...
	      }
	      then {
	        three-color-policer {
	          single-rate *policer-name*;
	        }
	      }
	    }
	  }
	}
	
	[edit interfaces]
	interface-name {
	  unit *number* {
	    family *family-name* {
	      filter {
	        input *filter-name*;
	        output *filter-name*;
	      }
	    }
	  }
	}
		Policer configuration:
		• Include the **physical-interface-policer** statement.
		
		Firewall filter configuration:
		• Include the **physical-interface-filter** statement.
		
		Application:
		• Include the **filter (input | output)** ***filter-name*** statement.
		
		Verification
		• To verify, use the **show firewall filter** ***filter-name*** operational mode command.

Table 71: Three-Color Policer Configuration and Application Overview *(continued)*

Policer Configuration	Layer 3 Application	Key Points
Basic Two-Rate Three-Color Policer Defines traffic rate limiting that you can apply to Layer 3 protocol-specific traffic at a logical interface. Can be applied as a firewall filter policer only. Provides moderate allowances for sustained periods of traffic that exceed the committed bandwidth limit or burst size.		

Basic two-rate TCM policer configuration:
 	
[edit firewall]
three-color-policer *policer-name* {
  two-rate {
    (color-aware | color-blind);
    committed-information-rate *bps*;
    committed-burst-size *bytes*;
    peak-information-rate *bps*;
    peak-burst-size *bytes*;
  }
  action {
    loss-priority high then discard;
  }
}
	Reference the policer from a firewall filter, and apply the filter to a protocol family on a logical interface:
	
	[edit firewall]
	family *family-name* {
	  filter *filter-name* {
	    term *term-name* {
	      from {
	        ... *match-conditions* ...
	      }
	      then {
	        three-color-policer {
	          two-rate *policer-name*;
	        }
	      }
	    }
	  }
	}
	
	[edit interfaces]
	interface-name {
	  unit *unit-number* {
	    family *family-name* {
	      filter {
	        input *filter-name*;
	        output *filter-name*;
	      }
	    }
	  }
	}
		Policer configuration:
		
		• Include the **two-rate (color-aware | color-blind)** statement.
		
		Firewall filter configuration:
		
		• Include the **three-color-policer two-rate *policer-name*** action.
		
		Applying the firewall filter to the logical interface:
		
		• Include the **filter (input | output) *filter-name*** statement.

Related Documentation

- Three-Color Policer Configuration Guidelines on page 1118
- Basic Single-Rate Three-Color Policers on page 1121
- Basic Two-Rate Three-Color Policers on page 1127
- Two-Color and Three-Color Logical Interface Policers on page 1141
- Two-Color and Three-Color Physical Interface Policers on page 1154

Three-Color Policer Configuration Guidelines

Platforms Supported for Three-Color Policers

Three-color policers are supported on the following Juniper Networks routers:

- M120 Multiservice Edge Routers
- M320 Multiservice Edge Routers and T Series Core Routers with Enhanced II Flexible PIC Concentrators (FPCs)
- MX Series 3D Universal Edge Routers
- T640 Core Routers with Enhanced Scaling FPC4
- T4000 Core Routers with FPC5

On MX Series and M120 routers, you can apply three-color policers to aggregated interfaces.

The **discard** action for a tricolor marking policer for a firewall filter is supported on the M120 routers, M320 routers with Enhanced-III FPCs, M7i and M10i routers with the Enhanced CFEB (CFEB-E), and MX Series routers with Trio MPCs, so it is not necessary to include the **logical-interface-policer** statement for them.

Color Modes for Three-Color Policers

Three-color policers—both single-rate and two-rate three-color policer schemes—can operate in either of two modes:

Color-Blind Mode

In *color-blind* mode, the three-color policer assumes that all packets examined have not been previously marked or metered. If you configure a three-color policer to be color-blind instead of color-aware, the policer ignores preexisting color markings that might have been set for a packet by another traffic policer configured at a previous network node.

Color-Aware Mode

In *color-aware* mode, the three-color policer assumes that all packets examined have been previously marked or metered. In other words, the three-color policer takes into account any coloring markings that might have been set for a packet by another traffic policer configured at a previous network node. At the node where color-aware policing is configured, any preexisting color markings are used in determining the appropriate policing action for the packet.

In color-aware mode, the three-color policer can increase the packet loss priority (PLP) level of a packet, but never decrease it. For example, if a color-aware three-color policer meters a packet with a medium PLP marking, it can raise the PLP level to high, but cannot reduce the PLP level to low.

For two-rate, three-color policing, the Junos OS uses two token buckets to manage bandwidth based on the two rates of traffic. For example, two-rate policing might be configured on a node upstream in the network. The two-rate policer has marked a packet as yellow (loss priority medium-low). The color-aware policer takes this yellow marking into account when determining the appropriate policing action. In color-aware policing, the yellow packet would never receive the action associated with either the green packets or red packets. This way, tokens for violating packets are never taken from the metering token buckets at the color-aware policing node.

NOTE: For a three-color policer operating in color-aware mode and when the PLP of the input packet is medium-low, the color of the input packet to the policer is mapped to the color yellow.

In such a scenario, if the color of the input packet remains unchanged, the policer operates in the following way:

- On a T1600 Enhanced Scaling Type 4 FPC (T1600-FPC4-ES), the PLP of the output packet remains medium-low.
- On a T4000 Type 5 FPC (T4000-FPC5-3D), the PLP of the output packet is marked as medium-high.

Because of this difference, for any applications (such as rewrite and WRED selection on egress interface) that use PLP, the packets are treated differently for the same flow depending on the FPC type (T1600 Enhanced Scaling FPC4 (T1600-FPC4-ES) or T4000 FPC5 (T4000-FPC5-3D)) on which the policer is applied.

Naming Conventions for Three-Color Policers

Because policers can be numerous and must be applied correctly to work, a simple naming convention makes it easier to apply the policers properly.

We recommend that you name your policer using a convention that identifies the basic components of the policer:

- Three-color policer type—Where **srTCM** identifies a *single-rate* three-color policer and **trTCM** identifies a *two-rate* three-color policer.
- Three-color policer color mode—Where **ca** identifies a *color-aware* three-color policer and **cb** identifies a *color-blind three-color policer*.

NOTE:

TCM stands for tricolor marking.

Table 72 on page 1121 describes a recommended naming convention for policers.

Table 72: Recommended Naming Convention for Policers

Three-Color Policer Type	Naming Convention	Example Names
Single-rate three-color, color-aware	srTCM*number*-ca	srTCM1-ca, srTCM2-ca, srTCM3-ca, ...
Single-rate three-color, color-blind	srTCM*number*-cb	srTCM1-cb, srTCM2-cb, srTCM3-cb, ...
Two-rate three-color, color-aware	trTCM*number*-ca	trTCM1-ca, trTCM2-ca, trTCM3-ca, ...
Two-rate three-color, color-blind	trTCM*number*-cb	trTCM1-cb, trTCM2-cb, trTCM3-cb, ...

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Guidelines for Applying Traffic Policers on page 983

Basic Single-Rate Three-Color Policers

- Single-Rate Three-Color Policer Overview on page 1121
- Example: Configuring a Single-Rate Three-Color Policer on page 1122

Single-Rate Three-Color Policer Overview

A single-rate three-color policer defines a bandwidth limit and a maximum burst size for guaranteed traffic and a second burst size for peak traffic. A single-rate three-color policer is most useful when a service is structured according to packet length and not peak arrival rate.

Single-rate three-color policing meters a traffic stream based on the following configured traffic criteria:

- Committed information rate (CIR)—Bandwidth limit for guaranteed traffic.
- Committed burst size (CBS)—Maximum packet size permitted for bursts of data that exceed the CIR.
- Excess burst size (EBS)—Maximum packet size permitted for peak traffic.

Single-rate tricolor marking (single-rate TCM) classifies traffic as belonging to one of three color categories and performs congestion-control actions on the packets based on the color marking:

- Green—Traffic that conforms to *either* the bandwidth limit *or* the burst size for guaranteed traffic (CIR or CBS). For a green traffic flow, single-rate marks the packets with an implicit loss priority of **low** and transmits the packets.
- Yellow—Traffic that exceeds *both* the bandwidth limit *and* the burst size for guaranteed traffic (CIR and CBS) but not the burst size for peak traffic (EBS). For a yellow traffic flow, single-rate marks the packets with an implicit loss priority of **medium-high** and transmits the packets.
- Red—Traffic that exceeds the burst size for peak traffic (EBS), single-rate marks packets with an implicit loss priority of **high** and, optionally, discards the packets.

If congestion occurs downstream, the packets with higher loss priority are more likely to be discarded.

NOTE: For both single-rate and two-rate three-color policers, the only *configurable* action is to discard packets in a red traffic flow.

The **discard** action for a tricolor marking policer for a firewall filter is supported on the M120 routers, M320 routers with Enhanced-III FPCs, M7i and M10i routers with the Enhanced CFEB (CFEB-E), and MX Series routers with MPCs, so it is not necessary to include the **logical-interface-policer** statement for them.

Example: Configuring a Single-Rate Three-Color Policer

This example shows how to configure a single-rate three-color policer.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

A single-rate three-color policer meters a traffic flow against a bandwidth limit and burst-size limit for guaranteed traffic, plus a second burst-size limit for excess traffic. Traffic that conforms to the limits for guaranteed traffic is categorized as green, and nonconforming traffic falls into one of two categories:

- Nonconforming traffic that does not exceed the burst size for excess traffic is categorized as yellow.

- Nonconforming traffic that exceeds the burst size for excess traffic is categorized as red.

Each category is associated with an action. For green traffic, packets are implicitly set with a loss-priority value of **low** and then transmitted. For yellow traffic, packets are implicitly set with a loss-priority value of **medium-high** and then transmitted. For red traffic, packets are implicitly set with a loss-priority value of **high** and then transmitted. If the policer configuration includes the optional action statement (**action** loss-priority high then discard), then packets in a red flow are discarded instead.

You can apply a three-color policer to Layer 3 traffic as a firewall filter policer only. You reference the policer from a stateless firewall filter term, and then you apply the filter to the input or output of a logical interface at the protocol level.

Topology

In this example, you apply a color-aware, single-rate three-color policer to the input IPv4 traffic at logical interface **ge-2/0/5.0**. The IPv4 firewall filter term that references the policer does not apply any packet-filtering. The filter is used only to apply the three-color policer to the interface.

You configure the policer to rate-limit traffic to a bandwidth limit of 40 Mbps and a burst-size limit of 100 KB for green traffic but also allow an excess burst-size limit of 200 KB for yellow traffic. Only nonconforming traffic that exceeds the peak burst-size limit is categorized as red. In this example, you configure the three-color policer action **loss-priority high then discard**, which overrides the implicit marking of red traffic to a **high** loss priority.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set firewall three-color-policer srTCM1-ca single-rate color-aware
set firewall three-color-policer srTCM1-ca single-rate committed-information-rate 40m
set firewall three-color-policer srTCM1-ca single-rate committed-burst-size 100k
set firewall three-color-policer srTCM1-ca single-rate excess-burst-size 200k
set firewall three-color-policer srTCM1-ca action loss-priority high then discard
set firewall family inet filter filter-srtcm1ca-all term 1 then three-color-policer single-rate
  srTCM1-ca
set class-of-service interfaces ge-2/0/5 unit 0 forwarding-class af
set interfaces ge-2/0/5 unit 0 family inet address 10.20.130.1/24
```



```
set interfaces ge-2/0/5 unit 0 family inet filter input filter-srtcm1ca-all
```

Configuring a Single-Rate Three-Color Policer

Step-by-Step Procedure

To configure a single-rate three-color policer:

1. Enable configuration of a three-color policer.

```
[edit]
user@host# edit firewall three-color-policer srTCM1-ca
```

2. Configure the color mode of the single-rate three-color policer.

```
[edit firewall three-color-policer srTCM1-ca]
user@host# set single-rate color-aware
```

3. Configure the single-rate guaranteed traffic limits.

```
[edit firewall three-color-policer srTCM1-ca]
user@host# set single-rate committed-information-rate 40m
user@host# set single-rate committed-burst-size 100k
```

4. Configure the single-rate burst-size limit that is used to classify nonconforming traffic.

```
[edit firewall three-color-policer srTCM1-ca]
user@host# set single-rate excess-burst-size 200k
```

5. (Optional) Configure the action for nonconforming traffic.

```
[edit firewall three-color-policer srTCM1-ca]
user@host# set action loss-priority high then discard
```

   For three-color policers, the only configurable action is to discard packets in a red traffic flow. In this example, packets in a red traffic flow have been implicitly marked with a **high** packet loss priority (PLP) level because the traffic flow exceeded the rate-limiting defined by the single rate-limit (specified by the **committed-information-rate 40m** statement) and the larger burst-size limit (specified by the **excess-burst-size 200k** statement). Because the optional **action** statement is included, this example takes the more severe action of discarding packets in a red traffic flow.

Results

Confirm the configuration of the hierarchical policer by entering the **show firewall** configuration command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
three-color-policer srTCM1-ca {
  action {
    loss-priority high then discard;
  }
  single-rate {
    color-aware;
```



```
        committed-information-rate 40m;
        committed-burst-size 100k;
        excess-burst-size 200k;
    }
}
```

Configuring an IPv4 Stateless Firewall Filter That References the Policer

Step-by-Step Procedure

To configure a standard stateless firewall filter that references the policer:

1. Enable configuration of an IPv4 standard stateless firewall filter.

```
[edit]
user@host# edit firewall family inet filter filter-srtcm1ca-all
```

2. Specify the filter term that references the policer.

```
[edit firewall family inet filter filter-srtcm1ca-all]
user@host# set term 1 then three-color-policer single-rate srTCM1-ca
```

   Note that the term does not specify any match conditions. The firewall filter passes all packets to the policer.

Results

Confirm the configuration of the firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
    filter filter-srtcm1ca-all {
        term 1 {
            then {
                three-color-policer {
                    single-rate srTCM1-ca;
                }
            }
        }
    }
}
three-color-policer srTCM1-ca {
    action {
        loss-priority high then discard;
    }
    single-rate {
        color-aware;
        committed-information-rate 40m;
        committed-burst-size 100k;
        excess-burst-size 200k;
    }
}
```

Applying the Filter to the Logical Interface

Step-by-Step Procedure

To apply the filter to the logical interface:

1. (MX Series routers only) (Optional) Reclassify all incoming packets on the logical interface **ge-2/0/5.0** to assured forwarding, regardless of any preexisting classification.

   ```
   [edit]
   user@host# set class-of-service interfaces ge-2/0/5 unit 0 forwarding-class af
   ```

   The classifier name can be a configured classifier or one of the default classifiers.

2. Enable configuration of the logical interface.

   ```
   [edit]
   user@host# edit interfaces ge-2/0/5 unit 0 family inet
   ```

3. Configure an IP address.

   ```
   [edit interfaces ge-2/0/5 unit 0 family inet]
   user@host# set address 10.20.130.1/24
   ```

4. Reference the filter as an input filter.

   ```
   [edit interfaces ge-2/0/5 unit 0 family inet]
   user@host# set filter input filter-srtcm1ca-all
   ```

Results

Confirm the configuration of the interface by entering the **show class-of-service** and **show interfaces** configuration mode commands. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show class-of-service
interfaces {
  ge-2/0/5 {
    unit 0 {
      forwarding-class af;
    }
  }
}
[edit]
user@host# show interfaces
ge-2/0/5 {
  unit 0 {
    family inet {
      filter {
        input filter-srtcm1ca-all;
        }
      address 10.20.130.1/24;
    }
  }
```



```
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

Displaying the Firewall Filters Applied to the Logical Interface

Purpose Verify that the firewall filter is applied to IPv4 input traffic at the logical interface.

Action Use the **show interfaces** operational mode command for the logical interface **ge-2/0/5.0**, and specify **detail** mode. The **Protocol inet** section of the command output displays IPv4 information for the logical interface. Within that section, the **Input Filters** field displays the name of the firewall filter applied to IPv4 input traffic at the logical interface.

```
user@host> show interfaces ge-2/0/5.0 detail
  Logical interface ge-2/0/5.0 (Index 105) (SNMP ifIndex 556) (Generation 170)
    Flags: Device-Down SNMP-Traps 0x4004000 Encapsulation: ENET2
    Traffic statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Local statistics:
     Input  bytes  :                   0
     Output bytes  :                   0
     Input  packets:                   0
     Output packets:                   0
    Transit statistics:
     Input  bytes  :                   0                    0 bps
     Output bytes  :                   0                    0 bps
     Input  packets:                   0                    0 pps
     Output packets:                   0                    0 pps
    Protocol inet, MTU: 1500, Generation: 242, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Input Filters: filter-srtcm1ca-all
      Addresses, Flags: Dest-route-down Is-Preferred Is-Primary
        Destination: 10.20.130/24, Local: 10.20.130.1, Broadcast: 10.20.130.255,

        Generation: 171
    Protocol multiservice, MTU: Unlimited, Generation: 243, Route table: 0
      Policer: Input: __default_arp_policer__
```

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Three-Color Policer Configuration Guidelines on page 1118

Basic Two-Rate Three-Color Policers

Two-Rate Three-Color Policer Overview

A two-rate three-color policer defines two bandwidth limits (one for guaranteed traffic and one for peak traffic) and two burst sizes (one for each of the bandwidth limits). A two-rate three-color policer is most useful when a service is structured according to arrival rates and not necessarily packet length.

Two-rate three-color policing meters a traffic stream based on the following configured traffic criteria:

- Committed information rate (CIR)—Bandwidth limit for guaranteed traffic.
- Committed burst size (CBS)—Maximum packet size permitted for bursts of data that exceed the CIR.
- Peak information rate (PIR)—Bandwidth limit for peak traffic.
- Peak burst size (PBS)—Maximum packet size permitted for bursts of data that exceed the PIR.

Two-rate tricolor marking (two-rate TCM) classifies traffic as belonging to one of three color categories and performs congestion-control actions on the packets based on the color marking:

- Green—Traffic that conforms to the bandwidth limit and burst size for guaranteed traffic (CIR and CBS). For a green traffic flow, two-rate TCM marks the packets with an implicit loss priority of **low** and transmits the packets.
- Yellow—Traffic that exceeds the bandwidth limit or burst size for guaranteed traffic (CIR or CBS) but not the bandwidth limit and burst size for peak traffic (PIR and PBS). For a yellow traffic flow, two-rate TCM marks packets with an implicit loss priority of **medium-high** and transmits the packets.
- Red—Traffic that exceeds the bandwidth limit and burst size for peak traffic (PIR and PBS). For a red traffic flow, two-rate TCM marks packets with an implicit loss priority of **high** and, optionally, discards the packets.

If congestion occurs downstream, the packets with higher loss priority are more likely to be discarded.

NOTE: For both single-rate and two-rate three-color policers, the only *configurable* action is to discard packets in a red traffic flow.

For a tricolor marking policer referenced by a firewall filter term, the **discard** policing action is supported on the following routing platforms:

- EX Series switches
- M7i and M10i routers with the Enhanced CFEB (CFEB-E)
- M120 and M320 routers with Enhanced-III FPCs
- MX Series routers with Trio MPCs

To apply a tricolor marking policer on these routing platforms, it is not necessary to include the **logical-interface-policer** statement.

Example: Configuring a Two-Rate Three-Color Policer

This example shows how to configure a two-rate three-color policer.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

A two-rate three-color policer meters a traffic flow against a bandwidth limit and burst-size limit for guaranteed traffic, plus a bandwidth limit and burst-size limit for peak traffic. Traffic that conforms to the limits for guaranteed traffic is categorized as green, and nonconforming traffic falls into one of two categories:

- Nonconforming traffic that does not exceed peak traffic limits is categorized as yellow.
- Nonconforming traffic that exceeds peak traffic limits is categorized as red.

Each category is associated with an action. For green traffic, packets are implicitly set with a loss-priority value of **low** and then transmitted. For yellow traffic, packets are implicitly set with a loss-priority value of **medium-high** and then transmitted. For red traffic, packets are implicitly set with a loss-priority value of **high** and then transmitted. If the policer configuration includes the optional action statement (**action** loss-priority high then discard), then packets in a red flow are discarded instead.

You can apply a three-color policer to Layer 3 traffic as a firewall filter policer only. You reference the policer from a stateless firewall filter term, and then you apply the filter to the input or output of a logical interface at the protocol level.

Topology

In this example, you apply a color-aware, two-rate three-color policer to the input IPv4 traffic at logical interface **fe-0/1/1.0**. The IPv4 firewall filter term that references the policer does not apply any packet-filtering. The filter is used only to apply the three-color policer to the interface.

You configure the policer to rate-limit traffic to a bandwidth limit of 40 Mbps and a burst-size limit of 100 KB for green traffic, and you configure the policer to also allow a peak bandwidth limit of 60 Mbps and a peak burst-size limit of 200 KB for yellow traffic. Only nonconforming traffic that exceeds the peak traffic limits is categorized as red. In this example, you configure the three-color policer action **loss-priority high then discard**, which overrides the implicit marking of red traffic to a **high** loss priority.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring a Two-Rate Three-Color Policer on page 1130
- Configuring an IPv4 Stateless Firewall Filter That References the Policer on page 1131
- Applying the Filter to a Logical Interface at the Protocol Family Level on page 1132

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and then paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

```
set firewall three-color-policer trTCM1-ca two-rate color-aware
set firewall three-color-policer trTCM1-ca two-rate committed-information-rate 40m
set firewall three-color-policer trTCM1-ca two-rate committed-burst-size 100k
set firewall three-color-policer trTCM1-ca two-rate peak-information-rate 60m
set firewall three-color-policer trTCM1-ca two-rate peak-burst-size 200k
set firewall three-color-policer trTCM1-ca action loss-priority high then discard
set firewall family inet filter filter-trtcm1ca-all term 1 then three-color-policer two-rate
   trTCM1-ca
set interfaces ge-2/0/5 unit 0 family inet address 10.10.10.1/30
set interfaces ge-2/0/5 unit 0 family inet filter input filter-trtcm1ca-all
set class-of-service interfaces ge-2/0/5 forwarding-class af
```

Configuring a Two-Rate Three-Color Policer

Step-by-Step Procedure

To configure a two-rate three-color policer:

1. Enable configuration of a three-color policer.

   ```
   [edit]
   user@host# set firewall three-color-policer trTCM1-ca
   ```

2. Configure the color mode of the two-rate three-color policer.

   ```
   [edit firewall three-color-policer trTCM1-ca]
   user@host# set two-rate color-aware
   ```

3. Configure the two-rate guaranteed traffic limits.

   ```
   [edit firewall three-color-policer trTCM1-ca]
   user@host# set two-rate committed-information-rate 40m
   user@host# set two-rate committed-burst-size 100k
   ```

   Traffic that does not exceed both of these limits is categorized as green. Packets in a green flow are implicitly set to **low** loss priority and then transmitted.

4. Configure the two-rate peak traffic limits.

   ```
   [edit firewall three-color-policer trTCM1-ca]
   user@host# set two-rate peak-information-rate 60m
   user@host# set two-rate peak-burst-size 200k
   ```

   Nonconforming traffic that does not exceed both of these limits is categorized as yellow. Packets in a yellow flow are implicitly set to **medium-high** loss priority and then transmitted. Nonconforming traffic that exceeds both of these limits is categorized as red. Packets in a red flow are implicitly set to **high** loss priority.

5. (Optional) Configure the policer action for red traffic.

   ```
   [edit firewall three-color-policer trTCM1-ca]
   user@host# set action loss-priority high then discard
   ```

   For three-color policers, the only configurable action is to discard red packets. Red packets are packets that have been assigned high loss priority because they exceeded the peak information rate (PIR) and the peak burst size (PBS).

Results

Confirm the configuration of the policer by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
three-color-policer trTCM1-ca {
  action {
    loss-priority high then discard;
  }
  two-rate {
    color-aware;
    committed-information-rate 40m;
    committed-burst-size 100k;
    peak-information-rate 60m;
    peak-burst-size 200k;
  }
}
```

Configuring an IPv4 Stateless Firewall Filter That References the Policer

Step-by-Step Procedure

To configure an IPv4 stateless firewall filter that references the policer:

1. Enable configuration of an IPv4 standard stateless firewall filter.

   ```
   [edit]
   user@host# set firewall family inet filter filter-trtcm1ca-all
   ```

2. Specify the filter term that references the policer.

   ```
   [edit firewall family inet filter filter-trtcm1ca-all]
   user@host# set term 1 then three-color-policer two-rate trTCM1-ca
   ```


Note that the term does not specify any match conditions. The firewall filter passes all packets to the policer.

Results

Confirm the configuration of the firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter filter-trtcm1ca-all {
    term 1 {
      then {
        three-color-policer {
          two-rate trTCM1-ca;
        }
      }
    }
  }
}
three-color-policer trTCM1-ca {
  action {
    loss-priority high then discard;
  }
  two-rate {
    color-aware;
    committed-information-rate 40m;
    committed-burst-size 100k;
    peak-information-rate 60m;
    peak-burst-size 200k;
  }
}
```

Applying the Filter to a Logical Interface at the Protocol Family Level

Step-by-Step Procedure

To apply the filter to the logical interface at the protocol family level:

1. Enable configuration of an IPv4 firewall filter.

   ```
   [edit]
   user@host# edit interfaces ge-2/0/5 unit 0 family inet
   ```

2. Apply the policer to the logical interface at the protocol family level.

   ```
   [edit interfaces ge-2/0/5 unit 0 family inet]
   user@host# set address 10.10.10.1/30
   user@host# set filter input filter-trtcm1ca-all
   ```

3. (MX Series routers and EX Series switches only) (Optional) For input policers, you can configure a fixed classifier. A fixed classifier reclassifies all incoming packets, regardless of any preexisting classification.

NOTE: Platform support depends on the Junos OS release in your implementation.

```
[edit]
user@host# set class-of-service interfaces ge-2/0/5 forwarding-class af
```

The classifier name can be a configured classifier or one of the default classifiers.

Results Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-2/0/5 {
  unit 0 {
    family inet {
      address 10.10.10.1/30;
      filter {
        input filter-trtcm1ca-all;
      }
    }
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying the Firewall Filters Applied to the Logical Interface on page 1133

Displaying the Firewall Filters Applied to the Logical Interface

Purpose Verify that the firewall filter is applied to IPv4 input traffic at the logical interface.

Action Use the **show interfaces** operational mode command for the logical interface **ge-2/0/5.0**, and specify **detail** mode. The **Protocol inet** section of the command output displays IPv4 information for the logical interface. Within that section, the **Input Filters** field displays the name of IPv4 firewall filters associated with the logical interface.

```
user@host> show interfaces ge-2/0/5.0 detail
  Logical interface ge-2/0/5.0 (Index 105) (SNMP ifIndex 556) (Generation 170)
    Flags: Device-Down SNMP-Traps 0x4004000 Encapsulation: ENET2
    Traffic statistics:
     Input  bytes  :                   0
```



```
      Output bytes  :                     0
      Input  packets:                     0
      Output packets:                     0
     Local statistics:
      Input  bytes  :                     0
      Output bytes  :                     0
      Input  packets:                     0
      Output packets:                     0
     Transit statistics:
      Input  bytes  :                     0                    0 bps
      Output bytes  :                     0                    0 bps
      Input  packets:                     0                    0 pps
      Output packets:                     0                    0 pps
     Protocol inet, MTU: 1500, Generation: 242, Route table: 0
       Flags: Sendbcast-pkt-to-re
       Input Filters: filter-trtcm1ca-all
       Addresses, Flags: Dest-route-down Is-Preferred Is-Primary
         Destination: 10.20.130/24, Local: 10.20.130.1, Broadcast: 10.20.130.255,

         Generation: 171
     Protocol multiservice, MTU: Unlimited, Generation: 243, Route table: 0
       Policer: Input: __default_arp_policer__
```

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Three-Color Policer Configuration Guidelines on page 1118

Example: Configuring a Two-Rate Three-Color Policer

This example shows how to configure a two-rate three-color policer.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

A two-rate three-color policer meters a traffic flow against a bandwidth limit and burst-size limit for guaranteed traffic, plus a bandwidth limit and burst-size limit for peak traffic. Traffic that conforms to the limits for guaranteed traffic is categorized as green, and nonconforming traffic falls into one of two categories:

- Nonconforming traffic that does not exceed peak traffic limits is categorized as yellow.
- Nonconforming traffic that exceeds peak traffic limits is categorized as red.

Each category is associated with an action. For green traffic, packets are implicitly set with a loss-priority value of **low** and then transmitted. For yellow traffic, packets are implicitly set with a loss-priority value of **medium-high** and then transmitted. For red traffic, packets are implicitly set with a loss-priority value of **high** and then transmitted. If the policer configuration includes the optional action statement (**action** loss-priority high then discard), then packets in a red flow are discarded instead.

You can apply a three-color policer to Layer 3 traffic as a firewall filter policer only. You reference the policer from a stateless firewall filter term, and then you apply the filter to the input or output of a logical interface at the protocol level.

Topology

In this example, you apply a color-aware, two-rate three-color policer to the input IPv4 traffic at logical interface **fe-0/1/1.0**. The IPv4 firewall filter term that references the policer does not apply any packet-filtering. The filter is used only to apply the three-color policer to the interface.

You configure the policer to rate-limit traffic to a bandwidth limit of 40 Mbps and a burst-size limit of 100 KB for green traffic, and you configure the policer to also allow a peak bandwidth limit of 60 Mbps and a peak burst-size limit of 200 KB for yellow traffic. Only nonconforming traffic that exceeds the peak traffic limits is categorized as red. In this example, you configure the three-color policer action **loss-priority high then discard**, which overrides the implicit marking of red traffic to a **high** loss priority.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring a Two-Rate Three-Color Policer on page 1136
- Configuring an IPv4 Stateless Firewall Filter That References the Policer on page 1137
- Applying the Filter to a Logical Interface at the Protocol Family Level on page 1138

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, copy and then paste the commands into the CLI at the **[edit]** hierarchy level, and then enter **commit** from configuration mode.

```
set firewall three-color-policer trTCM1-ca two-rate color-aware
set firewall three-color-policer trTCM1-ca two-rate committed-information-rate 40m
set firewall three-color-policer trTCM1-ca two-rate committed-burst-size 100k
set firewall three-color-policer trTCM1-ca two-rate peak-information-rate 60m
set firewall three-color-policer trTCM1-ca two-rate peak-burst-size 200k
set firewall three-color-policer trTCM1-ca action loss-priority high then discard
set firewall family inet filter filter-trtcm1ca-all term 1 then three-color-policer two-rate
  trTCM1-ca
set interfaces ge-2/0/5 unit 0 family inet address 10.10.10.1/30
set interfaces ge-2/0/5 unit 0 family inet filter input filter-trtcm1ca-all
set class-of-service interfaces ge-2/0/5 forwarding-class af
```

Configuring a Two-Rate Three-Color Policer

Step-by-Step Procedure

To configure a two-rate three-color policer:

1. Enable configuration of a three-color policer.

   ```
   [edit]
   user@host# set firewall three-color-policer trTCM1-ca
   ```

2. Configure the color mode of the two-rate three-color policer.

   ```
   [edit firewall three-color-policer trTCM1-ca]
   user@host# set two-rate color-aware
   ```

3. Configure the two-rate guaranteed traffic limits.

   ```
   [edit firewall three-color-policer trTCM1-ca]
   user@host# set two-rate committed-information-rate 40m
   user@host# set two-rate committed-burst-size 100k
   ```

   Traffic that does not exceed both of these limits is categorized as green. Packets in a green flow are implicitly set to **low** loss priority and then transmitted.

4. Configure the two-rate peak traffic limits.

   ```
   [edit firewall three-color-policer trTCM1-ca]
   user@host# set two-rate peak-information-rate 60m
   user@host# set two-rate peak-burst-size 200k
   ```

   Nonconforming traffic that does not exceed both of these limits is categorized as yellow. Packets in a yellow flow are implicitly set to **medium-high** loss priority and then transmitted. Nonconforming traffic that exceeds both of these limits is categorized as red. Packets in a red flow are implicitly set to **high** loss priority.

5. (Optional) Configure the policer action for red traffic.

   ```
   [edit firewall three-color-policer trTCM1-ca]
   user@host# set action loss-priority high then discard
   ```

   For three-color policers, the only configurable action is to discard red packets. Red packets are packets that have been assigned high loss priority because they exceeded the peak information rate (PIR) and the peak burst size (PBS).

Results

Confirm the configuration of the policer by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
three-color-policer trTCM1-ca {
  action {
    loss-priority high then discard;
```

```
    }
    two-rate {
      color-aware;
      committed-information-rate 40m;
      committed-burst-size 100k;
      peak-information-rate 60m;
      peak-burst-size 200k;
    }
  }
```

Configuring an IPv4 Stateless Firewall Filter That References the Policer

Step-by-Step Procedure

To configure an IPv4 stateless firewall filter that references the policer:

1. Enable configuration of an IPv4 standard stateless firewall filter.

   ```
   [edit]
   user@host# set firewall family inet filter filter-trtcm1ca-all
   ```

2. Specify the filter term that references the policer.

   ```
   [edit firewall family inet filter filter-trtcm1ca-all]
   user@host# set term 1 then three-color-policer two-rate trTCM1-ca
   ```

   Note that the term does not specify any match conditions. The firewall filter passes all packets to the policer.

Results

Confirm the configuration of the firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter filter-trtcm1ca-all {
    term 1 {
      then {
        three-color-policer {
          two-rate trTCM1-ca;
        }
      }
    }
  }
}
three-color-policer trTCM1-ca {
  action {
    loss-priority high then discard;
  }
  two-rate {
    color-aware;
    committed-information-rate 40m;
    committed-burst-size 100k;
    peak-information-rate 60m;
    peak-burst-size 200k;
```

```
    }
  }
```

Applying the Filter to a Logical Interface at the Protocol Family Level

Step-by-Step Procedure

To apply the filter to the logical interface at the protocol family level:

1. Enable configuration of an IPv4 firewall filter.

   ```
   [edit]
   user@host# edit interfaces ge-2/0/5 unit 0 family inet
   ```

2. Apply the policer to the logical interface at the protocol family level.

   ```
   [edit interfaces ge-2/0/5 unit 0 family inet]
   user@host# set address 10.10.10.1/30
   user@host# set filter input filter-trtcm1ca-all
   ```

3. (MX Series routers and EX Series switches only) (Optional) For input policers, you can configure a fixed classifier. A fixed classifier reclassifies all incoming packets, regardless of any preexisting classification.

   NOTE: Platform support depends on the Junos OS release in your implementation.

   ```
   [edit]
   user@host# set class-of-service interfaces ge-2/0/5 forwarding-class af
   ```

   The classifier name can be a configured classifier or one of the default classifiers.

Results

Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-2/0/5 {
  unit 0 {
    family inet {
      address 10.10.10.1/30;
      filter {
        input filter-trtcm1ca-all;
      }
    }
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying the Firewall Filters Applied to the Logical Interface on page 1139

Displaying the Firewall Filters Applied to the Logical Interface

Purpose Verify that the firewall filter is applied to IPv4 input traffic at the logical interface.

Action Use the **show interfaces** operational mode command for the logical interface **ge-2/0/5.0**, and specify **detail** mode. The **Protocol inet** section of the command output displays IPv4 information for the logical interface. Within that section, the **Input Filters** field displays the name of IPv4 firewall filters associated with the logical interface.

```
user@host> show interfaces ge-2/0/5.0 detail
  Logical interface ge-2/0/5.0 (Index 105) (SNMP ifIndex 556) (Generation 170)
    Flags: Device-Down SNMP-Traps 0x4004000 Encapsulation: ENET2
    Traffic statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Local statistics:
     Input  bytes  :                    0
     Output bytes  :                    0
     Input  packets:                    0
     Output packets:                    0
    Transit statistics:
     Input  bytes  :                    0                    0 bps
     Output bytes  :                    0                    0 bps
     Input  packets:                    0                    0 pps
     Output packets:                    0                    0 pps
    Protocol inet, MTU: 1500, Generation: 242, Route table: 0
      Flags: Sendbcast-pkt-to-re
      Input Filters: filter-trtcm1ca-all
      Addresses, Flags: Dest-route-down Is-Preferred Is-Primary
        Destination: 10.20.130/24, Local: 10.20.130.1, Broadcast: 10.20.130.255,

        Generation: 171
    Protocol multiservice, MTU: Unlimited, Generation: 243, Route table: 0
      Policer: Input: __default_arp_policer__
```

Related Documentation

- Two-Rate Three-Color Policer Overview on page 1128

CHAPTER 32

Configuring Logical and Physical Interface Traffic Policers at Layer 3

Two-Color and Three-Color Logical Interface Policers

Logical Interface (Aggregate) Policer Overview

A *logical interface policer*—also called an *aggregate policer*—is a two-color or three-color policer that defines traffic rate limiting that you can apply to input or output traffic for multiple protocol families on the same logical interface without creating multiple instances of the policer.

To configure a single-rate two-color logical interface policer, include the **logical-interface-policer** statement at one of the following hierarchy levels:

- **[edit firewall policer *policer-name*]**
- **[edit logical-systems *logical-system-name* firewall policer *policer-name*]**

To configure a single-rate or two-rate three-color logical interface policer, include the **logical-interface-policer** statement at one of the following hierarchy levels:

- **[edit firewall three-color-policer *name*]**
- **[edit logical-systems *logical-system-name* firewall three-color-policer *name*]**

NOTE: A three-color policer can be applied to Layer 2 traffic as a logical interface policer only. You cannot apply a three-color policer to Layer 2 traffic as a physical interface policer (through a firewall filter).

You apply a logical interface policer to Layer 3 traffic directly to the interface configuration at the logical unit level (to rate-limit all traffic types, regardless of the protocol family) or at the protocol family level (to rate-limit traffic of a specific protocol family). It is OK to reference a logical interface policer from a stateless firewall filter term and then apply the filter to a logical interface.

You can apply a logical interface policer to unicast traffic only. For information about configuring a stateless firewall filter for flooded traffic, see "*Applying Forwarding Table Filters*" in the "Traffic Sampling, Forwarding, and Monitoring" section of the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*.

To display a logical interface policer on a particular interface, issue the **show interfaces policers** operational mode command.

Example: Configuring a Two-Color Logical Interface (Aggregate) Policer

This example shows how to configure a single-rate two-color policer as a logical interface policer and apply it to incoming IPv4 traffic on a logical interface.

Requirements

Before you begin, make sure that the logical interface to which you apply the two-color logical interface policer is hosted on a Gigabit Ethernet interface (**ge-**) or a 10-Gigabit Ethernet interface (**xe-**).

Overview

In this example, you configure the single-rate two-color policer **policer_IFL** as a logical interface policer and apply it to incoming IPv4 traffic at logical interface **ge-1/3/1.0**.

Topology

If the input IPv4 traffic on the physical interface **ge-1/3/1** exceeds the bandwidth limit equal to 90 percent of the media rate with a 300 KB burst-size limit, then the logical interface policer **policer_IFL** rate-limits the input IPv4 traffic on the logical interface **ge-1/3/1.0**. Configure the policer to mark nonconforming traffic by setting packet loss priority (PLP) levels to **high** and classifying packets as **best-effort**.

As the incoming IPv4 traffic rate on the physical interface slows and conforms to the configured limits, Junos OS stops marking the incoming IPv4 packets at the logical interface.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring the Logical Interfaces on page 1143
- Configuring the Single-Rate Two-Color Policer as a Logical Interface Policer on page 1144
- Applying the Logical Interface Policer to Input IPv4 Traffic at a Logical Interface on page 1146

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces ge-1/3/1 vlan-tagging
set interfaces ge-1/3/1 unit 0 vlan-id 100
set interfaces ge-1/3/1 unit 0 family inet address 10.10.10.1/30
set interfaces ge-1/3/1 unit 1 vlan-id 101
set interfaces ge-1/3/1 unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
  00:00:11:22:33:44
set firewall policer policer_IFL logical-interface-policer
set firewall policer policer_IFL if-exceeding bandwidth-percent 90
set firewall policer policer_IFL if-exceeding burst-size-limit 300k
set firewall policer policer_IFL then loss-priority high
set firewall policer policer_IFL then forwarding-class best-effort
set interfaces ge-1/3/1 unit 0 family inet policer input policer_IFL
```

Configuring the Logical Interfaces

Step-by-Step Procedure

To configure the logical interfaces:

1. Enable configuration of the interface.

```
[edit]
user@host# edit interfaces ge-1/3/1
```

2. Configure single tagging.

```
[edit interfaces ge-1/3/1]
user@host# set vlan-tagging
```

3. Configure logical interface **ge-1/3/1.0**.

```
[edit interfaces ge-1/3/1]
user@host# set unit 0 vlan-id 100
user@host# set unit 0 family inet address 10.10.10.1/30
```

4. Configure logical interface **ge-1/3/1.0**.

```
[edit interfaces ge-1/3/1]
user@host# set unit 1 vlan-id 101
user@host# set unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
  00:00:11:22:33:44
```

Results

Confirm the configuration of the logical interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/1 {
  vlan-tagging;
  unit 0 {
    vlan-id 100;
    family inet {
      address 10.10.10.1/30;
    }
  }
  unit 1 {
    vlan-id 101;
    family inet {
      address 20.20.20.1/30 {
        arp 20.20.20.2 mac 00:00:11:22:33:44;
      }
    }
  }
}
```

Configuring the Single-Rate Two-Color Policer as a Logical Interface Policer

Step-by-Step Procedure

To configure a single-rate two-color policer as a logical interface policer:

1. Enable configuration of a single-rate two-color policer.

```
[edit]
user@host# edit firewall policer policer_IFL
```

2. Specify that the policer is a logical interface (aggregate) policer.

```
[edit firewall policer policer_IFL]
user@host# set logical-interface-policer
```

   A logical interface policer rate-limits traffic based on a percentage of the media rate of the physical interface underlying the logical interface to which the policer is applied. The policer is applied directly to the interface rather than referenced by a firewall filter.

3. Specify the policer traffic limits.

   a. Specify the bandwidth limit.

   - To specify the bandwidth limit as an absolute rate, from 8,000 bits per second through 50,000,000,000 bits per second, include the **bandwidth-limit** ***bps*** statement.
   - To specify the bandwidth limit as a percentage of the physical port speed on the interface, include the **bandwidth-percent** ***percent*** statement.

In this example, the CLI commands and output are based on a bandwidth limit specified as a percentage rather than as an absolute rate.

```
[edit firewall policer policer_IFL]
user@host# set if-exceeding bandwidth-percent 90
```

b. Specify the burst-size limit, from 1,500 bytes through 100,000,000,000 bytes, which is the maximum packet size to be permitted for bursts of data that exceed the specified bandwidth limit.

```
[edit firewall policer policer_IFL]
user@host# set if-exceeding burst-size-limit 300k
```

4. Specify the policer actions to be taken on traffic that exceeds the configured rate limits.

   - To discard the packet, include the **discard** statement.
   - To set the loss-priority value of the packet, include the **loss-priority (low | medium-low | medium-high | high)** statement.
   - To classify the packet to a forwarding class, include the **forwarding-class (*forwarding-class* | assured-forwarding | best-effort | expedited-forwarding | network-control)** statement.

   In this example, the CLI commands and output are based on both setting the packet loss priority level and classifying the packet.

```
[edit firewall policer policer_IFL]
user@host# set then loss-priority high
user@host# set then forwarding-class best-effort
```

Results Confirm the configuration of the policer by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer policer_IFL {
  logical-interface-policer;
  if-exceeding {
    bandwidth-percent 90;
    burst-size-limit 300k;
  }
  then {
    loss-priority high;
    forwarding-class best-effort;
  }
}
```

Applying the Logical Interface Policer to Input IPv4 Traffic at a Logical Interface

Step-by-Step Procedure

To apply the two-color logical interface policer to input IPv4 traffic a logical interface:

1. Enable configuration of the logical interface.

```
[edit]
user@host# edit interfaces ge-1/3/1 unit 0
```

2. Apply the policer to all traffic types or to a specific traffic type on the logical interface.

   - To apply the policer to all traffic types, regardless of the protocol family, include the **policer (input | output)** ***policer-name*** statement at the **[edit interfaces** ***interface-name*** **unit** ***number*****]** hierarchy level.
   - To apply the policer to traffic of a specific protocol family, include the **policer (input | output)** ***policer-name*** statement at the **[edit interfaces** ***interface-name*** **unit** ***unit-number*** **family** ***family-name*****]** hierarchy level.

   To apply the logical interface policer to incoming packets, use the **policer input** ***policer-name*** statement. To apply the logical interface policer to outgoing packets, use the **policer output** ***policer-name*** statement.

   In this example, the CLI commands and output are based on rate-limiting the IPv4 input traffic at logical interface **ge-1/3/1.0**.

```
[edit interfaces ge-1/3/1 unit 0]
user@host# set family inet policer input policer_IFL
```

Results

Confirm the configuration of the interface by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/1 {
   vlan-tagging;
   unit 0 {
      vlan-id 100;
      family inet {
         policer input policer_IFL;
         address 10.10.10.1/30;
      }
   }
   unit 1 {
      vlan-id 101;
      family inet {
         address 20.20.20.1/30 {
            arp 20.20.20.2 mac 00:00:11:22:33:44;
         }
      }
   }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying Traffic Statistics and Policers for the Logical Interface on page 1147
- Displaying Statistics for the Policer on page 1147

Displaying Traffic Statistics and Policers for the Logical Interface

Purpose Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

Action Use the **show interfaces** operational mode command for logical interface **ge-1/3/1.0**, and include the **detail** or **extensive** option. The command output section for **Traffic statistics** lists the number of bytes and packets received and transmitted on the logical interface. The **Protocol inet** subsection contains a **Policer** field that would list the policer **policer_IFL** as an input or output logical interface policer as follows:

- **Input: policer_IFL-ge-1/3/1.0-log_int-i**
- **Output: policer_IFL-ge-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to input traffic only.

Displaying Statistics for the Policer

Purpose Verify the number of packets evaluated by the policer.

Action Use the **show policer** operational mode command and optionally specify the name of the policer. The command output displays the number of packets evaluated by each configured policer (or the specified policer), in each direction. For the policer **policer_IFL**, the input and output policer names are displayed as follows:

- **policer_IFL-ge-1/3/1.0-log_int-i**
- **policer_IFL-ge-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to input traffic only.

Example: Configuring a Three-Color Logical Interface (Aggregate) Policer

This example shows how to configure a two-rate three-color color-blind policer as a logical interface (aggregate) policer and apply the policer directly to Layer 2 input traffic at a supported logical interface.

Requirements

Before you begin, make sure that the logical interface to which you apply the three-color logical interface policer is hosted on a Gigabit Ethernet interface (**ge-**) or a 10-Gigabit Ethernet interface (**xe-**) on an MX Series router.

Overview

A two-rate three-color policer meters a traffic flow against a bandwidth limit and burst-size limit for guaranteed traffic, plus a second set of bandwidth and burst-size limits for peak traffic. Traffic that conforms to the limits for guaranteed traffic is categorized as green, and nonconforming traffic falls into one of two categories:

- Nonconforming traffic that does not exceed the bandwidth and burst-size limits for peak traffic is categorized as yellow.
- Nonconforming traffic that exceeds the bandwidth and burst-size limits for peak traffic is categorized as red.

A logical interface policer defines traffic rate-limiting rules that you can apply to multiple protocol families on the same logical interface without creating multiple instances of the policer.

NOTE: You apply a logical interface policer directly to a logical interface at the logical unit level, and not by referencing the policer in a stateless firewall filter and then applying the filter to the logical interface at the protocol family level.

Topology

In this example, you configure the two-rate three-color policer **trTCM2-cb** as a color-blind logical interface policer and apply the policer to incoming Layer 2 traffic on logical interface **ge-1/3/1.0**.

NOTE: When using a three-color policer to rate-limit Layer 2 traffic, color-aware policing can be applied to egress traffic only.

The policer defines guaranteed traffic rate limits such that traffic that conforms to the bandwidth limit of 40 Mbps with a 100 KB allowance for traffic bursting (based on the token-bucket formula) is categorized as green. As with any policed traffic, the packets in a green flow are implicitly set to a **low** loss priority and then transmitted.

Nonconforming traffic that falls within the peak traffic limits of a 60 Mbps bandwidth limit and a 200 KB allowance for traffic bursting (based on the token-bucket formula) is categorized as yellow. The packets in a yellow traffic flow are implicitly set to a **medium-high** loss priority and then transmitted.

Nonconforming traffic that exceeds the peak traffic limits are categorized as red. The packets in a red traffic flow are implicitly set to a **high** loss priority. In this example, the optional policer action for red traffic (**loss-priority high then discard**) is configured, so packets in a red traffic flow are discarded instead of transmitted.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring the Logical Interfaces on page 1149
- Configuring the Two-Rate Three-Color Policer as a Logical Interface Policer on page 1150
- Applying the Three-Color Policer to the Layer 2 Input at the Logical Interface on page 1152

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces ge-1/3/1 vlan-tagging
set interfaces ge-1/3/1 unit 0 vlan-id 100
set interfaces ge-1/3/1 unit 0 family inet address 10.10.10.1/30
set interfaces ge-1/3/1 unit 1 vlan-id 101
set interfaces ge-1/3/1 unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
  00:00:11:22:33:44
set firewall three-color-policer trTCM2-cb logical-interface-policer
set firewall three-color-policer trTCM2-cb two-rate color-blind
set firewall three-color-policer trTCM2-cb two-rate committed-information-rate 40m
set firewall three-color-policer trTCM2-cb two-rate committed-burst-size 100k
set firewall three-color-policer trTCM2-cb two-rate peak-information-rate 60m
set firewall three-color-policer trTCM2-cb two-rate peak-burst-size 200k
set firewall three-color-policer trTCM2-cb action loss-priority high then discard
set interfaces ge-1/3/1 unit 0 layer2-policer input-three-color trTCM2-cb
```

Configuring the Logical Interfaces

Step-by-Step Procedure

To configure the logical interfaces:

1. Enable configuration of the interface.

```
[edit]
```



```
user@host# edit interfaces ge-1/3/1
```

2. Configure single tagging.

```
[edit interfaces ge-1/3/1]
user@host# set vlan-tagging
```

3. Configure logical interface **ge-1/3/1.0**.

```
[edit interfaces ge-1/3/1]
user@host# set unit 0 vlan-id 100
user@host# set unit 0 family inet address 10.10.10.1/30
```

4. Configure logical interface **ge-1/3/1.0**.

```
[edit interfaces ge-1/3/1]
user@host# set unit 1 vlan-id 101
user@host# set unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
  00:00:11:22:33:44
```

Results

Confirm the configuration of the logical interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/1 {
  vlan-tagging;
  unit 0 {
    vlan-id 100;
    family inet {
      address 10.10.10.1/30;
    }
  }
  unit 1 {
    vlan-id 101;
    family inet {
      address 20.20.20.1/30 {
        arp 20.20.20.2 mac 00:00:11:22:33:44;
      }
    }
  }
}
```

Configuring the Two-Rate Three-Color Policer as a Logical Interface Policer

Step-by-Step Procedure

To configure the two-rate three-color policer as a logical interface policer:

1. Enable configuration of a three-color policer.

```
[edit]
user@host# edit firewall three-color-policer trTCM2-cb
```

2. Specify that the policer is a logical interface (aggregate) policer.

   ```
   [edit firewall three-color-policer trTCM2-cb]
   user@host# set logical-interface-policer
   ```

   A logical interface policer rate-limits traffic based on a percentage of the media rate of the physical interface underlying the logical interface to which the policer is applied, and the policer is applied directly to the interface rather than referenced by a firewall filter.

3. Specify that the policer is two-rate and color-blind.

   ```
   [edit firewall three-color-policer trTCM2-cb]
   user@host# set two-rate color-blind
   ```

   A color-aware three-color policer takes into account any coloring markings that might have been set for a packet by another traffic policer configured at a previous network node, and any preexisting color markings are used in determining the appropriate policing action for the packet.

   Because you are applying this three-color policer applied to input at Layer 2, you must configure the policer to be color-blind.

4. Specify the policer traffic limits used to classify a green traffic flow.

   ```
   [edit firewall three-color-policer trTCM2-cb]
   user@host# set two-rate committed-information-rate 40m
   user@host# set two-rate committed-burst-size 100k
   ```

5. Specify the additional policer traffic limits used to classify a yellow or red traffic flow.

   ```
   [edit firewall three-color-policer trTCM2-cb]
   user@host# set two-rate peak-information-rate 60m
   user@host# set two-rate peak-burst-size 200k
   ```

6. (Optional) Specify the configured policer action for packets in a red traffic flow.

   ```
   [edit firewall three-color-policer trTCM2-cb]
   user@host# set action loss-priority high then discard
   ```

   In color-aware mode, the three-color policer configured action can increase the packet loss priority (PLP) level of a packet, but never decrease it. For example, if a color-aware three-color policer meters a packet with a medium PLP marking, it can raise the PLP level to high, but cannot reduce the PLP level to low.

Results Confirm the configuration of the three-color policer by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
three-color-policer trTCM2-cb {
```

```
    logical-interface-policer;
    action {
      loss-priority high then discard;
    }
    two-rate {
      color-blind;
      committed-information-rate 40m;
      committed-burst-size 100k;
      peak-information-rate 60m;
      peak-burst-size 200k;
    }
  }
```

Applying the Three-Color Policer to the Layer 2 Input at the Logical Interface

Step-by-Step Procedure

To apply the three-color policer to the Layer 2 input at the logical interface:

1. Enable application of Layer 2 logical interface policers.

```
[edit]
user@host# edit interfaces ge-1/3/1 unit 0
```

2. Apply the three-color logical interface policer to a logical interface input.

```
[edit interfaces ge-1/3/1 unit 0]
user@host# set layer2-policerinput-three-color trTCM2-cb
```

Results

Confirm the configuration of the logical interfaces by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/1 {
  vlan-tagging;
  unit 0 {
    vlan-id 100;
    layer2-policer {
      input-three-color trTCM2-cb;
    }
    family inet {
      address 10.10.10.1/30;
    }
  }
  unit 1 {
    vlan-id 101;
    family inet {
      address 20.20.20.1/30 {
        arp 20.20.20.2 mac 00:00:11:22:33:44;
      }
    }
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying Traffic Statistics and Policers for the Logical Interface on page 1153
- Displaying Statistics for the Policer on page 1153

Displaying Traffic Statistics and Policers for the Logical Interface

Purpose Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

Action Use the **show interfaces** operational mode command for logical interface **ge-1/3/1.0**, and include the **detail** or **extensive** option. The command output section for **Traffic statistics** lists the number of bytes and packets received and transmitted on the logical interface, and the **Protocol inet** section contains a **Policer** field that would list the policer **trTCM2-cb** as an input or output policer as follows:

- **Input: trTCM2-cb-ge-1/3/1.0-log_int-i**
- **Output: trTCM2-cb-ge-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to in the input direction only.

Displaying Statistics for the Policer

Purpose Verify the number of packets evaluated by the policer.

Action Use the **show policer** operational mode command and optionally specify the name of the policer. The command output displays the number of packets evaluated by each configured policer (or the specified policer), in each direction. For the policer **trTCM2-cb**, the input and output policer names are displayed as follows:

- **trTCM2-cb-ge-1/3/1.0-log_int-i**
- **trTCM2-cb-e-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to input traffic only.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Three-Color Policer Configuration Overview on page 1115
- Guidelines for Applying Traffic Policers on page 983

Two-Color and Three-Color Physical Interface Policers

- Physical Interface Policer Overview on page 1154
- Example: Configuring a Physical Interface Policer for Aggregate Traffic at a Physical Interface on page 1155

Physical Interface Policer Overview

A *physical interface policer* is a two-color or three-color policer that defines traffic rate limiting that you can apply to input or output traffic for all the logical interfaces and protocol families configured on a physical interface, even if the logical interfaces belong to different routing instances. This feature is useful when you want to perform aggregate policing for different protocol families and different logical interfaces on the same physical interface.

For example, suppose that a provider edge (PE) router has numerous logical interfaces, each corresponding to a different customer, configured on the same link to a customer edge (CE) device. Now suppose that a customer wants to apply one set of rate limits aggregately for certain types of traffic on a single physical interface. To accomplish this, you could apply a single physical interface policer to the physical interface, which rate-limits all the logical interfaces configured on the interface and all the routing instances to which those interfaces belong.

To configure a single-rate two-color physical interface policer, include the **physical-interface-policer** statement at one of the following hierarchy levels:

- **[edit firewall policer *policer-name*]**
- **[edit logical-system *logical-system-name* firewall policer *policer-name*]**
- **[edit routing-instances *routing-instance-name* firewall policer *policer-name*]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* firewall policer *policer-name*]**

To configure a single-rate or two-rate three-color physical interface policer, include the **physical-interface-policer** statement at one of the following hierarchy levels:

- **[edit firewall three-color-policer *policer-name*]**
- **[edit logical-system *logical-system-name* firewall three-color-policer *policer-name*]**
- **[edit routing-instances *routing-instance-name* firewall three-color-policer *policer-name*]**
- **[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* firewall three-color-policer *policer-name*]**

You apply a physical interface policer to Layer 3 traffic by referencing the policer from a stateless firewall filter term and then applying the filter to a logical interface. You cannot apply a physical interface to Layer 3 traffic directly to the interface configuration.

To reference a single-rate two-color policer from a stateless firewall filter term, use the **policer** nonterminating action. To reference a single-rate or two-rate three-color policer from a stateless firewall filter term, use the **three-color-policer** nonterminating action.

The following requirements apply to a stateless firewall filter that references a physical interface policer:

- You must configure the firewall filter for a specific, supported protocol family: **ipv4**, **ipv6**, **mpls**, **vpls**, or circuit cross-connect (**ccc**), but not for **family any**.
- You must configure the firewall filter as a *physical interface filter* by including the **physical-interface-filter** statement at the **[edit firewall family *family-name* filter *filter-name*]** hierarchy level.
- A firewall filter that is defined as a physical interface filter can reference a physical interface policer only.
- A firewall filter that is defined as a physical interface filter cannot reference a policer configured with the **interface-specific** statement.
- You cannot configure a firewall filter as both a physical interface filter and as a logical interface filter that also includes the **interface-specific** statement.

Example: Configuring a Physical Interface Policer for Aggregate Traffic at a Physical Interface

This example shows how to configure a single-rate two-color policer as a physical interface policer.

Requirements

No special configuration beyond device initialization is required before configuring this example.

Overview

A *physical interface policer* specifies rate-limiting for aggregate traffic, which encompasses all protocol families and logical interfaces configured on a physical interface, even if the interfaces belong to different routing instances.

You can apply a physical interface policer to Layer 3 input or output traffic only by referencing the policer from a stateless firewall filter that is configured for specific a specific protocol family (not for **family any**) and configured as a physical interface filter. You configure the filter terms with match conditions that select the types of packets you want to rate-limit, and you specify the physical interface policer as the action to apply to matched packets.

Topology

The physical interface policer in this example, **shared-policer-A**, rate-limits to 10,000,000 bps and permits a maximum burst of traffic of 500,000 bytes. You configure the policer to discard packets in nonconforming flows, but you could instead configure the policer to re-mark nonconforming traffic with a forwarding class, a packet loss priority (PLP) level, or both.

To be able to use the policer to rate-limit IPv4 traffic, you reference the policer from an IPv4 physical interface filter. For this example, you configure the filter to pass the policer IPv4 packets that meet either of the following match terms:

- Packets received through TCP and with the IP precedence fields **critical-ecp** (0xa0), **immediate** (0x40), or **priority** (0x20)
- Packets received through TCP and with the IP precedence fields **internet-control** (0xc0) or **routine** (0x00)

You could also reference the policer from physical interface filters for other protocol families.

Configuration

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see *Using the CLI Editor in Configuration Mode*.

To configure this example, perform the following tasks:

- Configuring the Logical Interfaces on the Physical Interface on page 1157
- Configuring a Physical Interface Policer on page 1157
- Configuring an IPv4 Physical Interface Filter on page 1158
- Applying the IPv4 Physical interface Filter to a Physical Interface on page 1159

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the **[edit]** hierarchy level.

```
set interfaces so-1/0/0 unit 0 family inet address 192.168.1.1/24
set interfaces so-1/0/0 unit 0 family vpls
set interfaces so-1/0/0 unit 1 family mpls
set firewall policer shared-policer-A physical-interface-policer
set firewall policer shared-policer-A if-exceeding bandwidth-limit 100m burst-size-limit
  500k
set firewall policer shared-policer-A then discard
set firewall family inet filter ipv4-filter physical-interface-filter
set firewall family inet filter ipv4-filter term tcp-police-1 from precedence [ critical-ecp
  immediate priority ]
set firewall family inet filter ipv4-filter term tcp-police-1 from protocol tcp
set firewall family inet filter ipv4-filter term tcp-police-1 then policer shared-policer-A
set firewall family inet filter ipv4-filter term tcp-police-2 from precedence [ internet-control
  routine ]
set firewall family inet filter ipv4-filter term tcp-police-2 from protocol tcp
```



```
set firewall family inet filter ipv4-filter term tcp-police-2 then policer shared-policer-A
set interfaces so-1/0/0 unit 0 family inet filter input ipv4-filter
```

Configuring the Logical Interfaces on the Physical Interface

Step-by-Step Procedure

To configure the logical interfaces on the physical interface:

1. Enable configuration of logical interfaces.

```
[edit]
user@host# edit interfaces so-1/0/0
```

2. Configure protocol families on logical unit 0.

```
[edit interfaces so-1/0/0]
user@host# set unit 0 family inet address 192.168.1.1/24
user@host# set unit 0 family vpls
```

3. Configure protocol families on logical unit 1.

```
[edit interfaces so-1/0/0]
user@host# set unit 1 family mpls
```

Results

Confirm the configuration of the firewall filter by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
so-1/0/0 {
  unit 0 {
    family inet {
      address 192.168.1.1/24;
    }
    family vpls;
  }
  unit 1 {
    family mpls;
  }
}
```

Configuring a Physical Interface Policer

Step-by-Step Procedure

To configure a physical interface policer:

1. Enable configuration of the two-color policer.

```
[edit]
user@host# edit firewall policer shared-policer-A
```

2. Configure the type of two-color policer.

```
[edit firewall policer shared-policer-A]
```

```
user@host# set physical-interface-policer
```

3. Configure the traffic limits and the action for packets in a nonconforming traffic flow.

```
[edit firewall policer shared-policer-A]
user@host# set if-exceeding bandwidth-limit 100m burst-size-limit 500k
user@host# set then discard
```

For a physical interface filter, the actions you can configure for packets in a nonconforming traffic flow are to discard the packets, assign a forwarding class, assign a PLP value, or assign both a forwarding class and a PLP value.

Results Confirm the configuration of the policer by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer shared-policer-A {
  physical-interface-policer;
  if-exceeding {
    bandwidth-limit 100m;
    burst-size-limit 500k;
  }
  then discard;
}
```

Configuring an IPv4 Physical Interface Filter

Step-by-Step Procedure To configure a physical interface policer as the action for terms in an IPv4 physical interface policer:

1. Configure a standard stateless firewall filter under a specific protocol family.

```
[edit]
user@host# edit firewall family inet filter ipv4-filter
```

You cannot configure a physical interface firewall filter for **family any**.

2. Configure the filter as a physical interface filter so that you can apply the physical interface policer as an action.

```
[edit firewall family inet filter ipv4-filter]
user@host# set physical-interface-filter
```

3. Configure the first term to match IPv4 packets received through TCP with the IP precedence fields **critical-ecp**, **immediate**, or **priority** and to apply the physical interface policer as a filter action.

```
[edit firewall family inet filter ipv4-filter]
user@host# set term tcp-police-1 from precedence [ critical-ecp immediate priority
  ]
```



```
user@host# set term tcp-police-1 from protocol tcp
user@host# set term tcp-police-1 then policer shared-policer-A
```

4. Configure the first term to match IPv4 packets received through TCP with the IP precedence fields **internet-control** or **routine** and to apply the physical interface policer as a filter action.

```
[edit firewall family inet filter ipv4-filter]
user@host# set term tcp-police-2 from precedence [ internet-control routine ]
user@host# set term tcp-police-2 from protocol tcp
user@host# set term tcp-police-2 then policer shared-policer-A
```

Results Confirm the configuration of the firewall filter by entering the **show firewall** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
family inet {
  filter ipv4-filter {
    physical-interface-filter;
    term tcp-police-1 {
      from {
        precedence [ critical-ecp immediate priority ];
        protocol tcp;
      }
      then policer shared-policer-A;
    }
    term tcp-police-2 {
      from {
        precedence [ internet-control routine ];
        protocol tcp;
      }
      then policer shared-policer-A;
    }
  }
}
policer shared-policer-A {
  physical-interface-policer;
  if-exceeding {
    bandwidth-limit 100m;
    burst-size-limit 500k;
  }
  then discard;
}
```

Applying the IPv4 Physical interface Filter to a Physical Interface

Step-by-Step Procedure To apply the physical interface filter to a physical interface:

1. Enable configuration of IPv4 on the logical interface.

```
[edit]
user@host# edit interfaces so-1/0/0 unit 0 family inet
```

2. Apply the IPv4 physical interface filter in the input direction.

```
[edit interfaces so-1/0/0 unit 0 family inet]
user@host# set filter input ipv4-filter
```

Results Confirm the configuration of the firewall filter by entering the **show interfaces** configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
so-1/0/0 {
  unit 0 {
    family inet {
      filter {
        input ipv4-filter;
      }
      address 192.168.1.1/24;
    }
    family vpls;
  }
  unit 1 {
    family mpls;
  }
}
```

If you are done configuring the device, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- Displaying the Firewall Filters Applied to an Interface on page 1160
- Displaying the Number of Packets Processed by the Policer at the Logical Interface on page 1161

Displaying the Firewall Filters Applied to an Interface

Purpose Verify that the firewall filter **ipv4-filter** is applied to the IPv4 input traffic at logical interface **so-1/0/0.0**.

Action Use the show interfaces statistics operational mode command for logical interface **so-1/0/0.0**, and include the **detail** option. In the **Protocol inet** section of the command output, the **Input Filters** field shows that the firewall filter **ipv4-filter** is applied in the input direction.

```
user@host> show interfaces statistics so-1/0/0 detail
  Logical interface so-1/0/0.0 (Index 79) (SNMP ifIndex 510) (Generation 149)
    Flags: Hardware-Down Point-To-Point SNMP-Traps 0x4000 Encapsulation: PPP
    Protocol inet, MTU: 4470, Generation: 173, Route table: 0
      Flags: Sendbcast-pkt-to-re, Protocol-Down
      Input Filters: ipv4-filter
      Addresses, Flags: Dest-route-down Is-Preferred Is-Primary
```



```
        Destination: 10.39/16, Local: 10.39.1.1, Broadcast: 10.39.255.255,
Generation: 163
```

Displaying the Number of Packets Processed by the Policer at the Logical Interface

Purpose Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

Action Use the **show firewall** operational mode command for the filter you applied to the logical interface.

```
user@host> show firewall filter ipv4-filter
Filter: ipv4-filter
Policers:
Name                                                  Packets
shared-policer-A-tcp-police-1                           32863
shared-policer-A-tcp-police-2                            3870
```

The command output displays the name of policer (**shared-policer-A**), the name of the filter term (**police-1**) under which the policer action is specified, and the number of packets that matched the filter term. This is only the number of out-of-specification (out-of-spec) packet counts, not all packets policed by the policer.

Related Documentation

- Firewall Filter Match Conditions Based on Numbers or Text Aliases on page 601
- Firewall Filter Match Conditions Based on Bit-Field Values on page 602
- Firewall Filter Match Conditions Based on Address Fields on page 606
- Firewall Filter Match Conditions Based on Address Classes on page 614
- Two-Color Policer Configuration Overview on page 1027
- Three-Color Policer Configuration Overview on page 1115
- Guidelines for Applying Traffic Policers on page 983
- **physical-interface-filter on page 1319**
- **physical-interface-policer on page 1320**

PART 5

Configuration Statements and Operational Commands

CHAPTER 33

Configuration Statements

Routing Policy Configuration Statements

address-family

Syntax

```
address-family {
  inet {
    address;
    table table-name;
  }
  ccc {
    interface-name;
    standby;
    peer-unit unit-number;
    table table-name;
  }
}
```

Hierarchy Level [edit logical-systems *logical-system-name* policy-options condition if-route-exists],
[edit policy-options condition if-route-exists],

Release Information Statement introduced in Junos OS Release 13.2.

Description Specify that the route must correspond to certain prefix type to be considered a match.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring Pseudowire Redundancy in a Mobile Backhaul Scenario*

aigp-adjust (Policy Action)

Syntax aigp-adjust *(add|divide|multipy|subtract) (user-value | distance-to-protocol-nexthop)*;

Hierarchy Level [edit policy-options policy-statement *policy-name* term *term-name* then],
[edit policy-options policy-statement *policy-name* then]

Release Information Statement introduced in Junos OS Release 16.1.

Description Specify this CLI policy action in an import or export policy to modify the accumulated interior gateway protocol (AIGP) BGP attribute. You can modify the AIGP attribute for one of the following reasons:

- Modify the AIGP metric on the route if it exists on the external BGP sessions between two ASBR routers where there is no interior gateway protocol (IGP)
- Scale up or down while transitioning from one IGP domain, such as OSPF to another IGP domain, such as IS-IS
- Make very minor adjustments on the AIGP from another AS domain or another vendor's routers

CAUTION: AIGP is rated very high in the best route decision and comes only after the BGP local preference rule. Therefore, use AIGP adjustment option with caution as it can have a huge impact on the network.

Options **add | divide | multiply | subtract**—Specify the mathematical operation that needs to be performed on the original AIGP attribute for adjustment.

user-value—Specify an unsigned 64-bit value in decimal.

distance-to-protocol-nexthop—Use the current metric2 value in the routing table as specified in the routing policy. Configure this option in the export policy if you want full control of the advertised AIGP value and do not want BGP to add up the distance to protocol nexthop.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- metric (Policy Action) on page 1220
- show policy on page 1364

aigp-originate

Syntax aigp-originate *distance*;

Hierarchy Level [edit logical-systems *logical-system-name* policy-options policy-statement *policy-name* term *term-name* then],
[edit logical-systems *logical-system-name* policy-options policy-statement *policy-name* then],
[edit policy-options policy-statement *policy-name* term *term-name* then],
[edit policy-options policy-statement *policy-name* then]

Release Information Statement introduced in Junos OS Release 12.1.

Description Originate an accumulated interior gateway protocol (AIGP) BGP attribute for a given prefix by export policy, using the **aigp-originate** policy action.

To originate an AIGP attribure, you need configure the policy action on only one node. The AIGP attribute is readvertised if the neighbors are AIGP enabled with the **aigp** statement in the BGP configuration.

Default If you omit the **aigp-originate** policy action, the node still readvertises the AIGP BGP attribute if AIGP is enabled in the BGP configuration. However, the node does not originate its own AIGP attribute for local prefixes.

As the route is readvertised by downstream nodes, the cost of the AIGP attribute reflects the IGP distance to the prefix (zero + IGP distance or configured distance + IGP distance).

Options ***distance***—(Optional) Associate an initial cost when advertising a local prefix with the AIGP BGP attribute.

Range: 0 through 4,294,967,295

Default: The initial cost associated with the AIGP attribute for a local prefix is zero. The ***distance*** option overrides the default zero value for the initial cost.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring the Accumulated IGP Attribute for BGP*
- *aigp*

apply-path

Syntax apply-path *path*;

Hierarchy Level [edit logical-systems *logical-system-name* policy-options prefix-list *name*],
[edit policy-options prefix-list *name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Expand a prefix list to include all prefixes pointed to by a defined path.

Options *path*—String of elements composed of identifiers or configuration keywords that points to a set of prefixes. You can include wildcards (enclosed in angle brackets) to match more than one identifier. You cannot add a path element, including wildcards, after a leaf statement. Path elements, including wildcards, can only be used after a container statement.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring Prefix Lists on page 280
- Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List on page 687

as-path (Policy Options)

Syntax as-path *name regular-expression*;

Hierarchy Level [edit dynamic policy-options],
[edit logical-systems *logical-system-name* policy-options],
[edit policy-options]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.

Description Define an autonomous system (AS) path regular expression for use in a routing policy match condition.

Options *name*—Name that identifies the regular expression. The name can contain letters, numbers, and hyphens (-) and can be up to 65,536 characters long. To include spaces in the name, enclose it in quotation marks (" ").

regular-expression—One or more regular expressions used to match the AS path.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions on page 309
- **dynamic-db on page 1184**

as-path-group

Syntax

```
as-path-group group-name {
  as-path name regular-expression;
}
```

Hierarchy Level

[edit dynamic policy-options],
[edit logical-systems *logical-system-name* policy-options],
[edit policy-options]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for dynamic database configuration introduced in Junos OS Release 9.5.
Support for dynamic database configuration introduced in Junos OS Release 9.5 for EX Series switches.

Description

Define a group containing multiple AS path regular expressions for use in a routing policy match condition.

Options

group-name—Name that identifies the AS path group. One or more AS path regular expressions must be listed below the **as-path-group** hierarchy.

name—Name that identifies the regular expression. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (" ").

regular-expression—One or more regular expressions used to match the AS path.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding AS Path Regular Expressions for Use as Routing Policy Match Conditions on page 309
- **dynamic-db on page 1184**

backup-selection (Protocols OSPF or OSPF3)

Syntax

```
backup-selection {
  destination prefix {
    interface (interface-name| all){
      admin-group {
        exclude [ group-name ];
        include-all [ group-name ];
        include-any [ group-name ];
        preference [ group-name ];
      }
      bandwidth-greater-equal-primary;
      dest-metric (highest | lowest);
      downstream-paths-only;
      metric-order [ root dest ];
      node {
        exclude [ node-address ];
        preference [ node-address ];
      }
      protection-type (link | node | node-link);
      root-metric (highest | lowest);
      srlg (loose | strict);
      evaluation-order [ admin-group srlg bandwidth protection-type node metric ] ;
    }
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-options],
[edit logical-systems *logical-system-name* routing-instances *instance-name* routing-options],
[edit routing-instances *instance-name* routing-options],
[edit routing-options]

Release Information

Statement introduced in Junos OS Release 15.1.

Description

Define backup selection policies, per prefix per primary next-hop interface, to enforce loop-free alternate (LFA) selection based on admin-group, srlg, bandwidth, protection-type, node, and metric attributes of the backup path.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Configuring Backup Selection Policy for the OSPF or OSPF3 Protocol on page 159
- Configuring Backup Selection Policy for the OSPF Protocol on page 154
- Understanding Backup Selection Policy for OSPF Protocol on page 152

ccc (Routing Policy Condition)

Syntax

```
ccc {
  interface-name;
  standby;
  peer-unit unit-number;
  table table-name;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family],
[edit policy-options condition if-route-exists address-family],

Release Information

Statement introduced in Junos OS Release 13.2.

Description

Specify that the route must correspond to a CCC prefix to be considered a match.

Options

interface-name—Interface used to establish the CCC route.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring Pseudowire Redundancy in a Mobile Backhaul Scenario*

community (Policy Options)

Syntax

```
community name {
  invert-match;
  members [ community-ids ];
}
```

Hierarchy Level

[edit dynamic policy-options],
[edit logical-systems *logical-system-name* policy-options],
[edit policy-options]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.
Support for BGP large community introduced in Junos OS Release 17.3 for MX Series, PTX Series, and QFX Series.

Description

Define a community, extended community or large community for use in a routing policy match condition.

Options

name—Name that identifies the regular expression. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters. To include spaces in the name, enclose it in quotation marks (" ").

invert-match—Invert the results of the community expression matching. The **community** match condition defines a regular expression and if it matches the community attribute of the received prefix, Junos OS returns a TRUE result. If not, Junos OS returns a FALSE result. The **invert-match** statement makes Junos OS behave to the contrary. If there is a match, Junos OS returns a FALSE result. If there is no match, Junos OS returns a TRUE result.

members *community-ids*—One or more community members. If you specify more than one member, you must enclose all members in brackets.

The format for ***community-ids*** is:

as-number:*community-value*

Starting in Junos OS Release 15.1, you can apply a wildcard member **segmented-nh:.*:0** to apply the BGP policy to all the S-PMSI A-D routes carrying extended community information.

as-number is the AS number and can be a value in the range from 0 through 65,535. ***community-value*** is the community identifier and can be a number in the range from 0 through 65,535.

You also can specify ***community-ids*** for communities as one of the following well-known community names, which are defined in RFC 1997, *BGP Communities Attribute*:

- **no-export**—Routes containing this community name are not advertised outside a BGP confederation boundary.
- **no-advertise**—Routes containing this community name are not advertised to other BGP peers.
- **no-export-subconfed**—Routes containing this community name are not advertised to external BGP peers, including peers in other members' ASs inside a BGP confederation.

You can explicitly exclude BGP community information with a static route using the **none** option. Include **none** when configuring an individual route in the **route** portion of the **static** statement to override a **community** option specified in the **defaults** portion of the statement.

The format for extended ***community-ids*** is the following:

type:administrator:assigned-number

type is the type of extended community and can be either a **bandwidth**, **target**, **origin**, **domain-id**, **src-as**, or **rt-import** community or a 16-bit number that identifies a specific BGP extended community. The **target** community identifies the destination to which the route is going. The **origin** community identifies where the route originated. The **domain-id** community identifies the OSPF domain from which the route originated. The **src-as** community identifies the autonomous system from which the route originated. The **rt-import** community identifies the route to install in the routing table.

NOTE: For src-as, you can specify only an AS number and not an IP address. For rt-import, you can specify only an IP address and not an AS number.

administrator is the administrator. It is either an AS number or an IPv4 address prefix, depending on the type of extended community.

assigned-number identifies the local provider.

The format for linking a bandwidth with an AS number is:

bandwidth:*as-number:bandwidth*

as-number specifies the AS number and ***bandwidth*** specifies the bandwidth in bytes per second.

NOTE: In Junos OS Release 9.1 and later, you can specify 4-byte AS numbers as defined in RFC 4893, *BGP Support for Four-octet AS Number Space*, as well as the 2-byte AS numbers that are supported in earlier releases of the Junos OS. In plain-number format, you can configure a value in the range from 1 through 4,294,967,295. To configure a **target** or **origin** extended community that includes a 4-byte AS number in the plain-number format, append the letter "L" to the end of number. For example, a target community with the 4-byte AS number 334,324 and an assigned number of 132 is represented as **target:334324L:132**.

In Junos OS Release 9.2 and later, you can also use AS-dot notation when defining a 4-byte AS number for the **target** and **origin** extended communities. Specify two integers joined by a period: *16-bit high-order value in decimal*.*16-bit low-order value in decimal*. For example, the 4-byte AS number represented in plain-number format as 65546 is represented in AS-dot notation as 1.10.

As defined in RFC 8092, BGP large community uses 12-byte encoding and the format for BGP large ***community-ids*** is:

large: *global-administrator*:*assigned-number*:*assigned-number*

large indicates BGP large community.

global-administrator is the administrator. It is a 4-byte AS number.

assigned-number is a 4-byte value used to identify the local provider. BGP large community uses two 4-byte assigned number to identify the local provider.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding BGP Communities, Extended Communities, and Large Communities as Routing Policy Match Conditions on page 359
- Understanding How to Define BGP Communities and Extended Communities on page 361
- **dynamic-db on page 1184**

condition

Syntax

```
condition condition-name {
  dynamic-db;
  if-route-exists{
    address;
    address-family {
      inet {
        address;
        table table-name;
      }
      ccc {
        interface-name;
        standby;
        peer-unit unit-number;
        table table-name;
      }
    }
    table table-name;
  }
}
```

Hierarchy Level

[edit dynamic policy-options],
[edit logical-systems *logical-system-name* policy-options],
[edit policy-options]

Release Information

Statement introduced in Junos OS Release 9.0.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.
Support for the address families introduced in Junos OS Release 13.2.

Description

Define a policy condition based on the existence of routes in specific tables for use in BGP export policies.

Options

condition-name—Name of the condition.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Conditional Advertisement and Import Policy (Routing Table) with certain match conditions on page 490
- *Example: Configuring Pseudowire Redundancy in a Mobile Backhaul Scenario*
- dynamic-db on page 1184

damping (Policy Options)

Syntax

```
damping name {
  disable;
  half-life minutes;
  max-suppress minutes;
  reuse number;
  suppress number;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* policy-options],
[edit policy-options]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description

Define route flap damping properties to set on BGP routes.

Options

disable—Disable damping on a per-prefix basis. Any damping state that is present in the routing table for a prefix is deleted if damping is disabled.

half-life *minutes*—Decay half-life. ***minutes*** is the interval after which the accumulated figure-of-merit value is reduced by half if the route remains stable.

Range: 1 through 45

Default: 15 minutes

NOTE: For the half-life, configure a value that is less than the max-suppress. If you do not, the configuration is rejected.

max-suppress *minutes*—Maximum hold-down time. ***minutes*** is the maximum time that a route can be suppressed no matter how unstable it has been.

Range: 1 through 720

Default: 60 minutes

NOTE: For the max-suppress, configure a value that is greater than the half-life. If you do not, the configuration is rejected.

name—Name that identifies the set of damping parameters. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (" ").

reuse *number*—Reuse threshold. ***number*** is the figure-of-merit value below which a suppressed route can be used again.

Range: 1 through 20,000

Default: 750 (unitless)

suppress ***number***—Cutoff (suppression) threshold. ***number*** is the figure-of-merit value above which a route is suppressed for use or inclusion in advertisements.

Range: 1 through 20,000

Default: 3000 (unitless)

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Configuring BGP Flap Damping Parameters on page 425
- Example: Configuring BGP Route Flap Damping Parameters on page 430
- Example: Configuring BGP Route Flap Damping Based on the MBGP MVPN Address Family on page 439

decapsulate (Firewall Filter)

Syntax

```
decapsulate {
  gre {
    apply-groups;
    apply-groups-except;
    forwarding-class;
    interface-group(0 -255)
    no-decrement-ttl;
    routing-instance;
    sample;
  }
  gre-in-udp{
  l2tp {
    apply-groups;
    apply-groups-except;
    cookie;
    forwarding-class;
    no-decrement-ttl;
    output-interface;
    sample;
  }
}
```

Hierarchy Level [edit firewall family *family-name* filter *filter-name* term *term-name* then],

Release Information Statement introduced in Junos OS Release 7.6.
output-interface and **cookie** options introduced in Junos OS Release 15.1.
decapsulate gre introduced in Junos OS Release 15.1F3 and 16.1R2 for PTX5000 routers with third generation FPCs and Junos OS Release 15.1F6 and 16.1R2 for PTX3000 routers with third-generation FPCs.
no-decrement-ttl attribute for the **decapsulate gre** filter action introduced in Junos OS Release 15.1F6 and 16.2R1 for PTX5000 routers with third-generation FPCs.

Description Define the termination action for GRE and L2TP tunnels.

Options

gre—(Optional) Terminate a GRE tunnel for the filter conditions that are matched.

l2tp—(Optional) Terminate an L2TP tunnel for the filter conditions that are matched.

output-interface ***interface-name***—(Optional) For L2TP tunnels, enable the packet to be duplicated and sent towards the customer or the network (based on the MAC address in the Ethernet payload),

cookie ***l2tpv3-cookie***—(Optional) For L2TP tunnels, specify the L2TP cookie for the duplicated packets. If the tunnel does not contain the receive-cookie configured, packet injection does not happen. In such a case, any received tunnel packet is counted and dropped in the same manner in which packets that arrive with a wrong cookie are counted and dropped.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- *Configuring Multifield Classifiers*
- Guidelines for Configuring and Applying Firewall Filters in Logical Systems on page 788

defaults (Policy Options)

Syntax

```
defaults {
  route-filter (no-walkup | walkup);
}
```

Hierarchy Level

[edit logical-system *logical-system-name* policy-options],
[edit logical-system *logical-system-name* policy-options policy-statement
  policy-statement-name],
[edit policy-options],
[edit policy-options policy-statement *policy-statement-name*]

Release Information

Statement introduced in Junos OS Release 13.3 on ACX Series, EX 4600, M Series, MX Series, PTX Series, QFabric System, QFX Series standalone switches, and T Series platforms.

Description

Establish defaults for a particular policy statement or globally. Defaults include the walkup feature, which examines more than the longest match route filters in a policy statement term with more than one route filter, allowing consolidation of terms and a potential performance enhancement.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- **no-walkup on page 1224**
- **walkup on page 1242**
- **route-filter on page 1235**
- Walkup for Route Filters Overview on page 234
- Configuring Walkup for Route Filters to Improve Operational Efficiency on page 238
- Example: Configuring Walkup for Route Filters Globally to Improve Operational Efficiency on page 247
- Example: Configuring Walkup for Route Filters Locally to Improve Operational Efficiency on page 252

destination (Protocols OSPF or OSPF3)

Syntax

```
destination prefix {
  interface (interface-name |all) {
    admin-group {
      exclude [ group-name ];
      include-all [ group-name ];
      include-any [ group-name ];
      preference [ group-name ];
    }
    bandwidth-greater-equal-primary;
    dest-metric (highest | lowest);
    downstream-paths-only ;
    evaluation-order [ admin-group srlg bandwidth protection-type neighbor neighbor-tag
      metric ];
    metric-order [ root dest ];
    node {
      exclude [ node-address ];
      preference [ node-address ];
    }
    protection-type (link |node | node-link);
    root-metric (highest | lowest) ;
    srlg (loose |strict);
  }
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-options backup-selection],
[edit logical-systems *logical-system-name* routing-instances *instance-name* routing-options backup-selection],
[edit routing-instances *instance-name* routing-options backup-selection],
[edit routing-options backup-selection]

Release Information Statement introduced in Junos OS Release 15.1.

Description Define the backup selection policy for a particular destination prefix or for all the prefixes.

Options ***prefix***— Destination prefix name and prefix length. You can specify 0/0 for the IPv4 least-specific prefix or 0::0/0 for the IPv6 least-specific prefix.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

•

dynamic-db

Syntax dynamic-db;

Hierarchy Level
[edit logical-systems *logical-system-name* policy-options as-path *path-name*],
[edit logical-systems *logical-system-name* policy-options as-path-group *group-name*],
[edit logical-systems *logical-system-name* policy-options community *community-name*],
[edit logical-systems *logical-system-name* policy-options condition *condition-name*],
[edit logical-systems *logical-system-name* policy-options policy-statement *policy-statement-name*],
[edit logical-systems *logical-system-name* policy-options prefix-list *prefix-list-name*],
[edit policy-options as-path *path-name*],
[edit policy-options as-path-group *group-name*],
[edit policy-options community *community-name*],
[edit policy-options condition *condition-name*],
[edit policy-options policy-statement *policy-statement-name*],
[edit policy-options prefix-list *prefix-list-name*]

Release Information Statement introduced in Junos OS Release 9.5.
Statement introduced in Junos OS Release 9.5 for EX Series switches.

Description Define routing policies and policy objects that reference policies configured in the dynamic database at the **[edit dynamic]** hierarchy level.

Required Privilege Level routing—To view this statement in the configuration.
routing-control-level—To add this statement to the configuration.

Related Documentation
- Example: Configuring Dynamic Routing Policies on page 527

export (Protocols BGP)

Syntax export [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols bgp],
[edit logical-systems *logical-system-name* protocols bgp group *group-name*],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* neighbor *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp group *group-name* neighbor *address*],
[edit protocols bgp],
[edit protocols bgp group *group-name*],
[edit protocols bgp group *group-name* neighbor *address*],
[edit routing-instances *routing-instance-name* protocols bgp],
[edit routing-instances *routing-instance-name* protocols bgp group *group-name*],
[edit routing-instances *routing-instance-name* protocols bgp group *group-name* neighbor *address*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Apply one or more policies to routes being exported from the routing table into BGP.

If you specify more than one policy, they are evaluated in the order specified, from left to right, and the first matching filter is applied to the route. If no routes match the filters, the routing table exports into BGP only the routes that it learned from BGP. If an action specified in one of the policies manipulates a route characteristic, the policy framework software carries the new route characteristic forward during the evaluation of the remaining policies. For example, if the action specified in the first policy of a chain sets a route's metric to 500, this route matches the criterion of **metric 500** defined in the next policy.

Options ***policy-names***—Name of one or more policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Configuring Routing Policies to Control BGP Route Advertisements*
- *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*
- import on page 1197

export (Protocols DVMRP)

Syntax

export [*policy-names*];

Hierarchy Level

[edit logical-systems *logical-system-name* protocols dvmrp],
[edit protocols dvmrp]

Release Information

NOTE: Distance Vector Multicast Routing Protocol (DVMRP) was deprecated in Junos OS Release 16.1. Although DVMRP commands continue to be available and configurable in the CLI, they are no longer visible and are scheduled for removal in a subsequent release.

Statement introduced before Junos OS Release 7.4.

Description

Apply one or more policies to routes being exported from the routing table into DVMRP. If you specify more than one policy, they are evaluated in the order specified, from first to last, and the first matching policy is applied to the route. If no match is found, the routing table exports into DVMRP only the routes that it learned from DVMRP and direct routes.

Options

policy-names—Name of one or more policies.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- import on page 1199
- *Example: Configuring DVMRP to Announce Unicast Routes*

export

Syntax export [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols isis],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols isis],
[edit protocols isis],
[edit routing-instances *routing-instance-name* protocols isis]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 12.1 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Apply one or more policies to routes being exported from the routing table into IS-IS.

All routing protocols store the routes that they learn in the routing table. The routing table uses this collected route information to determine the active routes to destinations. The routing table then installs the active routes into its forwarding table and exports them into the routing protocols. It is these exported routes that the protocols advertise.

For each protocol, you control which routes the protocol stores in the routing table and which routes the routing table exports into the protocol from the routing table by defining a *routing policy* for that protocol.

NOTE: For IS-IS, you cannot apply routing policies that affect how routes are imported into the routing table; doing so with a link-state protocol can easily lead to an inconsistent topology database.

Options ***policy-names***—Name of one or more policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

export (Protocols LDP)

Syntax export [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols ldp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ldp],
[edit protocols ldp],
[edit routing-instances *routing-instance-name* protocols ldp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.3X50 for the QFX Series.

Description Apply policy filters to outbound LDP label bindings. Filters are applied to all label bindings from all neighbors.

Options ***policy-names***—Name of one or more routing policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- *Filtering Outbound LDP Label Bindings*

export (Protocols MSDP)

Syntax export [*policy-names*];

Hierarchy Level
[edit logical-systems *logical-system-name* protocols msdp],
[edit logical-systems *logical-system-name* protocols msdp group *group-name*],
[edit logical-systems *logical-system-name* protocols msdp group *group-name* peer *address*],
[edit logical-systems *logical-system-name* protocols msdp peer *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp group *group-name* peer *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp peer *address*],
[edit protocols msdp],
[edit protocols msdp group *group-name*],
[edit protocols msdp group *group-name* peer *address*],
[edit protocols msdp peer *address*],
[edit routing-instances *routing-instance-name* protocols msdp],
[edit routing-instances *routing-instance-name* protocols msdp group *group-name*],
[edit routing-instances *routing-instance-name* protocols msdp group *group-name* peer *address*],
[edit routing-instances *routing-instance-name* protocols msdp peer *address*]

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.1 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Apply one or more policies to routes being exported from the routing table into MSDP.

Options ***policy-names***—Name of one or more policies.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- *Example: Configuring MSDP in a Routing Instance*
- import on page 1201

export

Syntax

export [*policy-names*];

Hierarchy Level

[edit logical-systems *logical-system-name* protocols (ospf | ospf3)],
[edit logical-systems *logical-system-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit protocols (ospf | ospf3)],
[edit protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit routing-instances *routing-instance-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for the **realm** statement introduced in Junos OS Release 9.2.
Support for the **realm** statement introduced in Junos OS Release 9.2 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description

Apply one or more policies to routes being exported from the routing table into OSPF.

Options

policy-names—Name of one or more policies.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Understanding OSPF Routing Policy*
- *Import and Export Policies for Network Summaries Overview*
- import on page 1202
- import on page 1202

export (Protocols PIM)

Syntax

export [*policy-names*];

Hierarchy Level

[edit logical-systems *logical-system-name* protocols pim],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols pim],
[edit protocols pim],
[edit routing-instances *routing-instance-name* protocols pim]

Release Information

Statement introduced in Junos OS Release 9.6.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description

Apply one or more export policies to control outgoing PIM join and prune messages. PIM join and prune filters can be applied to PIM-SM and PIM-SSM messages. PIM join and prune filters cannot be applied to PIM-DM messages.

Required Privilege Level

view-level—To view this statement in the configuration.
control-level—To add this statement to the configuration.

Related Documentation

- *Filtering Outgoing PIM Join Messages*

export (Bootstrap)

Syntax export [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols pim rp bootstrap family (inet | inet6)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols pim rp bootstrap family (inet | inet6)],
[edit protocols pim rp bootstrap family (inet | inet6)],
[edit routing-instances *routing-instance-name* protocols pim rp bootstrap family (inet | inet6)]

Release Information Statement introduced in Junos OS Release 7.6.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Apply one or more export policies to control outgoing PIM bootstrap messages.

Options ***policy-names***—Name of one or more import policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Configuring PIM Bootstrap Properties for IPv4*
- *Configuring PIM Bootstrap Properties for IPv4 or IPv6*
- import (Protocols PIM Bootstrap) on page 1204

export

Syntax export [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols rip group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols rip group *group-name*],
[edit protocols rip group *group-name*],
[edit routing-instances *routing-instance-name* protocols rip group *group-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Apply a policy to routes being exported to the neighbors.

By default, RIP does not export routes it has learned to its neighbors. To enable RIP to export routes, apply one or more export policies.

If no routes match the policies, the local routing device does not export any routes to its neighbors. Export policies override any metric values determined through calculations involving the values configured with the **metric-in** and **metric-out** statements.

NOTE: The export policy on RIP does not support manipulating routing information of the next hop.

Options *policy-names*—Name of one or more policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- import on page 1205

export (Protocols RIPng)

Syntax

export [*policy-names*];

Hierarchy Level

[edit logical-systems *logical-system-name* protocols ripng group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ripng group *group-name*],
[edit protocols ripng group *group-name*],
[edit routing-instances *routing-instance-name* protocols ripng group *group-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for routing instances introduced in Junos OS Release 9.0.

Description

Apply a policy or list of policies to routes being exported to the neighbors.

By default, RIPng does not export routes it has learned to its neighbors. To have RIPng export routes, apply one or more export policies. To apply export policies and to filter routes being exported from the local routing device to its neighbors, include the **export** statement and list the name of the policy to be evaluated.

You can define one or more export policies. If no routes match the policies, the local routing device does not export any routes to its neighbors. Export policies override any metric values determined through calculations involving the values configured with the **metric-in** and **metric-out** statements.

Options

policy-names—Name of one or more policies.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- import on page 1206

export (Routing Options)

Syntax export [*policy-name*];

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options forwarding-table],
[edit logical-systems *logical-system-name* routing-options forwarding-table],
[edit routing-instances *routing-instance-name* routing-options forwarding-table],
[edit routing-options forwarding-table]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 12.3 for ACX Series routers.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Apply one or more policies to routes being exported from the routing table into the forwarding table.

In the **export** statement, list the name of the routing policy to be evaluated when routes are being exported from the routing table into the forwarding table. Only active routes are exported from the routing table.

You can reference the same routing policy one or more times in the same or a different **export** statement.

You can apply export policies to routes being exported from the routing table into the forwarding table for the following features:

- Per-packet load balancing
- Class of service (CoS)

Options ***policy-name***—Name of one or more policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- *Example: Load Balancing BGP Traffic*

if-route-exists

Syntax

```
if-route-exists {
  address;
  address-family {
    inet {
      address;
      table table-name;
    }
    ccc {
      interface-name;
      standby;
      peer-unit unit-number;
      table table-name;
    }
  }
  table table-name;
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* policy-options condition],
[edit policy-options condition],

Release Information

Statement introduced in Junos OS Release 13.2.

Description

Specify the route match conditions.

Options

(Optional) ***address***—Specify the IP address that the route must have to be considered a match.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring Pseudowire Redundancy in a Mobile Backhaul Scenario*
- Example: Configuring a Routing Policy for Conditional Advertisement Enabling Conditional Installation of Prefixes in a Routing Table on page 493

import

Syntax import [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols bgp],
[edit logical-systems *logical-system-name* protocols bgp group *group-name*],
[edit logical-systems *logical-system-name* protocols bgp group *group-name* neighbor *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols bgp group *group-name* neighbor *address*],
[edit protocols bgp],
[edit protocols bgp group *group-name*],
[edit protocols bgp group *group-name* neighbor *address*],
[edit routing-instances *routing-instance-name* protocols bgp],
[edit routing-instances *routing-instance-name* protocols bgp group *group-name*],
[edit routing-instances *routing-instance-name* protocols bgp group *group-name* neighbor *address*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Apply one or more routing policies to routes being imported into the Junos OS routing table from BGP.

If you specify more than one policy, they are evaluated in the order specified, from left to right, and the first matching filter is applied to the route. If no match is found, BGP places into the routing table only those routes that were learned from BGP routing devices. The policy framework software evaluates the routing policies in a chain sequentially. If an action specified in one of the policies manipulates a route characteristic, the policy framework software carries the new route characteristic forward during the evaluation of the remaining policies. For example, if the action specified in the first policy of a chain sets a route's metric to 500, this route matches the criterion of **metric 500** defined in the next policy.

It is also important to understand that in Junos OS, although an import policy (inbound route filter) might reject a route, not use it for traffic forwarding, and not include it in an advertisement to other peers, the router retains these routes as hidden routes. These hidden routes are not available for policy or routing purposes. However, they do occupy memory space on the router. A service provider filtering routes to control the amount of information being kept in memory and processed by a router might want the router to entirely drop the routes being rejected by the import policy.

Hidden routes can be viewed by using the **show route receive-protocol bgp *neighbor-address* hidden** command. The hidden routes can then be retained or dropped from the routing table by configuring the **keep all | none** statement at the **[edit protocols bgp]** or **[edit protocols bgp group *group-name*]** hierarchy level.

The rules of BGP route retention are as follows:

- By default, all routes learned from BGP are retained, except those where the AS path is looped. (The AS path includes the local AS.)
- By configuring the **keep all** statement, all routes learned from BGP are retained, even those with the local AS in the AS path.
- By configuring the **keep none** statement, all routes received are discarded. When this statement is configured and the inbound policy changes, Junos OS re-advertises all the routes advertised by the peer.

Options ***policy-names***—Name of one or more policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring BGP Interactions with IGPs*
- *Configuring Routing Policies to Control BGP Route Advertisements*
- *Understanding Routing Policies*
- export on page 1185

import (Protocols DVMRP)

Syntax
import [*policy-names*];

Hierarchy Level
[edit logical-systems *logical-system-name* protocols dvmrp],
[edit protocols dvmrp]

Release Information

NOTE: Distance Vector Multicast Routing Protocol (DVMRP) was deprecated in Junos OS Release 16.1. Although DVMRP commands continue to be available and configurable in the CLI, they are no longer visible and are scheduled for removal in a subsequent release.

Statement introduced before Junos OS Release 7.4.

Description
Apply one or more policies to routes being imported into the routing table from DVMRP. If you specify more than one policy, they are evaluated in the order specified, from first to last, and the first matching policy is applied to the route. If no match is found, DVMRP shares with the routing table only those routes that were learned from DVMRP routers.

Options
policy-names—Name of one or more policies.

Required Privilege Level
routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- export on page 1186
- *Example: Configuring DVMRP to Announce Unicast Routes*

import (Protocols LDP)

Syntax import [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols ldp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ldp],
[edit protocols ldp],
[edit routing-instances *routing-instance-name* protocols ldp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.3X50 for the QFX Series.

Description Apply policy filters to received LDP label bindings. Filters are applied to all label bindings from all neighbors.

Options ***policy-names***—Name of one or more routing policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- *Filtering Inbound LDP Label Bindings*

import (Protocols MSDP)

Syntax import [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols msdp],
[edit logical-systems *logical-system-name* protocols msdp group *group-name*],
[edit logical-systems *logical-system-name* protocols msdp group *group-name* peer *address*],
[edit logical-systems *logical-system-name* protocols msdp peer *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp group *group-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp group *group-name* peer *address*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols msdp peer *address*],
[edit protocols msdp],
[edit protocols msdp group *group-name*],
[edit protocols msdp group *group-name* peer *address*],
[edit protocols msdp peer *address*],
[edit routing-instances *routing-instance-name* protocols msdp],
[edit routing-instances *routing-instance-name* protocols msdp group *group-name*],
[edit routing-instances *routing-instance-name* protocols msdp group *group-name* peer *address*],
[edit routing-instances *routing-instance-name* protocols msdp peer *address*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.1 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Apply one or more policies to routes being imported into the routing table from MSDP.

Options ***policy-names***—Name of one or more policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring MSDP in a Routing Instance*
- export on page 1189

import

Syntax import [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols (ospf | ospf3)],
[edit logical-systems *logical-system-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit protocols (ospf | ospf3)],
[edit protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)],
[edit routing-instances *routing-instance-name* protocols (ospf | ospf3)],
[edit routing-instances *routing-instance-name* protocols ospf3 realm (ipv4-unicast | ipv4-multicast | ipv6-multicast)]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for the **realm** statement introduced in Junos OS Release 9.2.
Support for the **realm** statement introduced in Junos OS Release 9.2 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Filter OSPF routes from being added to the routing table.

Options ***policy-names***—Name of one or more policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Understanding OSPF Routing Policy*
- *Import and Export Policies for Network Summaries Overview*
- export on page 1190
- export on page 1190

import (Protocols PIM)

Syntax import [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols pim],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols pim],
[edit protocols pim],
[edit routing-instances *routing-instance-name* protocols pim]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Apply one or more policies to routes being imported into the routing table from PIM. Use the **import** statement to filter PIM join messages and prevent them from entering the network.

Options ***policy-names***—Name of one or more policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Filtering Incoming PIM Join Messages*

import (Protocols PIM Bootstrap)

Syntax

import [*policy-names*];

Hierarchy Level

[edit logical-systems *logical-system-name* protocols pim rp bootstrap (inet | inet6)],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols pim rp bootstrap (inet | inet6)],
[edit protocols pim rp bootstrap (inet | inet6)],
[edit routing-instances *routing-instance-name* protocols pim rp bootstrap (inet | inet6)]

Release Information

Statement introduced in Junos OS Release 7.6.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description

Apply one or more import policies to control incoming PIM bootstrap messages.

Options

policy-names—Name of one or more import policies.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Configuring PIM Bootstrap Properties for IPv4*
- *Configuring PIM Bootstrap Properties for IPv4 or IPv6*
- export (Bootstrap) on page 1192

import (Protocols RIP)

Syntax

import [*policy-names*];

Hierarchy Level

[edit logical-systems *logical-system-name* protocols rip],
[edit logical-systems *logical-system-name* protocols rip group *group-name* neighbor *neighbor-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols rip],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols rip group *group-name* neighbor *neighbor-name*],
[edit protocols rip],
[edit protocols rip group *group-name* neighbor *neighbor-name*],
[edit routing-instances *routing-instance-name* protocols rip],
[edit routing-instances *routing-instance-name* protocols rip group *group-name* neighbor *neighbor-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 12.1 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description

Apply one or more policies to routes being imported by the local routing device from neighbors.

Options

policy-names—Name of one or more policies.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Applying Policies to RIP Routes Imported from Neighbors*
- Junos OS Routing Policies, Firewall Filters, and Traffic Policers Feature Guide for Routing Devices
- export on page 1193

import (Protocols RIPng)

Syntax import [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* protocols ripng],
[edit logical-systems *logical-system-name* protocols ripng group *group-name* neighbor *neighbor-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ripng],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* protocols ripng group *group-name* neighbor *neighbor-name*],
[edit protocols ripng],
[edit protocols ripng group *group-name* neighbor *neighbor-name*],
[edit routing-instances *routing-instance-name* protocols ripng],
[edit routing-instances *routing-instance-name* protocols ripng group *group-name* neighbor *neighbor-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for routing instances introduced in Junos OS Release 9.0.

Description Apply one or more policies to routes being imported into the local routing device from its neighbors.

Options ***policy-names***—Name of one or more policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Applying Policies to RIPng Routes Imported from Neighbors*
- export on page 1194

import

Syntax import [*policy-names*];

Hierarchy Level [edit logical-systems *logical-system-name* routing-instances *routing-instance-name* routing-options resolution rib],
[edit logical-systems *logical-system-name* routing-options resolution rib],
[edit routing-instances *routing-instance-name* routing-options resolution rib],
[edit routing-options resolution rib]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
Statement introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Specify one or more import policies to use for route resolution.

Options ***policy-names***—Name of one or more import policies.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- *Example: Configuring Route Resolution on PE Routers*

ingress-queuing-filter

Syntax ingress-queuing-filter *filter-name*;

Hierarchy Level [edit interfaces *interface-name* unit *unit-number* family *family-name*],
[edit logical systems *logical-system-name* interfaces *interface-name* unit *unit-number* family *family-name*]

Release Information Statement introduced in Junos OS Release 16.1 for MX Series routers with MPCs.

Description Use the **ingress-queuing-filter** statement to set the packet loss priority and forwarding class for the packet, or drop the packet prior to input queue selection. This assists in traffic shaping.

The **ingress-queuing-filter** statement is available only for the following protocol families: **bridge, ccc, inet, inet6, mpls,** and **vpls**.

ingress-queuing-filter takes ***filter-name*** as an argument. The named filter is a normal firewall filter that must be configured with at least one of the following actions: **accept, discard, forwarding-class,** and **loss-priority**.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Example: Configuring a Filter for Use as an Ingress Queuing Filter on page 741

inet (Routing Policy Condition)

Syntax

inet {
  address;
  table *table-name*;
}

Hierarchy Level

[edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family],
[edit policy-options condition if-route-exists address-family],

Release Information

Statement introduced in Junos OS Release 13.2.

Description

Specify that the route must correspond to a IPv4 prefix to be considered a match.

Options

(Optional) ***address***—Specify the IP address that the route must have to be considered a match.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring Pseudowire Redundancy in a Mobile Backhaul Scenario*

instance-shared

Syntax instance-shared;

Hierarchy Level [edit firewall family *protocol-family-name* filter *filter-name*],
[edit logical systems *logical-system-name* firewall family *protocol-family-name* filter *filter-name*]

Release Information Statement introduced in Junos OS Release 14.2.

Description Specify to share the firewall filter across multiple routing instances. By default, firewall filters are not automatically shared across multiple instances. You can configure both shared and nonshared firewall filters on the same routing device. This statement can be used only when network services for the device are configured with enhanced IP mode.

The following protocol families are supported: Bridge, IPv4, IPv6, Layer 2 CCC, MPLS, and VPLS.

NOTE: Only Modular Port Concentrators (MPCs) are supported.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- *network-services*

interface (Protocols ISIS)

Syntax

```
interface (interface-name | all) {
  admin-group {
    exclude [ group-name ];
    include-all [ group-name ];
    include-any [ group-name ];
    preference [ group-name ];
  }
  bandwidth-greater-equal-primary;
  dest-metric (highest | lowest);
  downstream-paths-only ;
  evaluation-order [ admin-group srlg bandwidth protection-type neighbor neighbor-tag
    metric ];
  metric-order [ root dest ];
  node {
    exclude [ neighbor-address ];
    preference [ neighbor-address ];
  }
  node-tag {
    exclude [ route-tag ];
    preference [ route-tag ];
  }
  protection-type (link | node| node-link);
  root-metric (highest | lowest);
  srlg (loose |strict);
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-options backup-selection destination *prefix*],
[edit logical-systems *logical-system-name* routing-instances *instance-name* routing-options backup-selection destination *prefix*],
[edit routing-instances *instance-name* routing-options backup-selection *prefix*],
[edit routing-options backup-selection destination *prefix*]

Release Information

Statement introduced in Junos OS Release 14.1.

Description

Define the backup selection policy for a specific primary next hop.

Options

interface-name— Name of the primary next-hop interface.

all— All the interfaces.

bandwidth-greater-equal-primary— Allow the selection of the backup next hop only if the bandwidth is greater than or equal to the bandwidth of the primary next hop.

dest-metric (highest | lowest)—Specifiy the metric from the one-hop neighbor or from the remote router such as an RSVP backup label-switched-path (LSP) tail-end router to the final destination.

highest— Select the backup path that has the highest destination metric.

lowest— Select the backup path that has the lowest destination metric.

downstream-paths-only— Select the backup path that is a downstream path to the destination.

evaluation-order [admin-group srlg bandwidth protection-type neighbor neighbor-tag metric]—Control the order and the criteria of evaluating the backup path. The default order of evaluation is admin-group, srlg, bandwidth, protection-type, neighbor, neighbor-tag, and metric.

NOTE: For the explicitly configured evaluation order, only the listed attributes influence the selection of the backup path.

metric-order [root dest]— Specify the order of preference of the root and the destination metric during the backup path selection. The preference order can be:

- [**root dest**] — Backup path selection or preference is first based on the root-metric criteria. If the criteria of all the root-metric is the same, then the selection or preference is based on the dest-metric.
- [**dest root**] — Backup path selection or preference is first based on the dest-metric criteria. If the criteria of all the dest-metric is the same, then the selection is based on the root-metric.

NOTE: **By default, backup paths with lower destination metric criteria are selected or preferred. If the criteria is the same, then the lowest root metric criteria is preferred or selected.**

root— The metric to a one-hop neighbor or a remote router.

dest— The metric from a one-hop neighbor or remote router to the final destination.

protection-type (link | node | node-link)—Specify the required protection type of the backup path.

NOTE: **If no protection-type is configured, then by default the first best path that matches all the other criteria is executed.**

link— Select the backup path that provides link protection.

node— Select the backup path that provides node protection.

node-link— Allow either node or link protection LFA where node-protection LFA is preferred over link-protection LFA.

root-metric (highest | lowest)—Specify the metric to the one-hop neighbor or to the remote router such as an RSVP backup label-switched-path (LSP) tail-end router.

highest— Select the highest root metric.

lowest— Select the lowest root metric.

srlg (loose | strict)—Define the backup selection to either allow or reject the common shared risk link groups (SRLGs) between the primary link and any link in the backup path.

loose— Allow the backup path that has common srlgs between the primary link and any link in the backup path. A backup path with a fewer number of srlg collisions is preferred.

strict— Reject the backup path that has common srlgs between the primary link and each link in the backup path.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Understanding Backup Selection Policy for IS-IS Protocol*
- *Configuring Backup Selection Policy for IS-IS Protocol*
- *backup-selection (Protocols ISIS)*
- *destination*

interface (Protocols OSPF or OSPF3)

Syntax

```
interface (interface-name | all) {
  admin-group {
    exclude [ group-name ];
    include-all [ group-name ];
    include-any [ group-name ];
    preference [ group-name ];
  }
  bandwidth-greater-equal-primary;
  dest-metric (highest | lowest);
  downstream-paths-only ;
  evaluation-order [ admin-group srlg bandwidth protection-type node metric ];
  metric-order [ root dest ];
  node {
    exclude [ node-address ];
    preference [ node-address ];
  }
  protection-type (link | node| node-link);
  root-metric (highest | lowest);
  srlg (loose |strict);
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-options backup-selection destination *prefix*],
[edit logical-systems *logical-system-name* routing-instances *instance-name* routing-options backup-selection destination *prefix*],
[edit routing-instances *instance-name* routing-options backup-selection *prefix*],
[edit routing-options backup-selection destination *prefix*]

Release Information

Statement introduced in Junos OS Release 15.1.

Description

Define the backup selection policy for a specific primary next hop.

Options

interface-name— Name of the primary next-hop interface.

all— All the interfaces.

bandwidth-greater-equal-primary— Allow the selection of the backup next hop only if the bandwidth is greater than or equal to the bandwidth of the primary next hop.

dest-metric (highest lowest)—Specifiy the metric from the one-hop neighbor or from the remote router such as an RSVP backup label-switched-path (LSP) tail-end router to the final destination.

> **highest**— Select the backup path that has the highest destination metric.
>
> **lowest**— Select the backup path that has the lowest destination metric.

downstream-paths-only— Select the backup path that is a downstream path to the destination.

evaluation-order [admin-group srlg bandwidth protection-type node metric]—Control the order and the criteria of evaluating the backup path. The default order of evaluation is admin-group, srlg, bandwidth, protection-type, node and metric.

NOTE: For the explicitly configured evaluation order, only the listed attributes influence the selection of the backup path.

metric-order [root dest]— Specify the order of preference of the root and the destination metric during the backup path selection. The preference order can be:

- [**root dest**] — Backup path selection or preference is first based on the root-metric criteria. If the criteria of all the root-metric is the same, then the selection or preference is based on the dest-metric.
- [**dest root**] — Backup path selection or preference is first based on the dest-metric criteria. If the criteria of all the dest-metric is the same, then the selection is based on the root-metric.

NOTE: Backup path selection or preference is first based on the dest-metric criteria. If the criteria of all the dest-metric is the same, then the selection is based on the root-metric. By default, backup paths with lower destination metric criteria are selected or preferred. If the criteria is the same, then the lowest root metric criteria is preferred or selected.

root— The metric to a one-hop neighbor or a remote router.

dest— The metric from a one-hop neighbor or remote router to the final destination.

protection-type (link | node | node-link)—Specify the required protection type of the backup path.

NOTE: If no protection-type is configured, then by default the first best path that matches all the other criteria is executed.

link— Select the backup path that provides link protection.

node— Select the backup path that provides node protection.

node-link— Allow either node or link protection LFA where node-protection LFA is preferred over link-protection LFA.

root-metric (highest lowest)—Specify the metric to the one-hop neighbor or to the remote router such as an RSVP backup label-switched-path (LSP) tail-end router.

highest— Select the highest root metric.

lowest— Select the lowest root metric.

srlg (loose | strict)—Define the backup selection to either allow or reject the common shared risk link groups (SRLGs) between the primary link and any link in the backup path.

loose— Allow the backup path that has common srlgs between the primary link and any link in the backup path. A backup path with a fewer number of srlg collisions is preferred.

strict— Reject the backup path that has common srlgs between the primary next-hop link and each link in the backup path.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Configuring Backup Selection Policy for the OSPF or OSPF3 Protocol on page 159
- Configuring Backup Selection Policy for the OSPF Protocol on page 154
- Understanding Backup Selection Policy for OSPF Protocol on page 152
- *backup-selection (Protocols ISIS)*

ip-options-protocol-queue

Syntax

```
ip-options-protocol-queue protocol-name {
  protocol-id protocol-id;
  queue-depth queue-depth;
}
```

Hierarchy Level [edit forwarding-options]

Release Information Statement introduced in Junos OS Release 16.1 for M Series, MX Series, PTX Series, and T Series.

Description Configure logical queue-depth in the PFE for ip-options packets for a given protocol such as TCP, UDP, ICMP, and so on, except IGMP. The queue-depth indicates the number of ip-otions packets that can be enqueued in the PFE logical queue, beyond which it will start dropping the packets. Currently, IGMP has a default queue-depth of 192 (which is not configurable), and other protocols have a cumulative default queue-depth of 25. The CLI supports configuration for a maximum of 16 protocols. The sum total of queue-depth for all the protocols should not exceed 1024 packets.

Options **protocol-id *protocol-id***—Identify the protocol.

Range: 1 through 254

queue-depth *queue-depth*—Size of the protocol logical options queue for a given protocol.

Range: 1 through 807 packets

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

metric (Policy Action)

Syntax metric (add *add* | aigp | expression | igp *metric_offset* | minimum-igp *metric_offset*| subtract *subtract*)

Hierarchy Level [edit policy-options policy-statement *policy-name* term *term-name* then],
[edit policy-options policy-statement *policy-name* then]

Release Information Statement introduced in Junos OS Release 16.1.

Description Specify this CLI policy action in an import or export policy to set the metric value to one of the following options as per your network requirement.

Options

add ***add***—Specify a constant that should be added to the metric value. Use this option if you want to increase the metric of a route.

expression—Calculate the metric value based on the route metric and metric2.

igp ***metric_offset***—Configure this option to set the IGP metric for BGP. Specify a metric offset to increase or decrease the calculated IGP metric.

minimum-igp ***metric_offset***—Configure this option to set the minimum IGP metric for BGP.

aigp—Configure this option to set IGP metric to the accumulated interior gateway protocol (AIGP) metric value if an AIGP attribute has been configured. Specify this value in an export policy to redistribute BGP routes to an IGP, such as the OSPF protocol.

subtract ***subtract***—Specify a constant that must be subtracted from the metric value. Use this option if you want to decrease the metric of a route.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- aigp-adjust (Policy Action) on page 1168
- show policy on page 1364

node

Syntax

```
node {
  exclude [ node-address ];
  preference [ node-address ];
}
```

Hierarchy Level

[edit logical systems *logical-system-name* routing-options backup-selection (Protocols ISIS) destination *prefix* interface *interface name*],
[edit logical-systems *logical-system-name* routing-instances *instance-name* routing-options backup-selection destination *prefix* interface *interface-name*],
[edit routing-instances *instance-name* routing-options backup-selection destination *prefix* interface *interface-name*],
[edit routing-options backup-selection (Protocols ISIS) destination *prefix* interface *interface name*]

Release Information

Statement introduced in Junos OS Release 14.1.

Description

Define a list of loop-back IP addresses of the adjacent nodes to either prefer or exclude in the backup path selection. The node can be a local (adjacent router) node, remote node, or any other router in the backup path.

NOTE: The nodes are identified through the TE-router-ID TLV advertised by a node in the LSP.

Options

exclude [*node-address*]— Specify the list of nodes to be excluded. The backup path that has a router from the list is not selected as the loop-free alternative or backup next hop.

node-address— Name of one or more nodes to be excluded during backup path selection.

preference [*node-address*]— Define an ordered set of nodes to be preferred. The backup path having the leftmost node is selected.

node-address— Name of one or more nodes to be preferred in the backup path selection.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Understanding Backup Selection Policy for IS-IS Protocol*
- *Configuring Backup Selection Policy for IS-IS Protocol*
- *backup-selection (Protocols ISIS)*

- *destination*
- interface on page 1211

node-tag

Syntax

```
node-tag {
  exclude [ route-tag ];
  preference [ route-tag ];
}
```

Hierarchy Level

[edit logical-systems *logical-system-name* routing-options backup-selection (Protocols ISIS) destination *prefix* interface *interface name*],
[edit logical-systems *logical-system-name* routing-instances *instance-name* routing-options backup-selection destination *prefix* interface *interface-name*],
[edit routing-instances *instance-name* routing-options backup-selection destination *prefix* interface *interface-name*],
[edit routing-options backup-selection destination *prefix* interface *interface name*]

Release Information

Statement introduced in Junos OS Release 14.1.

Description

Define per-neighbor policy to either prefer or exclude a backup path.

NOTE: This statement identifies a group of nodes in the network based on criteria such as the same neighbor tag values for all PE nodes. This is implemented using IS-IS admin-tags.

Options

exclude [*route-tag*]— Specify that the backup path which has any node or router with route-tag from this list is not selected as the loop-free alternative or backup-next hop.

route-tag— Name of one or more tags advertised as part of extended IP reachability with a /32 prefix length that represents the TE-router-ID or node ID of a router.

preference [*route-tag*]— Specify the set of route tags in descending order of preference.

route-tag— Name of one or more tags advertised as part of extended IP reachability with a /32 prefix length that represents the TE-router-ID or node ID of a router.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Understanding Backup Selection Policy for IS-IS Protocol*
- *Configuring Backup Selection Policy for IS-IS Protocol*
- *backup-selection (Protocols ISIS)*
- *destination*
- interface on page 1211

no-walkup

Syntax no-walkup;

Hierarchy Level [edit logical-system *logical-system-name* policy-options defaults route-filter],
[edit logical-system *logical-system-name* policy-options policy-statement *policy-statement-name* defaults route-filter]

Release Information Statement introduced in Junos OS Release 13.3 on ACX Series, EX 4600, M Series, MX Series, PTX Series, QFabric System, QFX Series standalone switches, and T Series platforms.

Description Override route filter walkup globally or locally for a particular policy statement. The walkup feature examines more than the longest match route filters in a policy statement term with more than one route filter, allowing consolidation of terms and a potential performance enhancement.

Default By default, the policy statement performs the type of route filter processing that is enabled at the global level.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- **walkup on page 1242**
- **route-filter on page 1235**
- Walkup for Route Filters Overview on page 234
- Configuring Walkup for Route Filters to Improve Operational Efficiency on page 238
- Example: Configuring Walkup for Route Filters Globally to Improve Operational Efficiency on page 247
- Example: Configuring Walkup for Route Filters Locally to Improve Operational Efficiency on page 252

peer-unit (Routing Policy Condition)

Syntax peer-unit *unit-number*;

Hierarchy Level [edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family ccc],
[edit policy-options condition if-route-exists address-family ccc],

Release Information Statement introduced in Junos OS Release 13.2.

Description Specify the associated logical tunnel interface's peer-unit. This is required for logical-tunnel-based routes.

Options ***unit-number***—Logical unit number of the logical tunnel peer interface.

Range: 0 through 8192

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring Pseudowire Redundancy in a Mobile Backhaul Scenario*

policy-options

Syntax

```
policy-options {
  as-path name regular-expression;
  as-path-group group-name;
  community name {
    invert-match;
    members [ community-ids ];
  }
  condition condition-name {
    if-route-exists address table table-name;
  }
  damping name {
    disable;
    half-life minutes;
    max-suppress minutes;
    reuse number;
    suppress number;
  }
  policy-statement policy-name {
    term term-name {
      from {
        family;
        fpc-pfes-offline pfes-offline-per-fpc;
        match-conditions;
        policy subroutine-policy-name;
        prefix-list name;
        route-filter destination-prefix match-type <actions>;
        source-address-filter source-prefix match-type <actions>;
      }
      to {
        match-conditions;
        policy subroutine-policy-name;
      }
      then actions;
      default-action (accept | reject);
      prefix-segment {
        index index;
        node-segment;
      }
    }
    then {
      no-entropy-label-capability;
      priority (high | medium | low);
    }
  }
  prefix-list name {
    ip-addresses;
  }
}
```

Hierarchy Level

[edit],
[edit dynamic-profiles *profile-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Support at the **[edit dynamic-profiles]** hierarchy level introduced in Junos OS Release 11.4.

Description Configure routing policy.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Example: Using Routing Policy in an ISP Network on page 98

policy-statement

Syntax

```
policy-statement policy-name {
  term term-name {
    from {
      as-path-unique-count count (equal | orhigher | orlower);
      family family-name;
      match-conditions;
      policy subroutine-policy-name;
      prefix-list prefix-list-name;
      prefix-list-filter prefix-list-name match-type <actions>;
      protocol protocol-name;
      route-filter destination-prefix match-type <actions>;
      source-address-filter source-prefix match-type <actions>;
      tag value;
      traffic-engineering;
    }
    to {
      match-conditions;
      policy subroutine-policy-name;
    }
    then actions;
  }
  then {
    dynamic-tunnel-attributes dynamic-tunnel-attributes
    multipath-resolve multipath-resolve;
    no-entropy-label-capability;
    prefix-segment {
      index index;
      node-segment;
    }
    priority (high | medium | low);
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* policy-options],
[edit logical-systems *logical-system-name* policy-options],
[edit policy-options]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.
inet-mdt option introduced in Junos OS Release 10.0R2.
Statement introduced in Junos OS Release 11.3 for the QFX Series.
route-target option introduced in Junos OS Release 12.2.
Statement introduced in Junos OS 14.1X53-D20 for the OCX Series.
protocol and **traffic-engineering** options introduced in Junos OS Release 14.2.
no-entropy-label-capability option introduced in Junos OS Release 15.1.
priority and **tag *value*** options introduced in Junos OS Release 17.1.

as-path-unique-count option introduced in Junos OS Release 17.2R1.
prefix-segment option introduced in Junos OS Release 17.2R1 for MX Series routers, PTX Series routers, QFX5100 switches, and QFX10000 switches.
multipath-resolve and **dynamic-tunnel-attributes** options introduced in Junos OS Release 17.3R1.

Description Define a routing policy, including subroutine policies.

A *term* is a named structure in which match conditions and actions are defined. Routing policies are made up of one or more terms. Each routing policy term is identified by a term name. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose the entire name in double quotation marks.

Each term contains a set of match conditions and a set of actions:

- Match conditions are criteria that a route must match before the actions can be applied. If a route matches all criteria, one or more actions are applied to the route.
- Actions specify whether to accept or reject the route, control how a series of policies are evaluated, and manipulate the characteristics associated with a route.

Generally, a router compares a route against the match conditions of each term in a routing policy, starting with the first and moving through the terms in the order in which they are defined, until a match is made and an explicitly configured or default action of **accept** or **reject** is taken. If none of the terms in the policy match the route, the router compares the route against the next policy, and so on, until either an action is taken or the default policy is evaluated.

If none of the match conditions of each term evaluates to true, the final action is executed. The final action is defined in an unnamed term. Additionally, you can define a default action (either **accept** or **reject**) that overrides any action intrinsic to the protocol.

The order of match conditions in a term is not relevant, because a route must match all match conditions in a term for an action to be taken.

To list the routing policies under the **[edit policy-options]** hierarchy level by **policy-statement** ***policy-name*** in alphabetical order, enter the **show policy-options** configuration command.

The statements are explained separately.

Options ***actions***—(Optional) One or more actions to take if the conditions match. The actions are described in "Configuring Flow Control Actions" on page 56.

family ***family-name***—(Optional) Specify an address family protocol. Specify **inet** for IPv4. Specify **inet6** for 128-bit IPv6, and to enable interpretation of IPv6 router filter addresses. For IS-IS traffic, specify **iso**. For IPv4 multicast VPN traffic, specify **inet-mvpn**. For IPv6 multicast VPN traffic, specify **inet6-mvpn**. For multicast-distribution-tree (MDT) IPv4 traffic, specify **inet-mdt**. For BGP route target VPN traffic, specify **route-target**. For traffic engineering, specify **traffic-engineering**.

NOTE: When **family** is not specified, the routing device or routing instance uses the address family or families carried by BGP. If multiprotocol BGP (MP-BGP) is enabled, the policy defaults to the protocol family or families carried in the network layer reachability information (NLRI) as configured in the *family* statement for BGP. If MP-BGP is not enabled, the policy uses the default BGP address family unicast IPv4.

from—(Optional) Match a route based on its source address.

as-path-unique-count ***count*** **(equal | orhigher | orlower)**—(Optional) Specify a number from 0 through 1024 to filter routes based on the number of unique autonomous systems (ASs) in the AS path. Specify the match condition for the unique AS path count .

dynamic-tunnel-attributes ***dynamic-tunnel-attributes***—(Optional) Choose a set of defined dynamic tunnel attributes for forwarding traffic over V4oV6 tunnels.

match-conditions—(Optional in **from** statement; required in **to** statement) One or more conditions to use to make a match. The qualifiers are described in "Routing Policy Match Conditions" on page 44.

multipath-resolve ***multipath-resolve***–(Optional) Enable the use of all paths for resolution over the specified prefix.

no-entropy-label-capability—(Optional) Disable the entropy label capability advertisement at egress or transit routes specified in the policy.

priority (high | medium | low)—(Optional) Configure the priority for an IS-IS route to change the default order in which the routes are installed in the routing table, in the event of a network topology change.

policy ***subroutine-policy-name***—Use another policy as a match condition within this policy. The name identifying the subroutine policy can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (" "). Policy names cannot take the form **__.*-internal__**, as this form is reserved. For information about how to configure subroutines, see "Understanding Policy Subroutines in Routing Policy Match Conditions" on page 198.

policy-name—Name that identifies the policy. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (" ").

prefix-list ***prefix-list-name***—Name of a list of IPv4 or IPv6 prefixes.

prefix-list-filter ***prefix-list-name***—Name of a prefix list to evaluate using qualifiers; ***match-type*** is the type of match (see "Configuring Prefix List Filters" on page 213), and ***actions*** is the action to take if the prefixes match.

protocol ***protocol-name***—Name of the protocol used to control traffic engineering database import at the originating point.

route-filter ***destination-prefix match-type <actions>***—(Optional) List of routes on which to perform an immediate match; ***destination-prefix*** is the IPv4 or IPv6 route prefix to match, ***match-type*** is the type of match (see "Configuring Route Filters" on page 216), and ***actions*** is the action to take if the ***destination-prefix*** matches.

source-address-filter ***source-prefix match-type <actions>***—(Optional) Unicast source addresses in multiprotocol BGP (MBGP) and Multicast Source Discovery Protocol (MSDP) environments on which to perform an immediate match. ***source-prefix*** is the IPv4 or IPv6 route prefix to match, ***match-type*** is the type of match (see "Configuring Route Filters" on page 216), and ***actions*** is the action to take if the ***source-prefix*** matches.

tag ***value***—(Optional) A numeric value that identifies a route. You can tag certain routes to prioritize them over other routes. In the event of a network topology change, Junos OS updates these routes in the routing table before updating other routes with lower priority. You can also tag some routes to identify and reject them based on your requirement.

term ***term-name***—Name that identifies the term. The term name must be unique in the policy. It can contain letters, numbers, and hyphens (-) and can be up to 64 characters long. To include spaces in the name, enclose the entire name in quotation marks (" "). A policy statement can include multiple terms. We recommend that you name all terms. However, you do have the option to include an unnamed term which must be the final term in the policy. To configure an unnamed term, omit the **term** statement when defining match conditions and actions.

to—(Optional) Match a route based on its destination address or the protocols into which the route is being advertised.

then—(Optional) Actions to take on matching routes. The actions are described in "Configuring Flow Control Actions" on page 56 and "Configuring Actions That Manipulate Route Characteristics" on page 57.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- dynamic-db on page 1184
- *Understanding Source Packet Routing in Networking (SPRING)*

prefix-list

Syntax

```
prefix-list name {
  ip-addresses;
  apply-path path;
}
```

Hierarchy Level

[edit dynamic policy-options],
[edit logical-systems *logical-system-name* policy-options],
[edit policy-options]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.
Support for the **vpls** protocol family introduced in Junos OS Release 10.2.
Support for IPv6 RA guard policy lists introduced in Junos OS Release 16.1 for EX Series switches.

Description

Define a list of IPv4 or IPv6 address prefixes for use in a routing policy statement or firewall filter statement, or a list of IPv6 addresses or address prefixes for use in an IPv6 RA guard policy.

You can configure up to 85,325 prefixes in each prefix list. To configure more than 85,325 prefixes, configure multiple prefix lists and apply them to multiple firewall filter terms.

Options

name—Name that identifies the list of IPv4 or IPv6 addresses or address prefixes.

ip-addresses—List of IPv4 or IPv6 addresses or address prefixes, one IP address per line in the configuration.

The remaining statement is explained separately. See CLI Explorer.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Prefix Lists for Use in Routing Policy Match Conditions on page 279
- **dynamic-db on page 1184**
- Firewall Filter Match Conditions Based on Address Fields on page 606 in the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*
- Example: Configuring a Filter to Limit TCP Access to a Port Based On a Prefix List on page 687 in the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*

prefix-list-filter

Syntax prefix-list-filter *prefix-list-name match-type* <*actions*>;

Hierarchy Level [edit logical-systems *logical-system-name* policy-options policy-statement *policy-statement-name* term *term--name* from],
[edit policy-options policy-statement *policy-statement-name* term *term--name* from]

Release Information Statement introduced before Junos OS Release 7.4.

Description Evaluate a list of prefixes within a prefix list using specified qualifiers.

Options ***prefix-list-name***—Name of the prefix list to evaluate.

exact—The prefix-length component of the match prefix is equal to the route's prefix length

longer—The route's prefix length is greater than the prefix-length component of the match prefix.

orlonger—The route's prefix length is equal to or greater than the prefix-length component of the configured match prefix.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Understanding Prefix Lists for Use in Routing Policy Match Conditions on page 279

route-filter

Syntax route-filter (no-walkup | walkup);

Hierarchy Level [edit logical-system *logical-system-name* policy-options defaults],
[edit logical-system *logical-system-name* policy-options policy-statement *policy-statement-name* defaults],
[edit policy-options defaults],
[edit policy-options policy-statement *policy-statement-name* defaults]

Release Information Statement introduced in Junos OS Release 13.3 on ACX Series, EX 4600, M Series, MX Series, PTX Series, QFabric System, QFX Series standalone switches, and T Series platforms.

Description Enable or disable walkup globally or locally for route filters in a particular policy statement or globally. The walkup feature examines more than the longest match route filters in a policy statement term with more than one route filter, allowing consolidation of terms and a potential performance enhancement.

Default By default, no route filter walkup is performed.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- no-walkup on page 1224
- walkup on page 1242
- Walkup for Route Filters Overview on page 234
- Configuring Walkup for Route Filters to Improve Operational Efficiency on page 238
- Example: Configuring Walkup for Route Filters Globally to Improve Operational Efficiency on page 247
- Example: Configuring Walkup for Route Filters Locally to Improve Operational Efficiency on page 252

route-filter-list

List of Syntax

route-filter-list (Create List) on page 1236
route-filter-list (Use list) on page 1236

route-filter-list (Create List)

route-filter-list *route-filter-list-name*;

Hierarchy Level

[edit logical-systems *logical-system-name* policy-options],
[edit policy-options]

route-filter-list (Use list)

route-filter-list *route-filter-list-name*;

Hierarchy Level

[edit logical-systems *logical-system-name* policy-statement *policy-statement-name* term *term-name* from],
[edit policy-options policy-statement *policy-statement-name* term *term-name* from]

Release Information

Statement first introduced in Junos OS Release 16.1 on M Series, MX Series, and T Series.

Description

The route filter list is a user-configured list of individual route filters that you create at the **[edit policy-options]** hierarchy level. Each item in the list consists of a complete route filter statement, made up of a destination prefix, a match type, and an optional action.

For example:

```
[edit]
user@router# show policy-options route-filter-list rf-test-list
203.0.113.0/24 address-mask 255.255.255.0;
192.0.2.0/26 orlonger reject;
198.51.100.8/29 exact accept;
```

Once configured, the **route-filter-list** is used by referencing its ***route-filter-list-name*** in a **policy-statement** at the **[edit policy-options policy-statement *policy-statement-name* term *term-name* from]** hierarchy level. Route filter lists can be used in conjunction with other **route-filter** statements.

For example:

```
[edit]
user@router# show policy-options policy-statement test-route-filter-list-statement
from {
  route-filter 198.51.100.32/29 exact accept;
  route-filter 192.0.2.1/32 exact;
  route-filter-list rf-test-list;
}
then reject;
```

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

rtf-prefix-list

Syntax

rtf-prefix-list *name route-targets*

Hierarchy Level

[edit logical-systems *logical-system-name* policy-options],
[edit logical-systems *logical-system-name* policy-options policy-statement *policy-name* term *term-name*],
[edit policy-options],
[edit policy-options policy-statement *policy-name* term *term-name*]

Release Information

Statement introduced in Junos OS Release 12.2.

Description

Define a list of route target prefixes for use in a routing policy statement. These prefixes are only useful for filtering routes in the bgp.rtarget.0 table.

The route target filtering prefix is in the format: *AS number:route target extended community/length*. The first number represents the autonomous system (AS) of the device that sent the advertisement. The second group of numbers represent the route target extended community. The format of the extended community is the same as the extended community type **target:**. For more information about extended communities, see "Understanding How to Define BGP Communities and Extended Communities" on page 361.

In this route target prefix example 64500:200:101/96, 64500 is the AS number, 200:101 is the BGP extended community used for the route target, and 96 is the prefix length.

For more information about the route target community, see RFC 4360, *BGP Extended Communities Attribute*.

For more information about the route target filtering prefix format, see RFC 4684, *Constrained Route Distribution for Border Gateway Protocol/MultiProtocol Label Switching (BGP/MPLS) Internet Protocol (IP) Virtual Private Networks (VPNs)*.

Options

name—Name that identifies the list of route target filtering prefixes. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose the entire name in quotation marks (" ").

route-targets—List of route target filtering prefixes, one route target filter per line in the configuration. When you use the **rtf-prefix-list** statement as a match condition, you do not have the option of configuring the list of route target filtering prefixes. You must first define and configure the route target filtering prefixes with the policy-options statement.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- *Example: Configuring an Export Policy for BGP Route Target Filtering for VPNs*

- *Configuring BGP Route Target Filtering for VPNs*
- *Understanding Proxy BGP Route Target Filtering for VPNs*
- Understanding How to Define BGP Communities and Extended Communities on page 361 in the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*
- *family route-target*

source-address-filter-list

List of Syntax
source-address-filter-list (Create List) on page 1239
source-address-filter-list (Use list) on page 1239

source-address-filter-list (Create List)
source-address-filter-list *source-address-filter-list-name*;

Hierarchy Level
[edit logical-systems *logical-system-name* policy-options],
[edit policy-options]

source-address-filter-list (Use list)
source-address-filter-list *source-address-filter-list-name*;

Hierarchy Level
[edit logical-systems *logical-system-name* policy-statement *policy-statement-name* term *term-name* from],
[edit policy-options policy-statement *policy-statement-name* term *term-name* from]

Release Information
Statement first introduced in Junos OS Release 16.1 on M Series, MX Series, and T Series.

Description
The source address filter list is a user configured list of individual source address filters, typically used to match an incoming route address to unicast source addresses in Multiprotocol BGP (MBGP) and Multicast Source Discovery Protocol (MSDP) environments, that you create at the **[edit policy-options]** hierarchy level. Each item in the list consists of a complete source address filter statement, made up of a source-prefix address, a match-type, and an optional action.

For example:

```
[edit]
user@router# show policy-options source-address-filter-list saf-test-list
203.0.113.0/26 exact;
192.0.2.0/24 longer accept;
198.51.100.8/29 exact reject;
```

Once configured, **source-address-filter-list** is used by referencing its ***source-address-filter-list-name*** in a **policy-statement** at the **[edit policy-options policy-statement *policy-statement-name* term *term-name* from]** hierarchy level. Source address filter lists can be used in conjunction with other **source-address-filter** statements.

For example:

```
[edit]
user@router# show policy-options policy-statement test-saf-list-statement
from {
source-address-filter 198.51.100.16/29 exact accept;
source-address-filter-list saf-test-list;
}
then reject;
```

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

standby (Routing Policy Condition)

Syntax standby;

Hierarchy Level [edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family ccc],
[edit policy-options condition if-route-exists address-family ccc],

Release Information Statement introduced in Junos OS Release 13.2.

Description Specify that the route must be in standby state to be considered a match.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation
- *Example: Configuring Pseudowire Redundancy in a Mobile Backhaul Scenario*

table

Syntax

table *table-name*;

Hierarchy Level

[edit dynamic policy-options condition],
[edit logical-systems *logical-system-name* policy-options condition if-route-exists],
[edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family ccc],
[edit logical-systems *logical-system-name* policy-options condition if-route-exists address-family inet],
[edit policy-options condition if-route-exists],
[edit policy-options condition if-route-exists address-family ccc],
[edit policy-options condition if-route-exists address-family inet]

Release Information

Statement introduced in Junos OS Release 9.0.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5.
Support for configuration in the dynamic database introduced in Junos OS Release 9.5 for EX Series switches.
Support for the address families introduced in Junos OS Release 13.2.

Description

Specify a routing table in which the route must exist for the condition to be met and the route to be considered a match.

Options

table *table-name*—Routing table name, such as inet.0.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Conditional Advertisement and Import Policy (Routing Table) with certain match conditions on page 490
- *Example: Configuring Pseudowire Redundancy in a Mobile Backhaul Scenario*
- dynamic-db on page 1184

walkup

Syntax walkup;

Hierarchy Level [edit logical-system *logical-system-name* policy-options defaults route-filter],
[edit logical-system *logical-system-name* policy-options policy-statement
policy-statement-name defaults route-filter],
[edit policy-options defaults route-filter],
[edit policy-options policy-statement *policy-statement-name* defaults route-filter]

Release Information Statement introduced in Junos OS Release 13.3 on ACX Series, EX 4600, M Series, MX Series, PTX Series, QFabric System, QFX Series standalone switches, and T Series platforms.

Description Enable route filter walkup globally or locally for a particular policy statement. The walkup feature examines more than the longest match route filters in a policy statement term with more than one route filter, allowing consolidation of terms and a potential performance enhancement.

Default By default, no route filter walkup is performed and only the longest match route filter in a policy statement term is examined.

Required Privilege Level routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- **no-walkup on page 1224**
- **route-filter on page 1235**
- Walkup for Route Filters Overview on page 234
- Configuring Walkup for Route Filters to Improve Operational Efficiency on page 238
- Example: Configuring Walkup for Route Filters Globally to Improve Operational Efficiency on page 247
- Example: Configuring Walkup for Route Filters Locally to Improve Operational Efficiency on page 252

priority (policy-options)

Syntax

```
policy-options {
  policy-statement policy-name {
    term term-name {
      from {
        match-conditions;
        route-filter destination-prefix match-type;
      }
    then {
      priority high | low;
    }
  }
}
```

Hierarchy Level

[edit logical-systems *name* policy-options policy-statement*policy-name* term *term-name* then],
[edit logical-systems *name* policy-options policy-statement*policy-name* then],
[edit policy-options policy-statement*policy-name* term *term-name* then],
[edit policy-options policy-statement*policy-name* then]

Release Information

Statement introduced in Junos OS Release 16.1.

Description

Sets the priority for route installation. You can choose a relative priority of **high**, **low** , or **medium** to ensure that high priority IGP and LDP routes are updated in the FIB (forwarding table) before medium and low priority routes. Routes are placed in different priority queues according to the priority. Any routes that are not explicitly assigned a priority are treated as medium priority. Within the same priority level, routes will continue to be updated in lexicographic order.

Options

high—Set priority to high.

low—Set priority to low.

medium—Set priority to medium.

Required Privilege Level

routing—To view this statement in the configuration.
routing-control—To add this statement to the configuration.

Related Documentation

- Prefix Prioritization Overview on page 12
- Example: Configuring the Priority for Route Prefixes in the RPD Infrastructure on page 293

Firewall Filter Configuration Statements

accounting-profile

Syntax accounting-profile *name*;

Hierarchy Level [edit firewall family *family-name* filter *filter-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.
Statement introduced in Junos OS Release 15.1F6 for PTX Series routers with third-generation FPCs installed.

Description Enable collection of accounting data for the specified filter.

Options *name*—Name assigned to the accounting profile.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Accounting for Firewall Filters Overview on page 825

enhanced-mode

Syntax enhanced-mode;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall family *family-name* filter *filter-name*],
[edit firewall filter *filter-name*],
[edit firewall family *family-name* filter *filter-name*],
[edit logical-systems *logical-system-name* firewall filter *filter-name*],
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name*]

Release Information Statement introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Limit static service filters or API-client filters to term-based filter format only for inet or inet6 families when enhanced network services mode is configured at the **[edit chassis network-services]** hierarchy level. When used with one of the chassis enhanced network services modes, firewall filters are generated in term-based format for use with MPC modules.

If enhanced network services are not configured for the chassis, the **enhanced-mode** statement is ignored and any enhanced mode firewall filters are generated in both term-based and the default, compiled format. Only term-based (enhanced) firewall filters will be generated, regardless of the setting of the **enhanced-mode** statement at the [**edit chassis network-services**] hierarchy level, if any of the following are true:

- Flexible filter match conditions are configured at the **[edit firewall family *family-name* filter *filter-name* term *term-name* from]** or **[edit firewall filter *filter-name* term *term-name* from]** hierarchy levels.
- A tunnel header push or pop action, such as GRE encapsulate or decapsulate is configured at the **[edit firewall family *family-name* filter *filter-name* term *term-name* then]** hierarchy level.
- Payload-protocol match conditions are configured at the **[edit firewall family *family-name* filter *filter-name* term *term-name* from]** or **[edit firewall filter *filter-name* term *term-name* from]** hierarchy levels.
- An extension-header match is configured at the **[edit firewall family *family-name* filter *filter-name* term *term-name* from]** or **[edit firewall filter *filter-name* term *term-name* from]** hierarchy levels.
- A match condition is configured that only works with MPC cards, such as firewall bridge filters for IPv6 traffic.

NOTE: You cannot attach enhanced mode filters to local loopback, management, or MS-DPC interfaces. These interfaces are processed by the Routing Engine and DPC modules and can accept only compiled firewall filter format. In cases where both filter formats are needed for dynamic service filters, you can use the *enhanced-mode-override* statement on the specific filter definition to override the default filter term-based only format of chassis network-service enhanced IP mode.

NOTE: Do not use enhanced mode for firewall filters that are intended for control plane traffic. Control plane filtering is handled by the Routing Engine kernel, which cannot use the term-based format of the enhanced mode filters.

For packets sourced from the Routing Engine, the Routing Engine processes Layer 3 packets by applying output filters to the packets and forwards Layer 2 packets to the Packet Forwarding Engine for transmission. By configuring the enhanced mode filter, you explicitly specify that only the term-based filter format is used, which also implies that the Routing Engine cannot use this filter.

NOTE: The **enhanced-mode** and the **enhanced-mode-override** statements are mutually exclusive; you can define the filter with either **enhanced-mode** or **enhanced-mode-override**, but not both.

NOTE: For MX Series routers with MPCs, you need to initialize certain new firewall filters by walking the corresponding SNMP MIB, for example, **show snmp mib walk *name* ascii**. This forces Junos to learn the filter counters and ensure that the filter statistics are displayed. This guidance applies to all enhanced mode firewall filters, filters with flexible conditions, and filters with the certain terminating actions. See those topics, listed under Related Documentation, for details.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *enhanced-mode-override*
- *Network Services Mode Overview*
- *Firewall Filters and Enhanced Network Services Mode Overview*
- *Configuring a Filter for Use with Enhanced Network Services Mode*
- Firewall Filter Match Conditions for IPv4 Traffic on page 589
- Firewall Filter Match Conditions for IPv6 Traffic on page 629
- Firewall Filter Terminating Actions on page 680
- Firewall Filter Flexible Match Conditions on page 667

direction (forwarding-class-accounting)

Syntax direction (ingress | egress | both)

Hierarchy Level [edit interfaces *interface-name* forwarding-class-accounting]
[edit interfaces *interface-name* unit *logical-unit-number*forwarding-class-accounting]

Release Information Statement introduced in Junos OS Release 13.3R3 in MX Series.

Description Specify the direction of traffic for which you want to apply counters. A single aggregate counter per forwarding class is used for flows. Forwarding class accounting applies to IPv4, IPv6, MPLS, Layer 2 and Other traffic.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *show class-of-service interface*
- clear interfaces statistics on page 1339

family (Firewall)

Syntax

```
family family-name {
  filter filter-name {
    accounting-profile name;
    enhanced-mode;
    interface-specific;
    physical-interface-filter;
  }
  prefix-action name {
    count;
    destination-prefix-length prefix-length;
    policer policer-name;
    source-prefix-length prefix-length;
    subnet-prefix-length prefix-length;
  }
  simple-filter filter-name {
    term term-name {
      from {
        match-conditions;
      }
      then {
        action;
        action-modifiers;
      }
    }
  }
}
```

Hierarchy Level

[edit firewall],
[edit logical-systems *logical-system-name* firewall]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
simple-filter statement introduced in Junos OS Release 7.6.
any family type introduced in Junos OS Release 8.0.
bridge family type introduced in Junos OS Release 8.4 (MX Series routers only).
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure a firewall filter for IP version 4 (IPv4) or IP version 6 (IPv6) traffic. Only on MX Series routers and EX Series switches, configure a firewall filter for Layer 2 traffic in a bridging environment.

Options

family-name—Version or type of addressing protocol:

- **any**—Protocol-independent match conditions.
- **bridge**—(MX Series routers only) Layer 2 packets that are part of bridging domain.
- **ethernet-switching**—(EX Series switches) Filter Layer 2 (Ethernet) packets and Layer 3 (IP) packets.

- **ccc**—Layer 2 switching cross-connects.
- **inet**—IPv4 addressing protocol.
- **inet6**—IPv6 addressing protocol.
- **mpls**—MPLS.
- **vpls**—Virtual private LAN service (VPLS).

The remaining statements are explained separately. See CLI Explorer.

NOTE: **The packet lengths that a policer considers depends on the address family of the firewall filter.**

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Guidelines for Configuring Service Filters on page 908
- Guidelines for Configuring Simple Filters on page 929

filter-list-template

Syntax filter-list-template;

Hierarchy Level [edit firewall family (inet | inet6) filter *filter-name*],
[edit logical-systems *logical-system-name* firewall family (inet | inet6) filter *filter-name*]

Release Information Statement introduced in Junos OS Release 13.3R9.

Description (MX5, MX10, MX40, and MX80 routers, and routers that use MX Series MPC line cards only) Configure all interfaces that use the same filter list to use a common template. This feature can be used to save microkernel memory and DMEM memory.

If the same filter list cannot be used on all interfaces, consider merging the filters and using the **from interface** firewall filter term to group the per-interface terms to produce a new common filter list.

NOTE: **If you configure both fast-lookup-filter and interface-specific statements, filter list templates are also used.**

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *input-list*
- *output-list*
- Firewall Filter Match Conditions for IPv6 Traffic on page 629

filter (Applying to a Logical Interface)

Syntax

```
filter {
  group filter-group-number;
  input filter-name;
  input-list [ filter-names ];
  output filter-name;
  output-list [ filter-names ];
}
```

Hierarchy Level

Protocol-independent firewall filter on MX Series router logical interface:

[edit interfaces *interface-name* unit *logical-unit-number*],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]

All other standard firewall filters on all other devices:

[edit interfaces *interface-name* unit *logical-unit-number* family *family*],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* family *family*]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Apply a stateless firewall filter to a logical interface at a specific protocol level.

Options

group *filter-group-number*—(Only Ex, M, MX, and T Series) Number of the group to which the interface belongs. Range: 1 through 255

input *filter-name*—Name of one filter to evaluate when packets are received on the interface.

input-list [*filter-names*]—Names of filters to evaluate when packets are received on the interface. Up to 16 filters can be included in a filter input list.

output *filter-name*—Name of one filter to evaluate when packets are transmitted on the interface.

output-list [*filter-names*]—Names of filters to evaluate when packets are transmitted on the interface. Up to 16 filters can be included in a filter output list.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Guidelines for Applying Standard Firewall Filters on page 581

filter (Configuring)

Syntax

```
filter filter-name {
  accounting-profile name;
  enhanced-mode;
  fast-lookup-filter;
  filter-list-template;
  interface-shared;
  interface-specific;
  physical-interface-filter;
  promote gre-key;
  term term-name {
    ... term configuration ...
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall family *family-name*],
[edit firewall family *family-name*],
[edit logical-systems *logical-system-name* firewall family *family-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
physical-interface-filter statement introduced in Junos OS Release 9.6.
Support for the **interface-shared** statement introduced in Junos OS Release 12.2.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure firewall filters.

Options

filter-name—Name that identifies the filter. This must be a non-reserved string of not more than 64 characters. To include spaces in the name, enclose it in quotation marks (“ ”). Firewall filter names are restricted from having the form __.*__ (beginning and ending with underscores) or __.* (beginning with an underscore.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Guidelines for Applying Standard Firewall Filters on page 581
- *Configuring Multifield Classifiers*
- *Using Multifield Classifiers to Set Packet Loss Priority*
- simple-filter on page 1264

firewall

Syntax

firewall {
  atm-policer *atm-policer-name* {
    ... *atm-policer-configuration* ...
  }
  family *protocol-family-name* {
    ... *protocol-family-configuration* ...
  }
  filter *ipv4-filter-name* {
    ... *ipv4-filter-configuration* ...
  }
  hierarchical-policer *hierarchical-policer-name* {
    ... *hierarchical-policer-configuration* ...
  }
  interface-set *interface-set-name* {
    ... *interface-set-configuration* ...
  }
  policer *two-color-policer-name* {
    ... *two-color-policer-configuration* ...
  }
  three-color-policer *three-color-policer-name* {
    ... *three-color-policer-configuration* ...
  }
}

Hierarchy Level

[edit],
[edit logical-systems *logical-system-name*]
[edit dynamic-profiles *profile-name*],

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure firewall filters.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Guidelines for Configuring Service Filters on page 908
- Guidelines for Configuring Simple Filters on page 929
- *Configuring Multifield Classifiers*
- *Using Multifield Classifiers to Set Packet Loss Priority*

force-premium (Firewall Filter Action)

Syntax force-premium;

Hierarchy Level [edit firewall family *family-name* filter *filter-name* term *term-name* then],
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name* term *term-name* then]

Release Information Statement introduced in Junos OS Release 12.3 for **family inet** and **inet6**.
Support for **family vpls**, **ccc**, and **bridge** added in Junos OS Releases 13.3R8, 13.3R10, 14.1R8, 14.2R7, 15.1R4,16.1R1, and 17.1R1.

Description Firewall filter option to force premium treatment for traffic (MX Series routers)— By default, a hierarchical policer processes the traffic it receives according to the traffic's forwarding class. Premium, expedited-forwarding traffic has priority for bandwidth over aggregate, best-effort traffic. Now you can include the **force-premium** option at the **[edit firewall filter *filter-name* term *term-name*]** hierarchy level to ensure that traffic matching the term is treated as premium traffic by a subsequent hierarchical policer, regardless of its forwarding class. This traffic is given preference over any aggregate traffic received by that policer. Consider a scenario where a firewall filter is applied to an interface that receives both expedited-forwarding voice traffic and best-effort video traffic. Traffic that matches the first term of the filter is passed to a hierarchical policer in the second term. The hierarchical policer also receives best-effort data traffic from another source. The filtered video traffic is treated the same as this data traffic, as aggregate traffic with a lower priority than the premium voice traffic. Consequently, some of the video traffic might be dropped and some of the data traffic passed on.

To avoid that situation, include the **force-premium** option in the firewall filter term that passes traffic to the hierarchical policer. This term forces the video traffic to be marked as premium traffic. The hierarchical policer gives both the voice traffic and the video traffic priority over the aggregate data traffic.

NOTE: The force-premium filter option is supported only on MPCs.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Firewall Filter Nonterminating Actions on page 673

forwarding-class (Firewall Filter Action)

Syntax forwarding-class *class-name*;

Hierarchy Level [edit firewall family *family-name* filter *filter-name* term *term-name* then],
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name* term *term-name* then]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Set the forwarding class of incoming packets.

Options ***class-name***—Name of the forwarding class.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Firewall Filter Nonterminating Actions on page 673
- Policer Color-Marking and Actions on page 990
- Multifield Classification Overview on page 1088

hierarchical-policer

List of Syntax

Syntax (M Series, MX Series, T Series - Bandwidth-Based) on page 1256
Syntax (MX Series - Packets-Per-Second (pps)-Based) on page 1256

Syntax (M Series, MX Series, T Series - Bandwidth-Based)

```
hierarchical-policer hierarchical-policer-name | uid {
  aggregate {
    if-exceeding {
      bandwidth-limit bps;
      burst-size-limit bytes;
    }
    then {
      discard;
    }
  }
  premium {
    if-exceeding {
      bandwidth-limit bps;
      burst-size-limit bytes;
    }
    then {
      discard;
    }
  }
}
```

Syntax (MX Series - Packets-Per-Second (pps)-Based)

```
hierarchical-policer hierarchical-policer-name | uid {
  aggregate {
    if-exceeding-pps {
      pps-limit pps;
      packet-burst packets;
    }
    then {
      discard;
    }
  }
  premium {
    if-exceeding-pps (Hierarchical Policer) {
      pps-limit (Hierarchical Policer) pps;
      packet-burst (Hierarchical Policer) packets;
    }
    then {
      discard;
    }
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall],
[edit firewall]

Release Information

Statement introduced in Junos OS Release 9.5.

Support at the **[edit dynamic-profiles *profile-name* firewall]** hierarchy level introduced in Junos OS Release 11.4.
Support for **if-exceeding-pps** statement on MX Series routers with MPCs introduced in Junos OS Release 15.2.

Description Use a hierarchical policer to rate-limit ingress Layer 2 traffic at a physical or logical interface and apply different policing actions based on whether the packets are classified as **premium** for expedited forwarding (EF) or **aggregate** for a lower priority. The two policers defined within the hierarchical policer are **aggregate** and **premium**.

Hierarchical policers are supported on Enhanced Intelligent Queuing (IQE) PICs and SONET interfaces hosted on the M120 and M320 with incoming Flexible PIC Concentrators (FPCs) as SFPC and outgoing FPCs as FFPC; on MPCs hosted on MX Series routers; on the T320, T640, and T1600 with Enhanced Intelligent Queuing (IQE) PICs; and on the T4000 with Type 5 FPC and Enhanced Scaling Type 4 FPC.

NOTE:

- The **if-exceeding-pps** statement is only supported on MX Series routers with MPCs.
- The **if-exceeding** and **if-exceeding-pps** statements are mutually exclusive and, therefore, cannot be applied at the same time.

Options *hierarchical-policer-name*—Name that identifies the policer. The name can contain letters, numbers, and hyphens (-), and can be up to 255 characters long. To include spaces in the name, enclose the name in quotation marks (“ ”).

uid—When you configure a hierarchical policer at the **[edit dynamic-profiles *profile name* firewall]** hierarchy level, you must assign a variable UID as the policer name.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policer Configuration Overview on page 981
- Hierarchical Policers on page 1009
- **aggregate (Hierarchical Policer) on page 1273**
- **bandwidth-limit (Hierarchical Policer) on page 1274**
- **burst-size-limit (Hierarchical Policer) on page 1280**
- **pps-limit (Hierarchical Policer) on page 1324**
- **packet-burst (Hierarchical Policer) on page 1315**
- **if-exceeding (Hierarchical Policer) on page 1298**
- *if-exceeding-pps (Hierarchical Policer)*
- **premium (Hierarchical Policer) on page 1329**

interface-group (Decapsulate GRE)

Syntax interface-group (0 -255)

Hierarchy Level [edit firewall family *family-name* filter *filter-name* term *term-name* then decapsulate gre],

Release Information Statement introduced in Junos OS Release 14.1 for MX 80, MX 240, MX 480, MX 960, MX 2010, and MX 2020 routers with MPC2, MPC3, MPC4, MPC5, or MPC6 MPCs.

Description Allows you to explicitly specify and add an interface group to packets after they have undergone GRE decapsulation. In releases prior to Junos OS Release 14.1, the interface group upon decapsulation was always 0 and could not be changed.

In Junos OS Release 14.1 and later, you can assign any arbitrary value in the range of 1 to 255 to the packets' **interface-group** upon GRE decapsulation. For example, you could use this command to retain the original interface from which the packet was received (if no value is set, the default interface group is 0). You could also use it to ensure that all decapsulated GRE packets are placed in the same group, for example to trigger additional filtering in the forwarding table on the basis of the this data from the inner packet.

The value used in **interface-group** is set after the GRE packet is decapsulate by a filter action in a filter attached to a given ingress interface.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Example: Configuring a Stateless Firewall Filter on an Interface Group on page 867
- *Applying Forwarding Table Filters*
- **decapsulate on page 1181**

interface-set

Syntax

```
interface-set interface-set-name {
  interface-name;
}
```

Hierarchy Level

[edit firewall],
[edit logical-systems *logical-system-name* firewall]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure an interface set.

Options

interface-name—Names of each interface to include in the interface set. You must specify more than one name.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Filtering Packets Received on an Interface Set Overview on page 862

interface-shared

Syntax interface-shared;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall family *family-name* filter *filter-name*],
[edit firewall family *family-name* filter *filter-name*],

Release Information Statement introduced in Junos OS Release 12.2.

Description Set the interface-shared attribute for a firewall filter.

NOTE: A firewall filter cannot be both interface-specific and interface-shared.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *Interface-Shared Filters Overview*
- *Understanding Dynamic Firewall Filters*
- *Classic Filters Overview*
- *Basic Classic Filter Syntax*

interface-specific (Firewall Filters)

Syntax interface-specific;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall family *family-name* filter *filter-name*],
[edit firewall family *family-name* filter *filter-name*],
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Configure interface-specific names for firewall counters.

NOTE: A firewall filter cannot be both interface-specific and interface-shared.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *Configuring Firewall Filters and Policers for VPLS*
- Interface-Specific Firewall Filter Instances Overview on page 859

promote gre-key

Syntax

promote gre-key;

Hierarchy Level

[edit firewall family *family-name* filter *filter-name*]

Release Information

Statement introduced in Junos OS Release 15.1F3 and 16.1R2 for PTX5000 routers with third-generation FPCs.
Statement introduced in Junos OS Release 15.1F6 and 16.1R2 for PTX3000 routers with third-generation FPCs.

Description

You must configure the **promote gre-key** statement if you want to use gre-key as one of the matches in a filter. When you configure **promote gre-key** and use gre-key in any of the terms in a filter, the entire filter is compiled in a way that optimizes performance of the filter for gre-key matching.

NOTE: The **promote gre-key** configuration statement is supported on PTX Series routers only when network services is set to **enhanced-mode**. For more information, see **enhanced-mode**.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- Guidelines for Applying Standard Firewall Filters on page 581
- Firewall Filter Match Conditions for IPv4 Traffic on page 589

service-filter (Firewall)

Syntax

```
service-filter filter-name {
  term term-name {
    from {
      match-conditions;
    }
    then {
      actions;
    }
  }
}
```

Hierarchy Level

[edit firewall family (inet | inet6],
[edit logical-systems *logical-system-name* firewall family (inet | inet6)]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Statement introduced in Junos OS Release 12.3X51–D15 for ACX Series routers.

Description

Configure service filters.

NOTE: ACX Series routers do not support family *inet6*.

Options

filter-name—Name that identifies the service filter. The name can contain letters, numbers, and hyphens (-) and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (“ ”).

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Guidelines for Configuring Service Filters on page 908
- Guidelines for Applying Service Filters on page 910

simple-filter

Syntax

```
simple-filter filter-name {
  term term-name {
    from {
      match-conditions;
    }
    then {
      forwarding-class class-name;
      loss-priority (high | low | medium);
    }
  }
}
```

Hierarchy Level

[edit firewall family inet],
[edit logical-systems *logical-system-name* firewall family inet]

Release Information

Statement introduced in Junos OS Release 7.6.
Logical systems support introduced in Junos OS Release 9.3.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure simple filters.

Options

filter-name—Name that identifies the simple filter. The name must be a non-reserved string of not more than 64 characters. No special characters are restricted. To include spaces in the name, enclose them in quotation marks (“ ”).

from—Match packet fields to values. If the **from** option is not included, all packets are considered to match and the actions and action modifiers in the **then** statement are taken.

match-conditions—One or more conditions to use to make a match.

term *term-name*—Define a simple-filter term. The name that identifies the term can contain letters, numbers, and hyphens (-), and can be up to 255 characters long. To include spaces in the name, enclose them in quotation marks (“ ”).

then—Actions to take on matching packets. If the **then** option is not included and a packet matches all the conditions in the **from** statement, the packet is accepted.

NOTE: Only **forwarding-class** and **loss-priority** are valid actions in a simple filter configuration.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *simple-filter (Applying to an Interface)*
- Simple Filter Overview on page 927
- How Simple Filters Evaluate Packets on page 927
- Guidelines for Configuring Simple Filters on page 929
- Guidelines for Applying Simple Filters on page 932

term (Firewall Filter)

Syntax

```
term term-name {
  from {
    match-conditions;
    vxlan {
      vni vni-id
        flags value mask-in-hex value
        reserved1 value
        reserved2 value
      }
    ip-version ipv4 {
      match-conditions-mpls-ipv4-address;
        protocol (tcp | udp) {
          match conditions-mpls-ipv4-port;
        }
    }
  }
  then {
    actions;
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall family *family-name* filter *filter-name*],
[edit firewall family *family-name* filter *filter-name*],
[edit firewall family *family-name* service-filter *filter-name*],
[edit firewall family *family-name* simple-filter *filter-name*],
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name*],
[edit logical-systems *logical-system-name* firewall family *family-name* service-filter *filter-name*],
[edit logical-systems *logical-system-name* firewall family *family-name* simple-filter *filter-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
filter option introduced in Junos OS Release 7.6.
Logical systems support introduced in Junos OS Release 9.3.
ip-version ipv4 support introduced in Junos OS Release 10.1.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Define a firewall filter term.

Options

actions—(Optional) Actions to perform on the packet if conditions match. You can specify one *terminating action* supported for the specified filter type. If you do not specify a terminating action, the packets that match the conditions in the **from** statement are accepted by default. As an option, you can specify one or more *nonterminating actions* supported for the specified filter type.

filter-name—(Optional) For family ***family-name*** filter ***filter-name*** only, reference another standard stateless firewall filter from within this term.

from—(Optional) Match packet fields to values. If not included, all packets are considered to match and the actions and action modifiers in the **then** statement are taken.

match-conditions—One or more conditions to use to make a match on a packet.

match-conditions-mpls-ipv4-address—(MPLS-tagged IPv4 traffic only) One or more IP address match conditions to match on the IPv4 packet header. Supports network-based service in a core network with IPv4 packets as an inner payload of an MPLS packet with labels stacked up to five deep.

match-conditions-mpls-ipv4-port—(MPLS-tagged IPv4 traffic only) One or more UDP or TCP port match conditions to use to match a packet in an MPLS flow. Supports network-based service in a core network with IPv4 packets as an inner payload of an MPLS packet with labels stacked up to five deep.

vxlan—(Optional) Match packets belonging to a particular VXLAN Network Identifier (VNI).

term-name—Name that identifies the term. The name can contain letters, numbers, and hyphens (-) and can be up to 64 characters long. To include spaces in the name, enclose it in quotation marks (" ").

then—(Optional) Actions to take on matching packets. If not included and a packet matches all the conditions in the **from** statement, the packet is accepted.

The Firewall Filer Match Conditions for the different protocols are explained separately:

- Firewall Filter Match Conditions for IPv4 Traffic on page 589
- Firewall Filter Match Conditions for IPv6 Traffic on page 629
- Firewall Filter Match Conditions for MPLS Traffic on page 639
- Firewall Filter Match Conditions for MPLS-Tagged IPv4 or IPv6 Traffic on page 640
- Firewall Filter Match Conditions for VPLS Traffic on page 643
- Firewall Filter Match Conditions for Protocol-Independent Traffic on page 615
- *Firewall Filter Match Conditions for Protocol-Independent Traffic in Dynamic Service Profiles*
- Firewall Filter Match Conditions Based on Numbers or Text Aliases on page 601
- Firewall Filter Match Conditions Based on Bit-Field Values on page 602
- Firewall Filter Match Conditions Based on Address Fields on page 606
- Firewall Filter Match Conditions Based on Address Classes on page 614
- Firewall Filter Match Conditions for Layer 2 Bridging Traffic on page 658
- Firewall Filter Match Conditions for Layer 2 CCC Traffic on page 654

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Guidelines for Configuring Firewall Filters on page 576
- *Configuring Multifield Classifiers*
- Guidelines for Configuring Simple Filters on page 929
- Guidelines for Configuring and Applying Firewall Filters in Logical Systems on page 788

tunnel-end-point

Syntax

```
tunnel-end-point tunnel-name {
  encapsulation-protocol [ protocol-options ];
  transport-protocol {
    destination-address destination-host-address;
    source-address source-host-address;
  }
}
```

Hierarchy Level

[edit firewall],
[edit logical-systems *logical-system-name* firewall]

Release Information

Statement introduced in Junos OS Release 12.3R2.

Description

On an MX Series router installed as an *encapsulator* (an ingress PE router) for filter-based IP tunneling, define a *tunnel template*. The template specifies a set of characteristics for transporting passenger protocol packets across an IP transport network.

Options

destination-host-address—IP address or address range of the de-encapsulator (the remote egress PE router).

encapsulation-protocol—Encapsulation protocol:

- **gre**—Filter-based GRE encapsulation is supported on IPv4 transport networks.
- **l2tp**—Filter-based L2TP encapsulation is supported on IPv4 transport networks.

protocol-options—(Optional) Protocol-specific encapsulation options:

- **key *number***—An integer value that uniquely identifies a GRE IPv4 tunnel if multiple traffic flows share the same ***source-address*** and ***destination-address*** pair. Range: 1 through 0xFFFFFFFF (4,294,967,295 decimal)

  If a tunnel definition specifies GRE IPv4 tunneling using a key, the system includes the key in the GRE header whenever a Packet Forwarding Engine is instructed to use that tunnel definition to encapsulate a packet.
- **session-id *session-id***—(Optional) Unique integer that identifies the L2TP control connection for the L2TP session. It is a 32-bit field containing a non-zero identifier for a session. L2TP sessions are named by identifiers that have local significance only
- **tunnel-id *tunnel-id***—(Optional) Unique integer that identifies the L2TP control connection for the tunnel defined.
- **cookie *l2tpv3-cookie***—(Optional) For L2TP tunnels, specify the L2TP cookie for the duplicated packets. If the tunnel does not contain the receive-cookie configured, packet injection does not happen. In such a case, any received tunnel packet is

counted and dropped in the same manner in which packets that arrive with a wrong cookie are counted and dropped.

source-host-address—IP address or address range of the encapsulator (the local ingress PE router).

transport-protocol—The IP network protocol used to transport encapsulated passenger protocol packets:

- **ipv4**—IPv4 can transport IPv4, IPv6, or MPLS packets encapsulated using filter-based generic routing encapsulation (GRE).

tunnel-name—Name that identifies the tunnel template using a non-reserved string of not more than 64 characters. To include spaces in the name, enclose it in quotation marks (" "). You can reference a tunnel template name in an ingress firewall filter of type **inet**, **inet6**, **any**, or **mpls** by configuring the **encapsulate** terminating action.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Understanding Filter-Based Tunneling Across IPv4 Networks on page 875
- Interfaces That Support Filter-Based Tunneling Across IPv4 Networks on page 881
- Components of Filter-Based Tunneling Across IPv4 Networks on page 883
- Example: Transporting IPv6 Traffic Across IPv4 Using Filter-Based Tunneling on page 888
- Firewall Filter Terminating Actions on page 680

Traffic Policer Configuration Statements

- two-rate on page 1335
- policer-overhead-adjustment on page 1336

action

Syntax

```
action {
  loss-priority high then discard;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall three-color-policer *name*],
[edit firewall three-color-policer *name*],
[edit logical-systems *logical-system-name* firewall three-color-policer *name*]

Release Information

Statement introduced in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... three-color-policer]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Discard traffic on a logical interface using tricolor marking policing.

NOTE: This statement is supported only on IQ2 interfaces.

The remaining statement is explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Basic Single-Rate Three-Color Policers on page 1121
- Basic Two-Rate Three-Color Policers on page 1127
- Two-Color and Three-Color Logical Interface Policers on page 1141
- Two-Color and Three-Color Physical Interface Policers on page 1154
- Two-Color and Three-Color Policers at Layer 2 on page 1016
- **loss-priority high then discard on page 1311**

aggregate (Hierarchical Policer)

Syntax

```
aggregate {
  if-exceeding {
    bandwidth-limit bandwidth;
    burst-size-limit burst;
  }
  then {
    discard;
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall hierarchical-policer *name*],
[edit firewall hierarchical-policer]

Release Information

Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... hierarchical-policer *name*]** hierarchy level introduced in Junos OS Release 11.4.

Description

On M40e, M120, and M320 edge routers with Flexible PIC Concentrator (FPC) input as FFPC and FPC output as SFPC, and on MX Series, T320, T640, and T1600 edge routers with Enhanced Intelligent Queuing (IQE) PICs, T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, configure an aggregate hierarchical policer.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policer Configuration Overview on page 981
- Hierarchical Policers on page 1009
- **bandwidth-limit (Hierarchical Policer) on page 1274**
- **burst-size-limit (Hierarchical Policer) on page 1280**
- **hierarchical-policer on page 1256**
- **if-exceeding (Hierarchical Policer) on page 1298**
- **premium on page 1329**

bandwidth-limit (Hierarchical Policer)

Syntax bandwidth-limit *bps*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall hierarchical-policer aggregate if-exceeding],
[editdynamic-profiles *profile-name* firewall hierarchical-policer premium if-exceeding],
[edit firewall hierarchical-policer aggregate if-exceeding],
[edit firewall hierarchical-policer premium if-exceeding]

Release Information Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... if-exceeding]** hierarchy level introduced in Junos OS Release 11.4.

Description On M40e, M120, and M320 (with FFPC and SFPC) edge routers; on MPCs hosted on MX Series routers; on T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs; and on T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, configure the maximum average bandwidth for premium or aggregate traffic in a hierarchical policer.

Options *bps*—You can specify the number of bits per second either as a decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range:

- 32,000 through 50,000,000,000 on M Series routers
- 32,000 through 100,000,000,000 on T Series routers
- 32,000 through 18,446,744,073,709,551,615 on MX Series routers

NOTE: When you specify a numeric value beyond the supported bandwidth of the PFE, the router caps the bandwidth at the maximum supported bandwidth of the PFE.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policer Configuration Overview on page 981
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Single Token Bucket Algorithm on page 992
- Determining Proper Burst Size for Traffic Policers on page 1002
- **aggregate (Hierarchical Policer) on page 1273**

- burst-size-limit (Hierarchical Policer) on page 1280
- premium (Hierarchical Policer) on page 1329

bandwidth-limit (Policer)

Syntax bandwidth-limit *bps*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name* if-exceeding],
[edit firewall policer *policer-name* if-exceeding],
[edit logical-systems *logical-system-name* policer *policer-name* if-exceeding]

Release Information Statement introduced before Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... if-exceeding]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description For a single-rate two-color policer, configure the bandwidth limit as a number of bits per second. Single-rate two-color policing uses the single token bucket algorithm to measure traffic-flow conformance to a two-color policer rate limit.

Traffic at the interface that conforms to the bandwidth limit is categorized green. Traffic that exceeds the specified rate is also categorized as green provided that sufficient tokens remain in the single token bucket. Packets in a green flow are implicitly marked with **low** packet loss priority (PLP) and then passed through the interface.

Traffic that exceeds the specified rate when insufficient tokens remain in the single token bucket is categorized red. Depending on the configuration of the two-color policer, packets in a red traffic flow might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface.

NOTE: This statement specifies the bandwidth limit as an absolute number of bits per second. Alternatively, for single-rate two-color policers only, you can use the **bandwidth-percent** *percentage* statement to specify the bandwidth limit as a percentage of either the physical interface port speed or the configured logical interface shaping rate.

Single-rate two-color policing allows bursts of traffic for short periods, whereas single-rate and two-rate three-color policing allows more sustained bursts of traffic.

Hierarchical policing is a form of two-color policing that applies different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority. You apply a hierarchical policer to ingress Layer 2 traffic to allows bursts of EF traffic for short period and bursts of non-EF traffic for short periods, with EF traffic always taking precedence over non-EF traffic.

Options *bps*—You can specify the number of bits per second either as a decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range:

- (M Series and T Series routers) 8000 through 100,000,000,000
- (Mx Series routers) 8000 through 18,446,744,073,709,551,615

NOTE: When you specify a numeric value beyond the supported bandwidth of the PFE, the router caps the bandwidth at the maximum supported bandwidth of the PFE.

Default: None.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Single Token Bucket Algorithm on page 992
- Determining Proper Burst Size for Traffic Policers on page 1002
- bandwidth-percent on page 1278
- burst-size-limit (Policer) on page 1281

bandwidth-percent

Syntax bandwidth-percent *percentage*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name* if-exceeding],
[edit firewall policer *policer-name* if-exceeding],
[edit logical-systems *logical-system-name* policer *policer-name* if-exceeding]

Release Information Statement introduced before Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... if-exceeding]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description For a single-rate two-color policer, configure the bandwidth limit as a percentage value. Single-rate two-color policing uses the *single token bucket algorithm* to measure traffic-flow conformance to a two-color policer rate limit.

Traffic at the interface that conforms to the bandwidth limit is categorized green. Traffic that exceeds the specified rate is also categorized as green provided that sufficient tokens remain in the single token bucket. Packets in a green flow are implicitly marked with **low** packet loss priority and then passed through the interface.

Traffic that exceeds the specified rate when insufficient tokens remain in the single token bucket is categorized red. Depending on the configuration of the two-color policer, packets in a red traffic flow might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface.

NOTE: This statement specifies the bandwidth limit as a percentage of either the physical interface port speed or the configured logical interface shaping rate. Alternatively, you can use the **bandwidth-limit *bps*** statement to specify the bandwidth limit as an absolute number of bits per second.

The function of the bandwidth limit is extended by the burst size (configured using the **burst-size-limit *bytes*** statement) to allow bursts of traffic up to a limit based on the overall traffic load:

- When a single-rate two-color policer is applied to the input or output traffic at an interface, the initial capacity for traffic bursting is equal to the number of bytes specified by this statement.
- During periods of relatively low traffic (traffic that arrives at or departs from the interface at overall rates below the token arrival rate), unused tokens accumulate in the bucket, but only up to the configured token bucket depth.

Single-rate two-color policing allows bursts of traffic for short periods, whereas single-rate and two-rate three-color policing allows more sustained bursts of traffic.

Hierarchical policing is a form of two-color policing that applies different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority. You apply a hierarchical policer to ingress Layer 2 traffic to allows bursts of EF traffic for short period and bursts of non-EF traffic for short periods, with EF traffic always taking precedence over non-EF traffic.

Options *percentage*—Traffic rate as a percentage of either the physical interface media rate or the logical interface configured shaping rate. You can configure a shaping rate on a logical interface by using class-of-service statement.

NOTE: You cannot rate-limit based on bandwidth percentage for tunnel or software interfaces. The bandwidth percentage policer also cannot be used for forwarding table filters. Bandwidth percentage policers can only be used for interface-specific filters. Bandwidth percentage policers applied on an aggregated Ethernet bundle or an aggregated SONET bundle *do* match the effective bandwidth and burst-size to user-configured values by default and do not require **shared-bandwidth-policer** configuration.

Range: 0 through 100

Default: None.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Single Token Bucket Algorithm on page 992
- Determining Proper Burst Size for Traffic Policers on page 1002
- Bandwidth Policers on page 1051
- **bandwidth-limit (Policer) on page 1276**
- **burst-size-limit (Policer) on page 1281**

burst-size-limit (Hierarchical Policer)

Syntax burst-size-limit *bytes*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall hierarchical-policer aggregate if-exceeding],
[editdynamic-profiles *profile-name* firewall hierarchical-policer premium if-exceeding],
[edit firewall hierarchical-policer aggregate if-exceeding],
[edit firewall hierarchical-policer premium if-exceeding]

Release Information Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... if exceeding]** hierarchy level introduced in Junos OS Release 11.4.

Description On M40e, M120, and M320 (with FFPC and SFPC) edge routers; on MPCs hosted on MX Series routers; on T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs; and on T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, configure the burst-size limit for premium or aggregate traffic in a hierarchical policer.

Options *bytes*—Burst-size limit in bytes. The minimum recommended value is the maximum transmission unit (MTU) of the IP packets being policed. You can specify the value either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 2,147,450,880 (1500 through 100,000,000,000 on MPCs hosted on MX Series routers)

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policer Configuration Overview on page 981
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Single Token Bucket Algorithm on page 992
- Determining Proper Burst Size for Traffic Policers on page 1002
- Hierarchical Policers on page 1009
- **aggregate (Hierarchical Policer) on page 1273**
- **bandwidth-limit (Hierarchical Policer) on page 1274**
- **premium (Hierarchical Policer) on page 1329**

burst-size-limit (Policer)

Syntax burst-size-limit *bytes*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name* if-exceeding],
[edit firewall policer *policer-name* if-exceeding],
[edit logical-systems *logical-system-name* policer *policer-name* if-exceeding]

Release Information Statement introduced before Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... if-exceeding]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description For a single-rate two-color policer, configure the burst size as a number of bytes. The burst size allows for short periods of traffic bursting (back-to-back traffic at average rates that exceed the configured bandwidth limit). Single-rate two-color policing uses the *single token bucket algorithm* to measure traffic-flow conformance to a two-color policer rate limit.

Traffic at the interface that conforms to the bandwidth limit is categorized green. Traffic that exceeds the specified rate is also categorized as green provided that sufficient tokens remain in the single token bucket. Packets in a green flow are implicitly marked with **low** packet loss priority and then passed through the interface.

Traffic that exceeds the specified rate when insufficient tokens remain in the single token bucket is categorized red. Depending on the configuration of the two-color policer, packets in a red traffic flow might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface.

The burst size extends the function of the bandwidth limit (configured using either the **bandwidth-limit** ***bps*** statement or the **bandwidth-percent** ***percentage*** statement) to allow bursts of traffic up to a limit based on the overall traffic load:

- When a single-rate two-color policer is applied to the input or output traffic at an interface, the initial capacity for traffic bursting is equal to the number of bytes specified by this statement.
- During periods of relatively low traffic (traffic that arrives at or departs from the interface at overall rates below the token arrival rate), unused tokens accumulate in the bucket, but only up to the configured token bucket depth.

Single-rate two-color policing allows bursts of traffic for short periods, whereas single-rate and two-rate three-color policing allows more sustained bursts of traffic.

Hierarchical policing is a form of two-color policing that applies different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority. You apply a hierarchical policer to ingress Layer 2 traffic to allows bursts of EF traffic for short period and bursts of non-EF traffic for short periods, with EF traffic always taking precedence over non-EF traffic.

Table 73 on page 1282 summarizes the relationship between the **bandwidth-limit** and the token arrival rate. This information is useful in calculating the minimum **burst-size-limit**.

Table 73: Bandwidth Limits and Token Rates

Bandwidth Limit	Token Rate
0-333 Mbps	low (262 μs)
334-666 Mbps	high (8.2 μs)
667-1333 Mbps	low
1334 Mbps and above	high

The burst-size limit enforced is based on the burst-size limit you configure. For a rate-limited logical interface, the Packet Forwarding Engine calculates the optimum burst-size-limit values and then applies the value closest to the burst-size-limit value specified in the policer configuration.

On MX Series routers and EX Series switches, the burst-size limit is not as freely configurable as it is on other platforms. Junos OS does not support an unlimited combination of policer bandwidth and burst-size limits on MX Series routers and EX Series switches. For a single-rate two-color policer on an MX Series router and on an EX Series switch, the minimum supported burst-size limit is equivalent to the amount of traffic allowed by the policer bandwidth limit in a time span of 1 millisecond. For example, for a policer configured with a **bandwidth-limit** value of 1 Gbps, the minimum supported value for **burst-size-limit** on an MX Series router is 125 KB. If you configure a value that is smaller than the minimum, Junos OS overrides the configuration and applies the actual minimum.

Options

bytes—Burst-size limit in bytes. The minimum recommended value is the maximum transmission unit (MTU) of the IP packets being policed. You can specify the value either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000

Default: None

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Single Token Bucket Algorithm on page 992
- Determining Proper Burst Size for Traffic Policers on page 1002
- **bandwidth-limit (Policer) on page 1276**
- **bandwidth-percent on page 1278**

color-aware

Syntax color-aware;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* single-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** and **[edit dynamic-profiles ... two-rate]** hierarchy levels introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description For a three-color policer, configure the way preclassified packets are metered. In color-aware mode, the local router can assign a higher packet loss priority, but cannot assign a lower packet loss priority.

For example, suppose an upstream router assigned medium-high packet loss priority to a packet because the packet exceeded the committed information rate on the upstream router interface.

- If the local router applies color-aware policing to the packet, the router *cannot* change the packet loss priority to low, even if the packet conforms to the configured committed information route on the local router interface.
- If the local router applies color-blind policing to the packet, the router *can* change the packet loss priority to low if the packet conforms to the configured committed information route on the local router interface.

NOTE: A color-aware policer cannot be applied to Layer 2 traffic.

Default If you omit the **color-aware** statement, the default behavior is color-aware mode.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Color Modes for Three-Color Policers on page 1119
- color-blind on page 1285

color-blind

Syntax color-blind;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* single-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** and **[edit dynamic-profiles ... two-rate]** hierarchy levels introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description For a three-color policer, configure the way preclassified packets are metered. In color-blind mode, the local router ignores the preclassification of packets and can assign a higher or lower packet loss priority.

For example, suppose an upstream router assigned medium-high packet loss priority to a packet because the packet exceeded the committed information rate on the upstream router interface.

- If the local router applies color-aware policing to the packet, the router *cannot* change the packet loss priority to low, even if the packet conforms to the configured committed information route on the local router interface.

NOTE: A color-aware policer cannot be applied to Layer 2 traffic.

- If the local router applies color-blind policing to the packet, the router *can* change the packet loss priority to low if the packet conforms to the configured committed information route on the local router interface.

Default If you omit the **color-blind** statement, the default behavior is color-aware mode.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Color Modes for Three-Color Policers on page 1119
- color-aware on page 1284

committed-burst-size

Syntax committed-burst-size *bytes*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* single-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** and **[edit dynamic-profiles ... two-rate]** hierarchy levels introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description For a three-color policer, configure the committed burst size (CBS) as a number of bytes.

NOTE: When you include the **committed-burst-size** statement in the configuration, you must also include the **committed-information-rate** statement at the same hierarchy level.

In three-color policing, a committed information rate (CIR) defines the guaranteed bandwidth for traffic arriving at or departing from the interface under normal line conditions. A flow of traffic at an average rate that conforms to the CIR is categorized green.

During periods of average traffic rates below the CIR, any unused bandwidth capacity accumulates up to a maximum amount defined by the CBS. Short periods of bursting traffic (back-to-back traffic at averages rates that exceed the CIR) are also categorized as green provided that unused bandwidth capacity is available.

Traffic that exceeds both the CIR and the CBS is considered nonconforming.

Single-rate three-color policers use a *dual token bucket algorithm* to measure traffic against a single rate limit. Nonconforming traffic is categorized as yellow or red, based on the **excess-burst-size** statement included in the policer configuration.

Two-rate three-color policers use a *dual-rate dual token bucket algorithm* to measure traffic against two rate limits. Nonconforming traffic is categorized as yellow or red based on the **peak-information-rate** and **peak-burst-rate** statements included in the policer configuration.

Options *bytes*—Number of bytes. You can specify a value in bytes either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000 bytes

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Dual Token Bucket Algorithms on page 994
- Determining Proper Burst Size for Traffic Policers on page 1002
- **committed-information-rate on page 1288**
- **excess-burst-size on page 1291**
- **peak-burst-size on page 1316**
- **peak-information-rate on page 1318**

committed-information-rate

Syntax committed-information-rate *bps*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* single-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** and **[edit dynamic-profiles ... two-rate]** hierarchy levels introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description For a three-color policer, configure the committed information rate as a number of bits per second. The committed information rate (CIR) is the guaranteed bandwidth for traffic arriving at or departing from the interface under normal line conditions.

NOTE: When you include the **committed-information-rate** statement in the configuration, you must also include the **committed-burst-size** statement at the same hierarchy level.

In three-color policing, a CIR defines the guaranteed bandwidth for traffic arriving at or departing from the interface under normal line conditions. A flow of traffic at an average rate that conforms to the CIR is categorized green.

During periods of average traffic rates below the CIR, any unused bandwidth capacity accumulates up to a maximum amount defined by the committed burst size (CBS). Short periods of bursting traffic (back-to-back traffic at averages rates that exceed the CIR) are also categorized as green provided that unused bandwidth capacity is available.

Traffic that exceeds both the CIR and the CBS is considered nonconforming.

Single-rate three-color policers use a *dual token bucket algorithm* to measure traffic against a single rate limit. Nonconforming traffic is categorized as yellow or red, based on the **excess-burst-size** statement included in the policer configuration.

Two-rate three-color policers use a *dual-rate dual token bucket algorithm* to measure traffic against two rate limits. Nonconforming traffic is categorized as yellow or red based on the **peak-information-rate** and **peak-burst-rate** statements included in the policer configuration.

Options *bps*—Number of bits per second. You can specify a value in bits per second either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range:

- 1500 through 100,000,000,000 bps on EX, M, and T Series routers
- 1500 through 18,446,744,073,709,551,615 bps on Mx Series routers

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Dual Token Bucket Algorithms on page 994
- Determining Proper Burst Size for Traffic Policers on page 1002
- **committed-burst-size on page 1286**
- **excess-burst-size on page 1291**
- **peak-burst-size on page 1316**
- **peak-information-rate on page 1318**

egress-policer-overhead

Syntax egress-policer-overhead *bytes*;

Hierarchy Level [edit chassis fpc *slot-number* pic *pic-number*]

Release Information Statement introduced before Junos OS Release 11.1.

Description Add the specified number of bytes to the actual length of an Ethernet frame when determining the actions of Layer 2 policers, MAC policers, or queue rate limits applied to output traffic on the line card. You can configure egress policer overhead to account for egress *shaping* overhead bytes added to output traffic on the line card.

On M Series and T Series routers, this statement is supported on Gigabit Ethernet Intelligent Queuing 2 (IQ2) PICs and Enhanced IQ2 (IQ2E) PICs. On MX Series routers, this statement is supported for interfaces configured on Dense Port Concentrators (DPCs).

NOTE: This statement is not supported on Modular Interface Cards (MICs) or Modular Port Concentrators (MPCs) in MX Series routers.

Options *bytes*—Number of bytes added to a packet exiting an interface.

Range: 0–255 bytes

Default: 0

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *egress-shaping-overhead*
- Policer Overhead to Account for Rate Shaping Overview on page 1106
- Example: Configuring Policer Overhead to Account for Rate Shaping on page 1106
- *Configuring a Policer Overhead*
- *CoS on Enhanced IQ2 PICs Overview*

excess-burst-size

Syntax

excess-burst-size *bytes*;

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall three-color-policer *name* single-rate],
[edit firewall three-color-policer *policer-name* single-rate]

Release Information

Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... single-rate]** hierarchy level introduced in Junos Release OS 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

For a single-rate three-color policer, configure the excess burst size (EBS) as a number of bytes. The EBS allows for moderate periods of bursting traffic that exceeds both the committed information rate (CIR) and the committed burst size (CBS).

NOTE: When you include the **excess-burst-size** statement in the configuration, you must also include the **committed-burst-size** and **committed-information-rate** statements at the same hierarchy level.

Traffic that exceeds both the CIR and the CBS is considered nonconforming.

Single-rate three-color policing uses a *dual token bucket algorithm* to measure traffic against a single rate limit. Nonconforming traffic is categorized as yellow or red based on the **excess-burst-size** statement included in the policer configuration.

During periods of traffic that conforms to the CIR, any unused portion of the guaranteed bandwidth capacity accumulates in the first token bucket, up to the maximum number of bytes defined by the CBS. If any accumulated bandwidth capacity overflows the first bucket, the excess accumulates in a second token bucket, up to the maximum number of bytes defined by the EBS.

A nonconforming traffic flow is categorized yellow if its size conforms to bandwidth capacity accumulated in the first token bucket. Packets in a yellow flow are marked with **medium-high** packet loss priority (PLP) and then passed through the interface.

A nonconforming traffic flow is categorized red if its size exceeds the bandwidth capacity accumulated in the second token bucket. Packets in a red traffic flow are marked with **high** PLP and then either passed through the interface or optionally discarded.

Options

bytes—Number of bytes. You can specify a value in bytes either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000 bytes

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Dual Token Bucket Algorithms on page 994
- Determining Proper Burst Size for Traffic Policers on page 1002
- **committed-burst-size on page 1286**
- **committed-information-rate on page 1288**

filter-specific

Syntax filter-specific;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit firewall family inet prefix-action *name*],
[edit firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall family inet prefix-action *name*]

Release Information Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description By default, a policer operates in *term-specific* mode, which means that for a given firewall filterthe Junos OS creates a separate policer instance for every filter term that references the policer. You can, however, use a common policer instance for all terms within the same firewall filter by setting the *filter-specific* option in the policer. In addition, for IPv4 firewall filters with multiple terms that reference the same policer, filter-specific mode counts and monitors the activity of the policer at the firewall filter level.

NOTE: Both filter-specific and term-specific apply to prefix-specific policer sets.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Filter-Specific Policer Overview on page 1060
- Prefix-Specific Counting and Policing Overview on page 1072
- Filter-Specific Counter and Policer Set Overview on page 1074

forwarding-class (Firewall Filter Action)

Syntax forwarding-class *class-name*;

Hierarchy Level [edit firewall family *family-name* filter *filter-name* term *term-name* then],
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name* term *term-name* then]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Set the forwarding class of incoming packets.

Options ***class-name***—Name of the forwarding class.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Firewall Filter Nonterminating Actions on page 673
- Policer Color-Marking and Actions on page 990
- Multifield Classification Overview on page 1088

hierarchical-policer

List of Syntax

Syntax (M Series, MX Series, T Series - Bandwidth-Based) on page 1295
Syntax (MX Series - Packets-Per-Second (pps)-Based) on page 1295

Syntax (M Series, MX Series, T Series - Bandwidth-Based)

```
hierarchical-policer hierarchical-policer-name | uid {
  aggregate {
    if-exceeding {
      bandwidth-limit bps;
      burst-size-limit bytes;
    }
    then {
      discard;
    }
  }
  premium {
    if-exceeding {
      bandwidth-limit bps;
      burst-size-limit bytes;
    }
    then {
      discard;
    }
  }
}
```

Syntax (MX Series - Packets-Per-Second (pps)-Based)

```
hierarchical-policer hierarchical-policer-name | uid {
  aggregate {
    if-exceeding-pps {
      pps-limit pps;
      packet-burst packets;
    }
    then {
      discard;
    }
  }
  premium {
    if-exceeding-pps (Hierarchical Policer) {
      pps-limit (Hierarchical Policer) pps;
      packet-burst (Hierarchical Policer) packets;
    }
    then {
      discard;
    }
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall],
[edit firewall]

Release Information

Statement introduced in Junos OS Release 9.5.

Support at the **[edit dynamic-profiles *profile-name* firewall]** hierarchy level introduced in Junos OS Release 11.4.
Support for **if-exceeding-pps** statement on MX Series routers with MPCs introduced in Junos OS Release 15.2.

Description Use a hierarchical policer to rate-limit ingress Layer 2 traffic at a physical or logical interface and apply different policing actions based on whether the packets are classified as **premium** for expedited forwarding (EF) or **aggregate** for a lower priority. The two policers defined within the hierarchical policer are **aggregate** and **premium**.

Hierarchical policers are supported on Enhanced Intelligent Queuing (IQE) PICs and SONET interfaces hosted on the M120 and M320 with incoming Flexible PIC Concentrators (FPCs) as SFPC and outgoing FPCs as FFPC; on MPCs hosted on MX Series routers; on the T320, T640, and T1600 with Enhanced Intelligent Queuing (IQE) PICs; and on the T4000 with Type 5 FPC and Enhanced Scaling Type 4 FPC.

NOTE:

- The **if-exceeding-pps** statement is only supported on MX Series routers with MPCs.
- The **if-exceeding** and **if-exceeding-pps** statements are mutually exclusive and, therefore, cannot be applied at the same time.

Options *hierarchical-policer-name*—Name that identifies the policer. The name can contain letters, numbers, and hyphens (-), and can be up to 255 characters long. To include spaces in the name, enclose the name in quotation marks (“ ”).

uid—When you configure a hierarchical policer at the **[edit dynamic-profiles *profile name* firewall]** hierarchy level, you must assign a variable UID as the policer name.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policer Configuration Overview on page 981
- Hierarchical Policers on page 1009
- **aggregate (Hierarchical Policer) on page 1273**
- **bandwidth-limit (Hierarchical Policer) on page 1274**
- **burst-size-limit (Hierarchical Policer) on page 1280**
- **pps-limit (Hierarchical Policer) on page 1324**
- **packet-burst (Hierarchical Policer) on page 1315**
- **if-exceeding (Hierarchical Policer) on page 1298**
- *if-exceeding-pps (Hierarchical Policer)*
- **premium (Hierarchical Policer) on page 1329**

if-exceeding (Hierarchical Policer)

Syntax

```
if-exceeding {
  bandwidth-limit bps;
  burst-size-limit bytes;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall hierarchical-policer aggregate],
[edit dynamic-profiles *profile-name* firewall hierarchical-policer premium],
[edit firewall hierarchical-policer aggregate],
[edit firewall hierarchical-policer premium]

Release Information

Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... aggregate]** and **[edit dynamic-profiles ... premium]** hierarchy level introduced in Junos OS Release 11.4.

Description

For M40e, M120, and M320 (with FFPC and SFPC) edge routers and T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs, T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, specify bandwidth and burst limits for a premium or aggregate component of a hierarchical policer.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policer Configuration Overview on page 981
- Hierarchical Policers on page 1009
- aggregate (Hierarchical Policer) on page 1273
- bandwidth-limit (Hierarchical Policer) on page 1274
- burst-size-limit (Hierarchical Policer) on page 1280
- hierarchical-policer on page 1256
- premium (Hierarchical Policer) on page 1329

if-exceeding (Policer)

Syntax

```
if-exceeding {
  (bandwidth-limit bps | bandwidth-percent number);
  burst-size-limit bytes;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall policer *policer-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure rate limits for a single-rate two-color policer.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Hierarchical Policer Configuration Overview on page 981
- Basic Single-Rate Two-Color Policers on page 1032
- Bandwidth Policers on page 1051
- Filter-Specific Counters and Policers on page 1060
- Prefix-Specific Counting and Policing Actions on page 1072
- Multifield Classification on page 1088
- Policer Overhead to Account for Rate Shaping in the Traffic Manager on page 1106
- Hierarchical Policers on page 1009

ingress-policer-overhead

Syntax ingress-policer-overhead *bytes*;

Hierarchy Level [edit chassis fpc *slot-number* pic *pic-number*]

Release Information Statement introduced before Junos OS Release 11.1.
Statement introduced in Junos OS Release 15.1X49-D30 for vSRX.

Description Add the configured number of bytes to the length of a packet entering the interface.

Configure a policer overhead to control the rate of traffic received on an interface. Use this feature to help prevent denial-of-service (DoS) attacks or to enforce traffic rates to conform to the service-level agreement (SLA). When you configure a policer overhead, the configured policer overhead value (bytes) is added to the length of the final Ethernet frame. This calculated length of frame is used to determine the policer or the rate-limiting action.

Traffic policing combines the configured policy bandwidth limits and the burst size to determine how to meter the incoming traffic. If you configure a policer overhead on an interface, Junos OS adds those bytes to the length of incoming Ethernet frames. This added overhead fills each frame closer to the burst size, allowing you to control the rate of traffic received on an interface.

You can configure the policer overhead to rate-limit queues and Layer 2 and Layer 3 policers, for standalone (SA) and high-avalability (HA) deployments. The policer overhead and the shaping overhead can be configured simultaneously on an interface.

NOTE: **vSRX supports policer overhead on Layer 3 policers only.**

The policer overhead applies to all interfaces on the PIC. In the following example, Junos OS adds 10 bytes of overhead to all incoming Ethernet frames on ports ge-0/0/0 through ge-0/0/4.

```
set chassis fpc 0 pic 0 ingress-policer-overhead 10
```

NOTE: **vSRX only supports fpc 0 pic 0. When you commit the ingress-policer-overhead statement, the vSRX takes the PIC offline and then back online.**

You need to craft the policer overhead size to match your network traffic. A value that is too low will have minimal impact on traffic bursts. A value that is too high will rate-limit too much of your incoming traffic.

In this example, the policer overhead of 255 bytes is configured for ge-0/0/0 through ge-0/0/4. The firewall policer is configured to discard traffic when the burst size is over 1500 bytes. This policer is applied to ge-0/0/0 and ge 0/0/1. Junos OS adds 255 bytes to every Ethernet frame that comes into the configured ports. If, during a burst of traffic, the combined length of incoming frames and the overhead bytes exceeds 1500 bytes, the policer starts to discard further incoming traffic.

```
set chassis fpc 0 pic 0 ingress-policer-overhead 255
set interfaces ge-0/0/0 unit 0 family inet policer input overhead_policer
set interfaces ge-0/0/0 unit 0 family inet address 10.9.1.2/24
set interfaces ge-0/0/1 unit 0 family inet policer input overhead_policer
set interfaces ge-0/0/1 unit 0 family inet address 10.9.2.2/24
set firewall policer overhead_policer if-exceeding bandwidth-limit 32k
set firewall policer overhead_policer if-exceeding burst-size-limit 1500
set firewall policer overhead_policer then discard
```

Options ***bytes***—Number of bytes added to a frame entering an interface.

Range: 0–255 bytes

Default: 0

```
[edit chassis fpc 0 pic 0]
user@host# set ingress-policer-overhead 10;
```

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *ingress-shaping-overhead*
- Policer Overhead to Account for Rate Shaping Overview on page 1106
- Example: Configuring Policer Overhead to Account for Rate Shaping on page 1106
- *Configuring a Policer Overhead*
- *CoS on Enhanced IQ2 PICs Overview*

input-hierarchical-policer

Syntax input-hierarchical-policer *policer-name*;

Hierarchy Level [edit interfaces *interface-name* layer2-policer],
[edit interfaces *interface-name* unit *logical-unit-number* layer2-policer],

Release Information Statement introduced in Junos OS Release 9.5.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Apply a hierarchical policer to the Layer 2 input traffic for all protocol families at the physical or logical interface.

Options ***policer-name***—Name of the hierarchical policer.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policers on page 1009
- **layer2-policer (Hierarchical Policer) on page 1306**

input-policer

Syntax

input-policer *policer-name*;

Hierarchy Level

[edit interfaces *interface-name* unit *logical-unit-number* layer2-policer]
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* layer2-policer]

Release Information

Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Apply a single-rate two-color policer to the Layer 2 input traffic at the logical interface. The **input-policer** and **input-three-color** statements are mutually exclusive.

Options

policer-name—Name of the single-rate two-color policer that you define at the **[edit firewall]** hierarchy level.

Usage Guidelines

See *Applying Layer 2 Policers to Gigabit Ethernet Interfaces*.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Two-Color and Three-Color Policers at Layer 2 on page 1016
- *Applying Layer 2 Policers to Gigabit Ethernet Interfaces*
- *Configuring a Gigabit Ethernet Policer*
- input-three-color on page 1304
- layer2-policer on page 1305
- logical-interface-policer on page 1309
- output-policer on page 1312
- output-three-color on page 1313

input-three-color

Syntax

input-three-color *policer-name*;

Hierarchy Level

[edit interfaces *interface-name* unit *logical-unit-number* layer2-policer]
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* layer2-policer]

Release Information

Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Apply a single-rate or two-rate three-color policer to the Layer 2 input traffic at the logical interface. The **input-three-color** and **input-policer** statements are mutually exclusive.

Options

policer-name—Name of the single-rate or two-rate three-color policer.

Usage Guidelines

See *Applying Layer 2 Policers to Gigabit Ethernet Interfaces*.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Two-Color and Three-Color Policers at Layer 2 on page 1016
- *Applying Layer 2 Policers to Gigabit Ethernet Interfaces*
- *Configuring a Gigabit Ethernet Policer*
- **input-policer on page 1303**
- **layer2-policer on page 1305**
- **logical-interface-policer on page 1309**
- **output-policer on page 1312**
- **output-three-color on page 1313**

layer2-policer

Syntax

```
layer2-policer {
  input-policer policer-name;
  input-three-color policer-name;
  output-policer policer-name;
  output-three-color policer-name;
}
```

Hierarchy Level

[edit interfaces *interface-name* unit *logical-unit-number*],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*],

Release Information

Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

For 1-Gigabit Ethernet and 10-Gigabit Ethernet IQ2 and IQ2-E interfaces on M Series, MX Series, and T Series routers, and for aggregated Ethernet, Gigabit Ethernet, and 10-Gigabit Ethernet interfaces on EX Series switches, apply Layer 2 logical interface policers. The following policers are supported:

- Two-color
- Single-rate tricolor marking (srTCM)
- Two-rate tricolor marking (trTCM)

Two-color and tricolor policers are configured at the **[edit firewall]** hierarchy level.

Options

input-policer ***policer-name***—Two-color input policer to associate with the interface. This statement is mutually exclusive with the **input-three-color** statement.

input-three-color ***policer-name***—Tricolor input policer to associate with the interface. This statement is mutually exclusive with the **input-policer** statement.

output-policer ***policer-name***—Two-color output policer to associate with the interface. This statement is mutually exclusive with the **output-three-color** statement.

output-three-color ***policer-name***—Tricolor output policer to associate with the interface. This statement is mutually exclusive with the **output-policer** statement.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- *Applying Layer 2 Policers to Gigabit Ethernet Interfaces*
- *Configuring Gigabit Ethernet Two-Color and Tricolor Policers*

layer2-policer (Hierarchical Policer)

Syntax

```
layer2-policer {
  input-hierarchical-policer policer-name
}
```

Hierarchy Level

[edit interfaces *interface-name*],
[edit interfaces *interface-name* unit *logical-unit-number*],

Release Information

Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Apply a hierarchical policer to the Layer 2 input traffic for all protocol families at the physical or logical interface. The following interfaces are supported:

- SONET interfaces hosted on M40e, M120, and M320 edge routers with incoming Flexible PIC Concentrators (FPCs) as SFPC and outgoing FPCs as FFPC
- Interfaces on MX Series, T320, T640, and T1600 core routers with Enhanced Intelligent Queuing (IQE) PICs

Options

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policers on page 1009
- **input-hierarchical-policer on page 1302**
- Two-Color and Three-Color Policers at Layer 2 on page 1016

load-balance-group

Syntax

```
load-balance-group group-name {
  next-hop-group [ group-names ];
}
```

Hierarchy Level [edit firewall]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure a load-balance group.

Options ***group-name***—Name of load-balance group.

group-names—Name of next-hop groups to include in the load-balance group set.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Configuring Load-Balance Groups* in the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*

logical-bandwidth-policer

Syntax logical-bandwidth-policer;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall policer *policer-name*]

Release Information Statement introduced in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description For a policer with a bandwidth limit configured as a percentage (using the **bandwidth-percent** statement), specify that the percentage be based on the shaping rate defined on the logical interface, rather than on the media rate of the physical interface.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Bandwidth Policers on page 1051
- *Configuring Policers Based on Logical Interface Bandwidth*
- **bandwidth-percent on page 1278** statement
- **interface-specific on page 1261** statement

logical-interface-policer

Syntax logical-interface-policer;

Hierarchy Level
[edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit dynamic-profiles *profile-name* firewall three-color-policer *name*],
[edit firewall atm-policer*atm-policer-name*],
[edit firewall policer *policer-name*],
[edit firewall policer *policer-template-name*],
[edit firewall three-color-policer *policer-name*],
[edit logical-systems *logical-system-name* firewall policer *policer-name*],
[edit logical-systems *logical-system-name* firewall three-color-policer *name*]

Release Information
Statement introduced before Junos OS Release 7.4.
Support at the **[edit firewall three-color-policer *policer-name*]** hierarchy level introduced in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** and **[edit dynamic-profiles ... three-color-policer *name*]**hierarchy levels introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Configure a logical interface policer.

NOTE: Starting in Junos OS Release 12.2R2, on T Series Core Routers only, you can configure an MPLS LSP policer for a specific LSP to be shared across different protocol family types. You must include the **logical-interface-policer** statement to do so.

Required Privilege Level
firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Two-Color and Three-Color Logical Interface Policers on page 1141
- Traffic Policer Types on page 976

- *Configuring and Applying Tricolor Marking Policers*

- action on page 1272

- *Configuring Gigabit Ethernet Two-Color and Tricolor Policers*
- *action*

loss-priority (Firewall Filter Action)

Syntax loss-priority (high | low);

Hierarchy Level [edit firewall family *family-name* filter *filter-name* term *term-name* then],
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name* term *term-name* then]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Set the loss priority of incoming packets.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Firewall Filter Nonterminating Actions on page 673
- Policer Color-Marking and Actions on page 990
- Multifield Classification Overview on page 1088

loss-priority high then discard (Three-Color Policer)

Syntax loss-priority high then discard;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* action],
[edit firewall three-color-policer *policer-name* action],
[edit logical-systems *logical-system-name* firewall three-color-policer *policer-name* action]

Release Information Statement introduced before Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... action]** hierarchy level introduced in Junos OS Release 11.4.

Description For packets with high loss priority, discard the packets. The loss priority setting is implicit and is not configurable. Include this statement if you do not want the local router to forward packets that have high packet loss priority.

For single-rate three-color policers, the Junos OS assigns high loss priority to packets that exceed the committed information rate and the excess burst size.

For two-rate three-color policers, the Junos OS assigns high loss priority to packets that exceed the peak information rate and the peak burst size.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Basic Single-Rate Three-Color Policers on page 1121
- Basic Two-Rate Three-Color Policers on page 1127
- Two-Color and Three-Color Logical Interface Policers on page 1141
- Two-Color and Three-Color Physical Interface Policers on page 1154
- Two-Color and Three-Color Policers at Layer 2 on page 1016
- **action on page 1272**

output-policer

Syntax output-policer *policer-name*;

Hierarchy Level [edit interfaces *interface-name* unit *logical-unit-number* layer2-policer],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* layer2-policer]

Release Information Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Apply a single-rate two-color policer to the Layer 2 output traffic at the logical interface. The **output-policer** and **output-three-color** statements are mutually exclusive.

Options *policer-name*—Name of the single-rate two-color policer that you define at the **[edit firewall]** hierarchy level.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Two-Color and Three-Color Policers at Layer 2 on page 1016
- *Applying Layer 2 Policers to Gigabit Ethernet Interfaces*
- *Configuring a Gigabit Ethernet Policer*
- input-policer on page 1303
- input-three-color on page 1304
- layer2-policer on page 1305
- logical-interface-policer on page 1309
- output-three-color on page 1313

output-three-color

Syntax

output-three-color *policer-name*;

Hierarchy Level

[edit interfaces *interface-name* unit *logical-unit-number* layer2-policer]
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* layer2-policer]

Release Information

Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Apply a single-rate or two-rate three-color policer to the Layer 2 output traffic at the logical interface. The **output-three-color** and **output-policer** statements are mutually exclusive.

Options

policer-name—Name of the single-rate or two-rate three-color policer.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Two-Color and Three-Color Policers at Layer 2 on page 1016
- *Applying Layer 2 Policers to Gigabit Ethernet Interfaces*
- *Configuring a Gigabit Ethernet Policer*
- input-three-color on page 1304
- input-policer on page 1303
- layer2-policer on page 1305
- logical-interface-policer on page 1309
- output-policer on page 1312

packet-burst (Policer)

Syntax packet-burst *packets*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name* if-exceeding-pps],
[edit firewall policer *policer-name* if-exceeding-pps],
[edit logical-systems *logical-system-name* firewall policer *policer-name* if-exceeding-pps]

Release Information Statement introduced in Junos OS Release 16.1 for MX Series routers with MPCs.

Description For a single-rate two-color policer, configure the **packet-burst** as a number of packets. Single-rate two-color policing uses the single token bucket algorithm to measure traffic-flow conformance to a two-color policer rate limit.

Traffic at the interface that conforms to the pps-limit is categorized green. Traffic that exceeds the specified rate is also categorized as green provided that sufficient tokens remain in the single token bucket. Packets in a green flow are implicitly marked with **low** packet loss priority (PLP) and then passed through the interface.

Traffic that exceeds the specified rate when insufficient tokens remain in the single token bucket is categorized red. Depending on the configuration of the two-color policer, packets in a red traffic flow might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface.

NOTE: This statement specifies the packet burst limit as an absolute number of packets.

Single-rate two-color policing allows bursts of traffic for short periods, whereas single-rate and two-rate three-color policing allows more sustained bursts of traffic.

Hierarchical policing is a form of two-color policing that applies different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority. You apply a hierarchical policer to ingress Layer 2 traffic to allows bursts of EF traffic for short period and bursts of non-EF traffic for short periods, with EF traffic always taking precedence over non-EF traffic.

Options *packets*—Specify the number of packets either as a decimal number or as a decimal number followed by the abbreviation **k** (1000), or **m** (1000000).

Range: 1 through 24414062

Default: None

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Single Token Bucket Algorithm on page 992
- Determining Proper Burst Size for Traffic Policers on page 1002
- **bandwidth-percent on page 1278**
- **burst-size-limit (Policer) on page 1281**

packet-burst (Hierarchical Policer)

Syntax

packet-burst *packets*;

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall hierarchical-policer *hierarchical-policer-name* aggregate if-exceeding-pps],
[edit dynamic-profiles *profile-name* firewall hierarchical-policer *hierarchical-policer-name* premium if-exceeding-pps],
[edit firewall hierarchical-policer *hierarchical-policer-name* aggregate if-exceeding-pps],
[edit firewall hierarchical-policer *hierarchical-policer-name* premium if-exceeding-pps]

Release Information

Statement introduced in Junos OS Release 16.1 for MX Series routers with MPCs.

Description

On MPCs hosted on MX Series routers, configure the packet burst limit for premium or aggregate traffic in a hierarchical policer. When used in combination with the *if-exceeding-pps*and pps-limit statements, you can control the number of packets that will be allowed over a configured packets-per-second limit when traffic is in burst state.

Options

packets—Packet burst limit in packets. You can specify the number of packets either as a decimal number or as a decimal number followed by the abbreviation **k** (1000), or **m** (1000000).

Range: 1 through 24414062

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policer Configuration Overview on page 981
- Policer Color-Marking and Actions on page 990
- Single Token Bucket Algorithm on page 992
- Hierarchical Policers on page 1009
- **aggregate (Hierarchical Policer) on page 1273**
- **bandwidth-limit (Hierarchical Policer) on page 1274**
- **premium (Hierarchical Policer) on page 1329**

peak-burst-size

Syntax peak-burst-size *bytes*;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... two-rate]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description For a two-rate three-color policer, configure the peak burst size (PBS) as a number of bytes. The PBS defines the maximum number of bytes of unused peak bandwidth capacity that can be accumulated. The accumulated bandwidth allows for moderate periods of bursting traffic that exceeds the peak information rate (PIR) and the committed burst size (CBS).

NOTE: When you include the **peak-burst-size** statement in the configuration, you must also include the **committed-burst-size** and **peak-information-rate** statements at the same hierarchy level.

Two-rate three-color policers use a *dual-rate dual token bucket algorithm* to measure traffic against two rate limits.

- A traffic flow is categorized green if it conforms to both the committed information rate (CIR) and the CBS-bounded accumulation of available committed bandwidth capacity.
- A traffic flow is categorized yellow if exceeds the CIR and CBS but conforms to the PIR. Packets in a yellow flow are marked with **medium-high** packet loss priority (PLP) and then passed through the interface.
- A traffic flow is categorized red if exceeds the PIR and the PBS-bounded accumulation of available peak bandwidth capacity. Packets in a red traffic flow are marked with **high** PLP and then either passed through the interface or optionally discarded.

Options *bytes*—Number of bytes. You can specify a value in bytes either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range: 1500 through 100,000,000,000 bytes

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Dual Token Bucket Algorithms on page 994
- Determining Proper Burst Size for Traffic Policers on page 1002
- **committed-burst-size on page 1286**
- **committed-information-rate on page 1288**
- **excess-burst-size on page 1291**
- **peak-information-rate on page 1318**

peak-information-rate

Syntax
peak-information-rate *bps*;

Hierarchy Level
[edit dynamic-profiles *profile-name* firewall three-color-policer *name* two-rate],
[edit firewall three-color-policer *policer-name* two-rate]

Release Information
Statement introduced in Junos OS Release 7.4.
Support at the **[edit dynamic-profiles ... two-rate]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description
For a two-rate three-color policer, configure the peak information rate (PIR) as a number of bits per second. The PIR is the maximum rate for traffic arriving at or departing from the interface under peak line conditions. Traffic that exceeds the committed information rate (CIR) and the committed burst size (CBS) is metered to the PIR.

NOTE: When you include the **peak-information-rate** statement in the configuration, you must also include the **committed-information-rate** and **peak-burst-size** statements at the same hierarchy level.

Two-rate three-color policers use a *dual-rate dual token bucket algorithm* to measure traffic against two rate limits.

- A traffic flow is categorized green if it conforms to both the CIR and the CBS-bounded accumulation of available committed bandwidth capacity.
- A traffic flow is categorized yellow if exceeds the CIR and CBS but conforms to the PIR. Packets in a yellow flow are marked with **medium-high** packet loss priority (PLP) and then passed through the interface.
- A traffic flow is categorized red if exceeds the PIR and the PBS-bounded accumulation of available peak bandwidth capacity. Packets in a red traffic flow are marked with **high** PLP and then either passed through the interface or optionally discarded.

Options
bps—Number of bits per second. You can specify a value in bits per second either as a complete decimal number or as a decimal number followed by the abbreviation **k** (1000), **m** (1,000,000), or **g** (1,000,000,000).

Range:

- 1500 through 100,000,000,000 bps on EX, M, and T Series routers
- 1500 through 18,446,744,073,709,551,615 bps on Mx Series routers

Required Privilege Level
firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Dual Token Bucket Algorithms on page 994
- Determining Proper Burst Size for Traffic Policers on page 1002
- **committed-burst-size on page 1286**
- **committed-information-rate on page 1288**
- **excess-burst-size on page 1291**
- **peak-burst-size on page 1316**

physical-interface-filter

Syntax physical-interface-filter;

Hierarchy Level

[edit firewall family *family-name* filter *filter-name*],
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name*],
[edit routing-instances *routing-instance-name* firewall family *family-name* filter *filter-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* firewall family *family-name* filter *filter-name*]

Release Information

Statement introduced in Junos OS Release 9.6.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure a physical interface filter. Use this statement to reference a physical interface policer for the specified protocol family.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Two-Color and Three-Color Physical Interface Policers on page 1154
- **physical-interface-policer on page 1320**
- **policer (Configuring) on page 1322**

physical-interface-policer

Syntax physical-interface-policer;

Hierarchy Level [edit dynamic-profiles *profile-name* firewall policer *policer-name*],
[edit firewall policer *policer-name*],
[edit firewall three-color-policer *policer-name*],
[edit logical-system *logical-system-name* firewall policer *policer-name*],
[edit logical-system *logical-system-name* three-color-policer *policer-name*],
[edit routing-instances *routing-instance-name* firewall policer *policer-name*],
[edit routing-instances *routing-instance-name* firewall three-color-policer *policer-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* firewall policer *policer-name*],
[edit logical-systems *logical-system-name* routing-instances *routing-instance-name* firewall three-color-policer *policer-name*]

Release Information Statement introduced in Junos OS Release 9.6.
Support at the **[edit dynamic-profiles ... policer *policer-name*]** hierarchy level introduced in Junos Release OS 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Configure an aggregate policer for a physical interface.

A physical interface policer can be a two-color or three-color policer. When you apply physical interface policer, to different protocol families on the same logical interface, the protocol families share the same policer instance. This means that rate limiting is performed aggregately for the protocol families for which the policer is applied. This feature enables you to use a single policer instance to perform aggregate policing for different protocol families on the same physical interface. If you want a policer instance to be associated with a protocol family, the corresponding physical interface filter needs to be applied to that protocol family. The policer is not automatically applied to all protocol families configured on the physical interface.

In contrast, with logical interface policers there are multiple separate policer instances.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Two-Color and Three-Color Physical Interface Policers on page 1154
- physical-interface-filter on page 1319

policer (Applying to a Logical Interface)

Syntax

```
policer {
  input policer-name;
  output policer-name;
}
```

Hierarchy Level

[edit interfaces *interface-name* unit *unit-number*],
[edit interfaces *interface-name* unit *unit-number* family *family*],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *unit-number*],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *unit-number* family *family*]

Description

Apply a single-rate two-color policer—except for a physical interface policer—to Layer 3 input or output traffic at a logical interface.

- To rate-limit all traffic types, regardless of the protocol family, you can apply a logical interface policer at the logical unit level of a supported interface.
- To rate-limit traffic of a specific protocol family, you can apply a basic two-color policer, a bandwidth policer, or a logical interface policer at the protocol family level of a supported interface.

NOTE: You cannot apply a physical interface policer as part of the interface configuration. You can apply a physical interface policer by referencing the policer from a physical interface filter term.

Options

input ***policer-name***—Name of one policer to evaluate packets received on the interface.

output ***policer-name***—Name of one policer to evaluate packets transmitted on the interface.

Required Privilege Level

interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Single-Rate Two-Color Policer Overview on page 1032
- Bandwidth Policer Overview on page 1051
- Logical Interface (Aggregate) Policer Overview on page 1141

policer (Configuring)

Syntax

```
policer policer-name {
  filter-specific;
  if-exceeding {
    bandwidth-limit bps;
    bandwidth-percent number;
    burst-size-limit bytes;
  }
  logical-bandwidth-policer;
  logical-interface-policer;
  physical-interface-policer;
  shared-bandwidth-policer;
  then {
    policer-action;
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall],
[edit firewall],
[edit logical-systems *logical-system-name* firewall]

Release Information

Statement introduced before Junos OS Release 7.4.
The **out-of-profile** policer action added in Junos OS Release 8.1.
The **logical-bandwidth-policer** statement added in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
The **physical-interface-policer** statement introduced in Junos OS Release 9.6.
The **shared-bandwidth-policer** statement added in Junos OS Release 11.2.
Support at the **[edit dynamic-profiles ... firewall]** hierarchy level introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure policer rate limits and actions. When included at the **[edit firewall]** hierarchy level, the **policer** statement creates a template, and you do not have to configure a policer individually for every firewall filter or interface. To activate a policer, you must include the **policer-action** modifier in the **then** statement in a firewall filter term or on an interface.

Options

policer-action—One or more actions to take:

- **discard**—Discard traffic that exceeds the rate limits.
- **forwarding-class** *class-name*—Specify the particular forwarding class.
- **loss-priority**—Set the packet loss priority (PLP) to **low**, **medium-low**, **medium-high**, or **high**.

policer-name—Name that identifies the policer. The name can contain letters, numbers, and hyphens (-), and can be up to 255 characters long. To include spaces in the name, enclose it in quotation marks (“ ”). Policer names cannot begin with an underscore in the form __.*.

then—Actions to take on matching packets.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Bandwidth Policer Overview on page 1051
- *Configuring Firewall Filters and Policers for VPLS*
- *Configuring Multifield Classifiers*
- Logical Interface (Aggregate) Policer Overview on page 1141
- Physical Interface Policer Overview on page 1154
- Single-Rate Two-Color Policer Overview on page 1032
- *Using Multifield Classifiers to Set Packet Loss Priority*
- **filter (Configuring) on page 1252**
- *priority (Schedulers)*

policer (Firewall Filter Action)

Syntax

policer *policer-name*;

Hierarchy Level

[edit firewall family *family-name* filter *filter-name* term *term-name* then],
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name* term *term-name* then]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

For T Series routers and M320 routers with Enhanced II Flexible PIC Concentrators (FPCs) and the T640 Core Router with Enhanced Scaling FPC4, apply a tricolor marking policer.

Options

policer-name—Name of a single-rate two-color policer to use to rate-limit traffic.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Firewall Filter Nonterminating Actions on page 673
- Two-Color Policer Configuration Overview on page 1027

pps-limit (Hierarchical Policer)

Syntax

pps-limit *pps*;

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall hierarchical-policer *hierarchical-policer-name* aggregate if-exceeding],
[edit dynamic-profiles *profile-name* firewall hierarchical-policer *hierarchical-policer-name* premium if-exceeding],
[edit firewall hierarchical-policer *hierarchical-policer-name* aggregate if-exceeding],
[edit firewall hierarchical-policer *hierarchical-policer-name* premium if-exceeding]

Release Information

Statement introduced in Junos OS Release 16.1 for MX Series routers with MPCs.

Description

Configure the maximum bandwidth in packets per second (pps) for premium or aggregate traffic in a hierarchical policer.

Hierarchical policing is a form of two-color policing that applies different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority. You apply a hierarchical policer to ingress Layer 2 traffic to allow bursts of EF traffic for short periods and bursts of non-EF traffic for short periods, with EF traffic always taking precedence over non-EF traffic.

Options

pps—Specify the number of packets per second either as a decimal number or as a decimal number followed by the abbreviation **k** (1000), or **m** (1000000).

Range: 2 through 24414062

Default: None

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Hierarchical Policer Configuration Overview on page 981
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Single Token Bucket Algorithm on page 992
- Determining Proper Burst Size for Traffic Policers on page 1002
- **aggregate (Hierarchical Policer) on page 1273**
- **burst-size-limit (Hierarchical Policer) on page 1280**
- **premium (Hierarchical Policer) on page 1329**

pps-limit (Policer)

Syntax

pps-limit *pps*;

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall policer *policer-name* if-exceeding-pps],
[edit firewall policer *policer-name* if-exceeding-pps],
[edit logical-systems *logical-system-name* firewallpolicer *policer-name* if-exceeding-pps]

Release Information

Statement introduced in Junos OS Release 16.1 for MX Series routers with MPCs.

Description

For a single-rate two-color policer, configure the packets-per-second (pps) limit as a number of packets per second. Single-rate two-color policing uses the single token bucket algorithm to measure traffic-flow conformance to a two-color policer rate limit.

Traffic at the interface that conforms to the pps limit is categorized green. Traffic that exceeds the specified rate is also categorized as green provided that sufficient tokens remain in the single token bucket. Packets in a green flow are implicitly marked with **low** packet loss priority (PLP) and then passed through the interface.

Traffic that exceeds the specified rate when insufficient tokens remain in the single token bucket is categorized red. Depending on the configuration of the two-color policer, packets in a red traffic flow might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface.

NOTE: **This statement specifies the pps limit as an absolute number of packets per second. You cannot use the pps limit as a percentage of interface bandwidth.**

Single-rate two-color policing allows bursts of traffic for short periods, whereas single-rate and two-rate three-color policing allow more sustained bursts of traffic.

Hierarchical policing is a form of two-color policing that applies different policing actions based on whether the packets are classified for expedited forwarding (EF) or for a lower priority. You apply a hierarchical policer to ingress Layer 2 traffic to allow bursts of EF traffic for short periods and bursts of non-EF traffic for short periods, with EF traffic always taking precedence over non-EF traffic.

Options

pps—Specify the number of packets per second either as a decimal number or as a decimal number followed by the abbreviation **k** (1000), or **m** (1000000).

Range: 2 through 24414062

Default: None

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Two-Color Policer Configuration Overview on page 1027
- Policer Bandwidth and Burst-Size Limits on page 989
- Policer Color-Marking and Actions on page 990
- Single Token Bucket Algorithm on page 992
- Determining Proper Burst Size for Traffic Policers on page 1002
- **bandwidth-percent on page 1278**
- **burst-size-limit (Policer) on page 1281**

prefix-action (Configuring)

Syntax

```
prefix-action prefix-action-name {
  count;
  destination-prefix-length prefix-length;
  filter-specific;
  policer policer-name;
  source-prefix-length prefix-length;
  subnet-prefix-length prefix-length;
}
```

Hierarchy Level

[edit firewall family inet],
[edit logical-systems *logical-system-name* firewall family inet]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure a prefix-specific action.

Options

count—Enable counter.

destination-prefix-length *prefix-length*—Destination prefix length.

Range: 0 through 32

filter-specific—Create the prefix-specific set of policers and counters as a filter-specific set. If this option is not specified, the prefix-specific set of policers and counters are created as term-specific.

policer *policer-name*—Policer name.

source-prefix-length *prefix-length*—Source prefix length.

Range: 0 through 32

subnet-prefix-length *prefix-length*—Subnet prefix length.

Range: 0 through 32

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Prefix-Specific Counting and Policing Actions on page 1072

prefix-action (Firewall Filter Action)

Syntax prefix-action *prefix-action-name*;

Hierarchy Level [edit firewall family inet filter *filter-name* term *term-name* then],
[edit logical-systems *logical-system-name* firewall family inet filter *filter-name* term *term-name* then]

Release Information Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Reference a prefix-specific action.

Options ***prefix-action-name***—Name of a prefix-specific action to use to rate-limit traffic.

Related Documentation

- Firewall Filter Nonterminating Actions on page 673
- Prefix-Specific Counting and Policing Actions on page 1072

premium (Hierarchical Policer)

Syntax

```
premium {
  if-exceeding {
    bandwidth-limit bandwidth;
    burst-size-limit burst;
  }
  then {
    discard;
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall hierarchical-policer],
[edit firewall hierarchical-policer]

Release Information

Statement introduced in Junos OS Release 9.5.
Support at the **[edit dynamic-profiles ... hierarchical-policer *name*]** hierarchy level introduced in Junos OS Release 11.4.

Description

On M40e, M120, and M320 edge routers with FPC input as FFPC and FPC output as SFPC, and on MX Series, T320, T640, and T1600 edge routers with Enhanced Intelligent Queuing (IQE) PICs, T4000 routers with Type 5 FPC and Enhanced Scaling Type 4 FPC, specify a premium level for a hierarchical policer.

Options

Options are described separately.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Applying Policers*
- Guidelines for Applying Traffic Policers on page 983
- Hierarchical Policer Configuration Overview on page 981
- Hierarchical Policers on page 1009
- **aggregate (Hierarchical Policer) on page 1273**
- **bandwidth-limit (Hierarchical Policer) on page 1274**
- **burst-size-limit (Hierarchical Policer) on page 1280**
- **hierarchical-policer on page 1256**
- **if-exceeding (Hierarchical Policer) on page 1298**

shared-bandwidth-policer (Configuring)

Syntax shared-bandwidth-policer;

Hierarchy Level [edit firewall policer *policer-name*],
[edit firewall three-color-policer *policer-name*],
[edit firewall hierarchical-policer *policer-name*]

Release Information Statement introduced in Junos OS Release 11.2.
Support for MX Series MPC and MIC interfaces added in Junos OS Release 12.1.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description Policer instances share bandwidth. This enables configuration of interface-specific policers applied on an aggregated Ethernet bundle or an aggregated SONET bundle to match the effective bandwidth and burst-size to user-configured values. This feature is supported on the following platforms: T Series routers, M120, M10i, M7i (CFEB-E only), M320 (SFPC only), MX240, MX480, and MX960 with DPC, MIC, and MPC interfaces and EX Series switches.

NOTE: This statement is not supported on T4000 Type 5 FPCs.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Policer Support for Aggregated Ethernet Interfaces Overview on page 984

single-rate

Syntax

```
single-rate {
  (color-aware | color-blind);
  committed-information-rate bps;
  committed-burst-size bytes;
  excess-burst-size bytes;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall three-color-policer *name*],
[edit firewall three-color-policer *policer-name*],
[edit logical-systems *logical-system-name* firewall three-color-policer *policer-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... three-color-policer *name*]** hierarchy level introduced in Junos OS Release 11.4.

Description

Configure a single-rate three-color policer in which marking is based on the committed information rate (CIR), committed burst size (CBS), and excess burst size (EBS).

Packets that conform to the CIR or the CBS are assigned low loss priority (green). Packets that exceed the CIR and the CBS but are within the EBS are assigned medium-high loss priority (yellow). Packets that exceed the EBS are assigned high loss priority (red).

Green and yellow packets are always forwarded; this action is not configurable. You can configure red packets to be discarded. By default, red packets are forwarded.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- color-aware on page 1284
- color-blind on page 1285
- two-rate on page 1335

three-color-policer (Applying)

Syntax

```
three-color-policer {
  (single-rate | two-rate) policer-name;
}
```

Hierarchy Level

[edit firewall family *family-name* filter *filter-name* term *term-name* then]
[edit logical-systems *logical-system-name* firewall family *family-name* filter *filter-name* term *term-name* then]

Release Information

Statement introduced in Junos OS Release 7.4.
single-rate statement added in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Apply a tricolor marking policer.

Options

single-rate—Named tricolor policer is a single-rate policer.

two-rate—Named tricolor policer is a two-rate policer.

policer-name—Name of a tricolor policer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Configuring and Applying Tricolor Marking Policers*
- Firewall Filter Nonterminating Actions on page 673
- Three-Color Policer Configuration Overview on page 1115

three-color-policer (Configuring)

Syntax

```
three-color-policer policer-name | uid {
  action {
    loss-priority high then discard;
  }
  filter-specific;
  logical-interface-policer;
  physical-interface-policer;
  shared-bandwidth-policer;
  single-rate {
    (color-aware | color-blind);
    committed-burst-size bytes;
    committed-information-rate bps;
    excess-burst-size bytes;
  }
  two-rate {
    (color-aware | color-blind);
    committed-burst-size bytes;
    committed-information-rate bps;
    peak-burst-size bytes;
    peak-information-rate bps;
  }
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall],
[edit firewall],
[edit logical-systems *logical-system-name* firewall]

Release Information

Statement introduced before Junos OS Release 7.4.
The **action** and **single-rate** statements added in Junos OS Release 8.2.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... firewall]** hierarchy level introduced in Junos OS Release 11.4.

Description

Configure a three-color policer.

Options

policer-name—Name of the three-color policer. Reference this name when you apply the policer to an interface.

uid—When you configure a policer at the **[edit dynamic-profiles]** hierarchy level, you must assign a variable UID as the policer name.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- *Configuring and Applying Tricolor Marking Policers*
- Three-Color Policer Configuration Guidelines on page 1118
- Basic Single-Rate Three-Color Policers on page 1121
- Basic Two-Rate Three-Color Policers on page 1127
- Two-Color and Three-Color Logical Interface Policers on page 1141
- Two-Color and Three-Color Physical Interface Policers on page 1154
- Two-Color and Three-Color Policers at Layer 2 on page 1016

two-rate

Syntax

```
two-rate {
  (color-aware | color-blind);
  committed-information-rate bps;
  committed-burst-size bytes;
  peak-information-rate bps;
  peak-burst-size bytes;
}
```

Hierarchy Level

[edit dynamic-profiles *profile-name* firewall three-color-policer *name*],
[edit firewall three-color-policer *policer-name*],
[edit logical-systems *logical-system-name* firewall three-color-policer *policer-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Logical systems support introduced in Junos OS Release 9.3.
Support at the **[edit dynamic-profiles ... three-color-policer *name*** hierarchy levels introduced in Junos OS Release 11.4.
Statement introduced in Junos OS Release 12.3R2 for EX Series switches.

Description

Configure a two-rate three-color policer in which marking is based on the committed information rate (CIR), committed burst size (CBS), peak information rate (PIR), and peak burst size (PBS).

Packets that conform to the CIR or the CBS are assigned low loss priority (green). Packets that exceed the CIR and the CBS but are within the PIR or the PBS are assigned medium-high loss priority (yellow). Packets that exceed the PIR and the PBS are assigned high loss priority (red).

Green and yellow packets are always forwarded; this action is not configurable. You can configure red packets to be discarded. By default, red packets are forwarded.

The remaining statements are explained separately. See CLI Explorer.

Required Privilege Level

firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Three-Color Policer Configuration Overview on page 1115
- color-aware on page 1284
- color-blind on page 1285
- single-rate on page 1331

policer-overhead-adjustment

Syntax

```
interfaces interface-name
   unit unit-number
policer-overhead <ingress | egress> <adjustment in-bytes>
policer-overhead <adjustment in-bytes>
```

Hierarchy Level [edit interfaces *interface-name* unit *logical-unit-number*]

Release Information Statement introduced in Junos OS Release 16.1.

Description Add the configured number of bytes to the length of a packet entering the interface or leaving the interface. The policer overhead adjustment at per IFL or direction granularity is in the range of -16 bytes to +16 bytes.

NOTE: If you configure the **policer-overhead** adjustment statement without the ingress or egress option and only with values in the range of -16 bytes to +16 bytes, it is considered for both ingress and egress.

For example: set interfaces xe-1/0/1 policer-overhead <*value*>

Options *adjustment-in-bytes*—Policer overhead bytes to be accounted in ingress and egress (-16 to 16 bytes). Number of bytes used to adjust the length of a packet used for policing purposes entering an interface and leaving an interface.

Range: -16 to +16 bytes

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration

Related Documentation

- Accounting of the Policer Overhead Attribute at the Interface Level on page 12
- *Configuring the Accounting of Policer Overhead in Interface Statistics*

CHAPTER 34

Operational Commands

Routing Policy Operational Commands

- clear interfaces statistics
- clear policy statistics
- show accounting profile
- show interfaces destination-class
- show interfaces source-class
- show interfaces statistics
- show policy
- show policy conditions
- show policy damping
- show route
- show route active-path
- show route advertising-protocol
- show route all
- show route aspath-regex
- show route best
- show route brief
- show route community
- show route community-name
- show route damping
- show route detail
- show route exact
- show route export
- show route extensive
- show route flow validation

- show route forwarding-table
- show route hidden
- show route inactive-path
- show route inactive-prefix
- show route instance
- show route next-hop
- show route no-community
- show route output
- show route protocol
- show route receive-protocol
- show route table
- show route terse
- show validation database
- show validation group
- show validation replication database
- show validation session
- show validation statistics
- test policy

clear interfaces statistics

Syntax clear interfaces statistics (all | *interface-name*)

Release Information Command introduced before Junos OS Release 7.4.

Description Set interface statistics to zero. If you issue the **clear interfaces statistics** ***interface-name*** command and then perform a graceful Routing Engine switchover, the interface statistics are not cleared on the new master. Reissue the command to clear the interface statistics again.

Starting in Junos OS Release 17.3R1, this command supports the clearing of Packet Forwarding Engine accounting statistics on logical interfaces configured with accounting options. On these interfaces, the current statistics values are stored as the new current baseline values and then the counters are reset to zero. If the **allow-clear** statement is included in the interface profile, then the cleared statistics values are reported to the accounting options flat file associated with the interface. Reporting is disabled by default; if **allow-clear** is not configured, then the CLI displays cleared statistics counters, but they are not reported to the flat file.

Options **all**—Set statistics on all interfaces to zero.

interface-name—Set statistics on a particular interface to zero.

Required Privilege Level clear

List of Sample Output clear interfaces statistics on page 1339

Output Fields When you enter this command, you are provided no feedback on the status of your request.

Sample Output

clear interfaces statistics

```
user@host> clear interfaces statistics
```

clear policy statistics

Syntax

clear policy statistics
<*policy -name*>

Release Information

Command introduced in Junos OS Release 15.1F6.

Description

Clear policy statistics.

Options

none—Clear statistics for all policies.

policy-name—(Optional) Clear statistics for the specified policy only.

Required Privilege Level

clear

Related Documentation

- show policy on page 1364
- test policy on page 1587

List of Sample Output

clear policy statistics on page 1340

Output Fields

When you enter this command, you are provided feedback on the status of your request.

Sample Output

clear policy statistics

```
user@host> clear policy statistics
```

show accounting profile

Syntax show accounting profile *profile-name*

Release Information Command introduced before Junos OS Release 7.4.

Description Display accounting profile information.

Options ***profile-name***—Name of the accounting profile.

Required Privilege Level view

List of Sample Output
show accounting profile (Interface) on page 1342
show accounting profile (Filter) on page 1343
show accounting profile (Destination Class) on page 1343
show accounting profile (Routing Engine) on page 1344

Output Fields Table 74 on page 1341 lists the output fields for the **show accounting profile** command. Output fields are listed in the approximate order in which they appear.

Table 74: show accounting profile Output Fields

Field Name	Field Description
Profile	Name of the accounting profile.
Sampling interval	Configured interval, in minutes, for statistic collection.
Profile Usage Count	Number of items configured for collecting accounting statistics.
file information	Information about the accounting profile log, including: • **File**—Name of accounting profile log. If no name is explicitly provided, the name of the accounting profile is used. All statistics files are placed in the **/var/log** directory. • **maximum size**—Configured size. When the size is exceeded, the log file closes and a new log file opens. • **maximum number**—Configured maximum number of log files. • **bytes written**—Number of bytes written to the log file.
Transfer Interval	Length of time (in minutes) the file remains open, receiving statistics before it is closed, transferred, and rotated. When either the time or the file size is exceeded, the file is closed and a new one opened, whether or not a transfer site is specified.
Next Scheduled Transfer	Time at which the next transfer occurs.

Table 74: show accounting profile Output Fields *(continued)*

Field Name	Field Description
Column Labels	Names of sampled statistics. This list varies depending on the configuration: • **profile-layout**—List of data fields reported, in the order they appear in the output. • **epoch-timestamp**—Number of seconds since the epoch. • **interfaces**—(For interface, filter, and destination class profiles) Name of the interfaces on which the filter is applied. • **filter-name**—(For filter profiles) Name of the filter. • **counter-name**—(For filter profiles) Name of the counter. • **packet-count**—(For filter and destination class profiles) Number of packets for the counter. • **byte-count**—(For filter and destination class profiles) Number of bytes for the counter. • **input-bytes**—(For interface profiles) Input bytes. • **input-errors**—(For interface profiles) Generic input error packets. • **input-multicast**—(For interface profiles) Input packets arriving by multicast. • **input-packets**—(For interface profiles) Input packets. • **input-unicast**—(For interface profiles) Input unicast packets. • **output-bytes**—(For interface profiles) Output bytes. • **output-errors**—(For interface profiles) Generic output error packets. • **output-multicast**—(For interface profiles) Output packets sent by multicast. • **output-packets**—(For interface profiles) Output packets. • **output-unicast**—(For interface profiles) Output unicast packets. • **no-proto**—(For interface profiles) Packets for unsupported protocol. • **snmp-index**—(For interface profiles) SNMP index. • **destination-class-name**—(For destination class profiles) Configured destination class name. • **host name**—(For Routing Engine profiles) Hostname for the router. • **date-*yyyymmdd***—(For Routing Engine profiles) Date. • **timeofday-*hhmmss***—(For Routing Engine profiles) Time of day. • **uptime**—(For Routing Engine profiles) Time since the last reboot, in seconds. • **cpu1min**—(For Routing Engine profiles) Average system load over the last 1 minute. • **cpu5min**—(For Routing Engine profiles) Average system load over the last 5 minutes. • **cpu15min**—(For Routing Engine profiles) Average system load over the last 15 minutes.
Interface name	Name of the interface configured for this accounting profile.
Filter name	Name of the filter configured for this accounting profile.
routing-engine-stats	Routing Engine accounting profile.
Next Scheduled Collection	Time for next collection of statistics for the named interface.

Sample Output

show accounting profile (Interface)

`user@host>` **show accounting profile if_prof**

```
Profile if_prof
Sampling interval: 1 minute(s),  Profile Usage Count: 2
File accounting_profile_stats: maximum size 1048576, maximum number 5, bytes
written 2196
Transfer Interval: 15 minute(s),  Next Scheduled Transfer: 2001-06-17-18:00:45
Column Labels:
  profile-layout
  epoch-timestamp
  interface-name
  snmp-index
  input-bytes
  output-bytes
  input-packets
  output-packets
  input-unicast
  output-unicast
  input-multicast
  output-multicast
  no-proto
  input-errors
  output-errors


Interface Name              Next Scheduled Collection
fxp0.0                      2001-06-18-18:00:30
fxp0                        2001-06-18-18:01:00
```

show accounting profile (Filter)

```
user@host> show accounting profile filter_profile
Profile filter_profile
Sampling interval: 1 minute(s),  Profile Usage Count: 0
File accounting_profile_stats: maximum size 1048576, maximum number 5, bytes
written 822
Transfer Interval: 15 minute(s),  Next Scheduled Transfer: 2001-06-17-18:00:46
Column Labels:
  profile-layout
  epoch-timestamp
  interfaces
  filter-name
  counter-name
  packet-count
  byte-count


Filter Name                 Next Scheduled Collection
myfiltero                         2001-06-03-04:32:59
```

show accounting profile (Destination Class)

```
user@host> show accounting profile dcu1
Profile dcu1
Sampling interval: 1 minute(s),  Profile Usage Count: 0
File accounting_profile_stats: maximum size 1048576, maximum number 5, bytes
written 901
Transfer Interval: 15 minute(s),  Next Scheduled Transfer: 2001-06-17-18:00:46
Column Labels:
  profile-layout
  epoch-timestamp
```

```
  interface-name
  destination-class-name
  packet-count
  byte-count


Interface Name              Next Scheduled Collection
so-0/3/3                    2001-06-03-04:34:00
```

show accounting profile (Routing Engine)

```
user@host> show accounting profile rep1
Profile rep1
Sampling interval: 1 minute(s),  Profile Usage Count: 1
File accounting_profile_stats: maximum size 1048576, maximum number 5, bytes
written 901
Transfer Interval: 15 minute(s),  Next Scheduled Transfer: 2001-06-17-18:00:46
Column Labels:
  profile-layout
  epoch-timestamp
  hostname
  date-yyyymmdd
  timeofday-hhmmss
  uptime
  cpu1min
  cpu5min
  cpu15min


Interface Name              Next Scheduled Collection
routing-engine-stats        2001-06-18-18:02:31
```


show interfaces destination-class

Syntax

show interfaces destination-class
(all | *destination-class-name logical-interface-name*)

Release Information

Command introduced before Junos OS Release 7.4.
all option introduced in Junos OS Release 8.0.

Description

Display information about interfaces grouped by destination class.

Options

all—Display information about all configured destination classes.

destination-class-name—Name of a logical grouping of prefixes that count packets having the destination address matching those prefixes. Whenever a destination class is specified, you must also specify a particular logical interface, not all interfaces.

logical interface-name—Name of a logical interface.

Additional Information

For interfaces that carry IPv4, IPv6, or Multiprotocol Label Switching (MPLS) traffic, you can maintain packet counts based on the entry and exit points for traffic passing through your network. Entry and exit points are identified by source and destination prefixes grouped into sets defined as source classes and destination classes. For more information, see the *Junos OS Network Interfaces Library for Routing Devices*.

Required Privilege Level

view

List of Sample Output

show interfaces destination-class all on page 1346

Output Fields

Table 75 on page 1345 lists the output fields for the **show interfaces destination-class** command. Output fields are listed in the approximate order in which they appear.

Table 75: show interfaces destination-class Output Fields

Field Name	Field Description
Logical interface	Name of the logical interface.
Destination class	Name of destination class usage (DCU) counters per class for this interface.
Packets	Packets going to designated user-selected prefixes.
Bytes	Bytes going to designated user-selected prefixes.

Sample Output

show interfaces destination-class all

```
user@host> show interfaces destination-class all
  Logical interface .local..1

  Logical interface .local..2

  Logical interface fxp0.0

  Logical interface fxp1.0

  Logical interface lo0.16384

  Logical interface lo0.16385

  Logical interface lo0.0

  Logical interface .local..3

  Logical interface .local..4

  Logical interface .local..5

  Logical interface .local..6

  Logical interface .local..7

  Logical interface .local..8

  Logical interface .local..9

  Logical interface .local..10

  Logical interface lo0.3

  Logical interface pfh-0/0/0.16383

  Logical interface fe-0/1/0.0
    Protocol inet
                                            Packets                Bytes
      Destination class           (packet-per-second)    (bits-per-second)

                          SILVER1                  0                    0
                                  (                0) (                 0)
                          SILVER2                  0                    0
                                  (                0) (                 0)
                          SILVER3                  0                    0
                                  (                0) (                 0)
                          v4-dest                  0                    0
                                  (                0) (                 0)
    Protocol inet6
                                            Packets                Bytes
      Destination class           (packet-per-second)    (bits-per-second)

                          SILVER1                  0                    0
                                  (                0) (                 0)
                          SILVER2                  0                    0
                                  (                0) (                 0)
```

```
                        SILVER3                    0                    0
                                (                  0) (                 0)
                        v4-dest                    0                    0
                                (                  0) (                 0)

Logical interface fe-0/1/1.0

Logical interface fe-0/1/2.0
  Description: CE1-to-PE2

Logical interface ge-0/3/0.0
  Description: CE1-to-PE1

Logical interface ge-0/3/2.0
  Description: CE2-to-PE1

Logical interface pc-0/3/0.16383

Logical interface lt-1/2/0.3
  Description: LS3->LS2

Logical interface lt-1/2/0.5
  Description: LS3->LS1

Logical interface sp-1/2/0.16383
```

show interfaces source-class

Syntax

show interfaces source-class
(all | *destination-class-name logical-interface-name*)

Release Information

Command introduced before Junos OS Release 7.4.
all option introduced in Junos OS Release 8.0.

Description

Display information about interfaces grouped by source class.

Options

all—Display information about all configured source classes.

source-class-name—Name of a logical grouping of prefixes that count packets having the source address matching those prefixes.

interface-name—Name of a logical interface.

Additional Information

For interfaces that carry IPv4, IPv6, or Multiprotocol Label Switching (MPLS) traffic, you can maintain packet counts based on the entry and exit points for traffic passing through your network. Entry and exit points are identified by source and destination prefixes grouped into sets defined as source classes and destination classes. For more information, see the *Junos OS Network Interfaces Library for Routing Devices*.

Required Privilege Level

view

List of Sample Output

show interfaces source-class all on page 1349

Output Fields

Table 76 on page 1348 lists the output fields for the **show interfaces source-class** command. Output fields are listed in the approximate order in which they appear.

Table 76: show interfaces source-class Output Fields

Field Name	Field Description
Logical interface	Name of the logical interface.
Source class	Source class usage (SCU) counters per class for this interface.
Packets	Packets going to designated user-selected prefixes.
Bytes	Bytes going to designated user-selected prefixes.

Sample Output

show interfaces source-class all

```
user@host> show interfaces source-class all
 Logical interface .local..1

  Logical interface .local..2

  Logical interface fxp0.0

  Logical interface fxp1.0

  Logical interface lo0.16384

  Logical interface lo0.16385

  Logical interface lo0.0

  Logical interface .local..3

  Logical interface .local..4

  Logical interface .local..5

  Logical interface .local..6

  Logical interface .local..7

  Logical interface .local..8

  Logical interface .local..9

  Logical interface .local..10

  Logical interface lo0.3

  Logical interface pfh-0/0/0.16383

  Logical interface fe-0/1/0.0

  Logical interface fe-0/1/1.0
    Protocol inet
                                         Packets                  Bytes
      Source class              (packet-per-second)    (bits-per-second)

                         GOLD1                    0                     0
                               (                  0) (                  0)
                         GOLD2                    0                     0
                               (                  0) (                  0)
                         GOLD3                    0                     0
                               (                  0) (                  0)
                        v4-src                    0                     0
                               (                  0) (                  0)
    Protocol inet6
                                         Packets                  Bytes
      Source class              (packet-per-second)    (bits-per-second)

                         GOLD1                    0                     0
                               (                  0) (                  0)
```



```
                          GOLD2                  0                     0
                                (                0) (                  0)
                          GOLD3                  0                     0
                                (                0) (                  0)
                         v4-src                  0                     0
                                (                0) (                  0)

Logical interface fe-0/1/2.0
  Description: CE1-to-PE2

Logical interface ge-0/3/0.0
  Description: CE1-to-PE1

Logical interface ge-0/3/2.0
  Description: CE2-to-PE1

Logical interface pc-0/3/0.16383

Logical interface lt-1/2/0.3
  Description: LS3->LS2

Logical interface lt-1/2/0.5
  Description: LS3->LS1

Logical interface sp-1/2/0.16383
```

show interfaces statistics

Syntax

show interfaces statistics *interface-name*
<satellite-device [*device-alias-name* |all]>
<detail>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 12.1X48 for PTX Series Packet Transport Routers.
Command introduced in Junos OS Release 12.2 for ACX Series Routers.
satellite-device option introduced in Junos OS Release 14.2R3.

Description

Display static interface statistics, such as errors.

NOTE: When the **show interfaces statistics** command is executed on an interface that is configured on T4000 Type 5 FPC, the *IPv6 transit statistics* field displays:

- Total statistics (sum of transit and local statistics) at the physical interface level
- Transit statistics at the logical interface level

Options

interface-name—Name of an interface.

satellite-device [*device-alias-name* | all]—(Junos Fusion only) (Optional) Display interface statistics for interfaces on the specified satellite device in the Junos Fusion, or on all satellite devices in the Junos Fusion.

NOTE: In a Junos Fusion Enterprise, logical interface statistics are not synced across aggregation devices in a dual-aggregation device topology.

detail—(Optional) Display detailed output.

Required Privilege Level

view

Related Documentation

- clear interfaces statistics on page 1339

List of Sample Output

show interfaces statistics (Fast Ethernet) on page 1352
show interfaces statistics (Gigabit Ethernet PIC—Egress) on page 1353

show interfaces statistics detail (Aggregated Ethernet) on page 1354
show interfaces statistics detail (Aggregated Ethernet—Ingress) on page 1356
show interfaces statistics detail (Aggregated Ethernet—Egress) on page 1357
show interfaces statistics (SONET/SDH) on page 1358
show interfaces statistics (Aggregated SONET/SDH—Ingress) on page 1359
show interfaces statistics (Aggregated SONET/SDH—Egress) on page 1360
show interfaces statistics (MX Series Routers) on page 1361
show interfaces statistics (PTX Series Packet Transport Routers) on page 1362
show interfaces statistics (ACX Series routers) on page 1362

Output Fields Output from both the **show interfaces** ***interface-name*** **detail** and the **show interfaces** ***interface-name*** **extensive** commands include all the information displayed in the output from the **show interfaces statistics** command. For more information, see the particular interface type in which you are interested. For information about destination class and source class statistics, see the “Destination Class Field” section and the “Source Class Field” section under *Common Output Fields Description*. For information about the input errors and output errors, see *Fast Ethernet and Gigabit Ethernet Counters*.

Sample Output

show interfaces statistics (Fast Ethernet)

```
user@host> show interfaces fe-1/3/1 statistics
Physical interface: fe-1/3/1, Enabled, Physical link is Up
  Interface index: 144, SNMP ifIndex: 1042
  Description: ford fe-1/3/1
  Link-level type: Ethernet, MTU: 1514, Speed: 100mbps, Loopback: Disabled,
  Source filtering: Disabled, Flow control: Enabled
  Device flags   : Present Running
  Interface flags: SNMP-Traps Internal: 0x4000
  CoS queues     : 4 supported, 4 maximum usable queues
  Current address: 00:00:5E:00:53:dc, Hardware address: 00:00:5E:00:53:dc
  Last flapped   : 2006-04-18 03:08:59 PDT (00:01:24 ago)
  Statistics last cleared: Never
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)
  Input errors: 0, Output errors: 0
  Active alarms  : None
  Active defects : None
  Logical interface fe-1/3/1.0 (Index 69) (SNMP ifIndex 50)
    Flags: SNMP-Traps Encapsulation: ENET2
    Protocol inet, MTU: 1500
      Flags: Is-Primary, DCU, SCU-in
                                              Packets                Bytes
      Destination class            (packet-per-second)    (bits-per-second)
                          silver1                    0                    0
                                  (                  0) (                 0)
                          silver2                    0                    0
                                  (                  0) (                 0)
                          silver3                    0                    0
                                  (                  0) (                 0)
      Addresses, Flags: Is-Default Is-Preferred Is-Primary
        Destination: 10.27.245/24, Local: 10.27.245.2,
        Broadcast: 10.27.245.255
    Protocol iso, MTU: 1497
      Flags: Is-Primary
```


show interfaces statistics (Gigabit Ethernet PIC—Egress)

```
user@host> show interfaces ge-5/2/0 statistics detail
Physical interface: ge-5/2/0, Enabled, Physical link is Up
  Interface index: 146, SNMP ifIndex: 519, Generation: 149
  Link-level type: Ethernet, MTU: 1514, Speed: 1000mbps, BPDU Error: None,
MAC-REWRITE Error: None, Loopback: Disabled,
  Source filtering: Disabled, Flow control: Enabled, Auto-negotiation: Enabled,
Remote fault: Online
  Device flags   : Present Running
  Interface flags: SNMP-Traps Internal: 0x4000
  Link flags     : None
  CoS queues     : 8 supported, 8 maximum usable queues
  Hold-times     : Up 0 ms, Down 0 ms
  Current address: 00:00:5E:00:53:74, Hardware address: 00:00:5E:00:53:74
  Last flapped   : 2009-11-11 11:24:00 PST (09:23:08 ago)
  Statistics last cleared: 2009-11-11 17:50:58 PST (02:56:10 ago)
  Traffic statistics:
   Input  bytes  :               271524                    0 bps
   Output bytes  :             37769598                  352 bps
   Input  packets:                 3664                    0 pps
   Output packets:               885790                    0 pps
   IPv6 transit statistics:
    Input  bytes  :                   0
    Output bytes  :            16681118
    Input  packets:                   0
    Output packets:              362633
   Multicast statistics:
    IPV4 multicast statistics:
     Input  bytes  :               112048                    0 bps
     Output bytes  :             20779920                    0 bps
     Input  packets:                 1801                    0 pps
     Output packets:               519498                    0 pps
    IPV6 multicast statistics:
     Input  bytes  :               156500                    0 bps
     Output bytes  :             16681118                    0 bps
     Input  packets:                 1818                    0 pps
     Output packets:               362633                    0 pps
  Input errors:
    Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Policed discards: 0, L3
incompletes: 0, L2 channel errors: 0,
    L2 mismatch timeouts: 0, FIFO errors: 0, Resource errors: 0
  Output errors:
    Carrier transitions: 0, Errors: 0, Drops: 0, Collisions: 0, Aged packets: 0,
 FIFO errors: 0, HS link CRC errors: 0, MTU errors: 0,
    Resource errors: 0
  Egress queues: 8 supported, 4 in use
  Queue counters:       Queued packets  Transmitted packets      Dropped packets

    0 best-effort               882558               882558                    0

    1 expedited-fo                   0                    0                    0

    2 assured-forw                   0                    0                    0

    3 network-cont                3232                 3232                    0

  Active alarms  : None
  Active defects : None
```



```
  Logical interface ge-5/2/0.0 (Index 71) (SNMP ifIndex 573) (Generation 135)
    Flags: SNMP-Traps 0x4000 Encapsulation: ENET2
    Egress account overhead: 100
    Ingress account overhead: 90
    Traffic statistics:
     Input  bytes  :              271524
     Output bytes  :            37769598
     Input  packets:                3664
     Output packets:              885790
     IPv6 transit statistics:
      Input  bytes  :                  0
      Output bytes  :           16681118
      Input  packets:                  0
      Output packets:             362633
    Local statistics:
     Input  bytes  :              271524
     Output bytes  :              308560
     Input  packets:                3664
     Output packets:                3659
    Transit statistics:
     Input  bytes  :                   0                   0 bps
     Output bytes  :            37461038                   0 bps
     Input  packets:                   0                   0 pps
     Output packets:              882131                   0 pps
     IPv6 transit statistics:
      Input  bytes  :                  0
      Output bytes  :           16681118
      Input  packets:                  0
      Output packets:             362633
    Multicast statistics:
     IPV4 multicast statistics:
      Input  bytes  :             112048                   0 bps
      Output bytes  :           20779920                   0 bps
      Input  packets:               1801                   0 pps
      Output packets:             519498                   0 pps
     IPV6 multicast statistics:
      Input  bytes  :             156500                   0 bps
      Output bytes  :           16681118                   0 bps
      Input  packets:               1818                   0 pps
      Output packets:             362633                   0 pps
    Protocol inet, MTU: 1500, Generation: 151, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.40.40.0/30, Local: 10.40.40.2, Broadcast: 10.40.40.3,
Generation: 167
    Protocol inet6, MTU: 1500, Generation: 152, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: ::10.40.40.0/126, Local: ::10.40.40.2
    Generation: 169
      Addresses, Flags: Is-Preferred
        Destination: fe80::/64, Local: fe80::21d:b5ff:fe61:d974
    Protocol multiservice, MTU: Unlimited, Generation: 171
    Generation: 153, Route table: 0
      Policer: Input: __default_arp_policer__
```

show interfaces statistics detail (Aggregated Ethernet)

```
user@host> show interfaces ae0 detail
Physical interface: ae0, Enabled, Physical link is Up
  Interface index: 186, SNMP ifIndex: 111, Generation: 187
  Link-level type: Ethernet, MTU: 1514, Speed: 2000mbps, Loopback: Disabled,
```

```
Source filtering: Disabled, Flow control: Disabled, Minimum links needed: 1,
Minimum bandwidth needed: 0
Device flags   : Present Running
Interface flags: SNMP-Traps Internal: 0x4000
Current address: 00:00:5E:0053:f0, Hardware address: 00:00:5E:00:53:f0
Last flapped   : Never
Statistics last cleared: 2006-12-23 03:04:16 PST (01:16:24 ago)
Traffic statistics:
 Input  bytes  :              28544                    0 bps
 Output bytes  :              39770                    0 bps
 Input  packets:                508                    0 pps
 Output packets:                509                    0 pps
 Input  bytes  :         IPv6 28544
 Output bytes  :             IPv6 0
 Input  packets:           IPv6 508
 Output packets:             IPv6 0
Input errors:
  Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0,
  Policed discards: 0, Resource errors: 0
Output errors:
  Carrier transitions: 0, Errors: 0, Drops: 0, MTU errors: 0,
  Resource errors: 0

Logical interface ae0.0 (Index 67) (SNMP ifIndex 139) (Generation 145)
  Flags: SNMP-Traps Encapsulation: ENET2
  Statistics        Packets        pps         Bytes          bps
  Bundle:
      Input :           508          0        28544           0
      Output:           509          0        35698           0
  Link:
    ge-3/3/8.0
      Input :           508          0        28544           0
      Output:             0          0            0           0
    ge-3/3/9.0
      Input :             0          0            0           0
      Output:             0          0            0           0
  Marker Statistics:   Marker Rx     Resp Tx   Unknown Rx   Illegal Rx
    ge-3/3/8.0                 0           0            0            0
    ge-3/3/9.0                 0           0            0            0
Egress queues: 8 supported, 8 in use
Queue counters:       Queued packets  Transmitted packets      Dropped packets

  0 best-effort                   0                    0                    0

  1 expedited-fo                  0                    0                    0

  2 assured-forw                  0                    0                    0

  3 network-cont                  0                    0                    0

  Protocol inet, MTU: 1500, Generation: 166, Route table: 0
    Flags: None
    Addresses, Flags: Is-Preferred Is-Primary
      Destination: 10.1.1/24, Local: 10.1.1.1, Broadcast: 10.1.1.255,
      Generation: 159
  Protocol inet6, MTU: 1500, Generation: 163, Route table: 0
    Flags: Is-Primary
    Addresses, Flags: Is-Preferred
      Destination: fe80::/64, Local: fe80::206:5bff:fe05:c321,
      Broadcast: Unspecified, Generation: 161
```


show interfaces statistics detail (Aggregated Ethernet—Ingress)

```
user@host> show interfaces statistics detail ae0 | no-more
Physical interface: ae0, Enabled, Physical link is Up
  Interface index: 128, SNMP ifIndex: 504, Generation: 278
  Link-level type: Ethernet, MTU: 1514, Speed: 1Gbps, BPDU Error: None, MAC-REWRITE
 Error: None, Loopback: Disabled,
  Source filtering: Disabled, Flow control: Disabled, Minimum links needed: 1,
Minimum bandwidth needed: 0
  Device flags   : Present Running
  Interface flags: SNMP-Traps Internal: 0x4000
  Current address: 00:00:5E:00:53:f0, Hardware address: 00:00:5E:00:53:f0
  Last flapped   : 2009-11-09 03:30:23 PST (00:01:28 ago)
  Statistics last cleared: 2009-11-09 03:26:18 PST (00:05:33 ago)
  Traffic statistics:
   Input  bytes  :            544009602             54761856 bps
   Output bytes  :                 3396                    0 bps
   Input  packets:             11826292               148809 pps
   Output packets:                   42                    0 pps
   IPv6 transit statistics:
    Input  bytes  :           350818604
    Output bytes  :                   0
    Input  packets:             7626488
    Output packets:                   0
  Input errors:
    Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0, Policed discards:
 0, Resource errors: 0
  Output errors:
    Carrier transitions: 0, Errors: 0, Drops: 0, MTU errors: 0, Resource errors:
 0
  Ingress queues: 8 supported, 4 in use
  Queue counters:       Queued packets  Transmitted packets      Dropped packets

    0 best-effort                    0                    0                    0

    1 expedited-fo                   0                    0                    0

    2 assured-forw                   0                    0                    0

    3 network-cont                   0                    0                    0

  Egress queues: 8 supported, 4 in use
  Queue counters:       Queued packets  Transmitted packets      Dropped packets

    0 best-effort                   21                   21                    0

    1 expedited-fo                   0                    0                    0

    2 assured-forw                   0                    0                    0

    3 network-cont                 451                  451                    0


  Logical interface ae0.0 (Index 70) (SNMP ifIndex 574) (Generation 177)
    Flags: SNMP-Traps 0x4000 Encapsulation: ENET2
    Statistics        Packets        pps         Bytes          bps
    Bundle:
        Input :      11826292     148809     544009602     54761856
        Output:            42          0          3396            0
    Link:
```

```
      ge-5/2/0.0
        Input :      11826292     148809     544009602     54761856
        Output:            42          0          3396            0
    Marker Statistics:   Marker Rx     Resp Tx   Unknown Rx   Illegal Rx
      ge-5/2/0.0                0           0            0            0
    Protocol inet, MTU: 1500, Generation: 236, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.30.30.0/30, Local: 10.30.30.2, Broadcast: 10.30.30.3,
Generation: 310
    Protocol inet6, MTU: 1500, Generation: 237, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: ::10.30.30.0/126, Local: ::10.30.30.2
    Generation: 312
      Addresses, Flags: Is-Preferred
        Destination: fe80::/64, Local: fe80::21d:b5ff:fe61:dbf0
    Protocol multiservice, MTU: Unlimited, Generation: 314
    Generation: 238, Route table: 0
      Policer: Input: __default_arp_policer__
```

show interfaces statistics detail (Aggregated Ethernet—Egress)

```
user@host> show interfaces statistics detail ae0 | no-more
Physical interface: ae0, Enabled, Physical link is Up
  Interface index: 128, SNMP ifIndex: 501, Generation: 319
  Link-level type: Ethernet, MTU: 1514, Speed: 1Gbps, BPDU Error: None, MAC-REWRITE
 Error: None, Loopback: Disabled,
  Source filtering: Disabled, Flow control: Disabled, Minimum links needed: 1,
Minimum bandwidth needed: 0
  Device flags   : Present Running
  Interface flags: SNMP-Traps Internal: 0x4000
  Current address: 00:00:5E:00:53:f0, Hardware address: 00:00:5E:00:53:f0
  Last flapped   : 2009-11-09 03:30:24 PST (00:02:42 ago)
  Statistics last cleared: 2009-11-09 03:26:42 PST (00:06:24 ago)
  Traffic statistics:
   Input  bytes  :                  440                    0 bps
   Output bytes  :           1047338120             54635848 bps
   Input  packets:                    7                    0 pps
   Output packets:             22768200               148466 pps
   IPv6 transit statistics:
    Input  bytes  :                 288
    Output bytes  :           723202616
    Input  packets:                   4
    Output packets:            15721796
  Input errors:
    Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0, Policed discards:
 0, Resource errors: 0
  Output errors:
    Carrier transitions: 0, Errors: 0, Drops: 0, MTU errors: 0, Resource errors:
 0
  Ingress queues: 8 supported, 4 in use
  Queue counters:       Queued packets  Transmitted packets      Dropped packets

    0 best-effort                    0                    0                    0

    1 expedited-fo                   0                    0                    0

    2 assured-forw                   0                    0                    0

    3 network-cont                   0                    0                    0

  Egress queues: 8 supported, 4 in use
```

```
  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0 best-effort            201985796            201985796                    0
    1 expedited-fo                   0                    0                    0
    2 assured-forw                   0                    0                    0
    3 network-cont                  65                   65                    0

  Logical interface ae0.0 (Index 72) (SNMP ifIndex 505) (Generation 204)
    Flags: SNMP-Traps 0x4000 Encapsulation: ENET2
    Statistics        Packets        pps         Bytes          bps
    Bundle:
        Input :             7          0           440            0
        Output:      22768200     148466    1047338120     54635848
    Link:
      ge-2/1/6.0
        Input :             7          0           440            0
        Output:      22768200     148466    1047338120     54635848
    Marker Statistics:   Marker Rx     Resp Tx   Unknown Rx   Illegal Rx
      ge-2/1/6.0                 0           0            0            0
    Protocol inet, MTU: 1500, Generation: 291, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.30.30.0/30, Local: 10.30.30.1, Broadcast: 10.30.30.3,
Generation: 420
    Protocol inet6, MTU: 1500, Generation: 292, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: ::/26, Local: ::10.30.30.1
    Generation: 422
      Addresses, Flags: Is-Preferred
        Destination: fe80::/64, Local: fe80::21f:12ff:fec2:37f0
    Protocol multiservice, MTU: Unlimited, Generation: 424
    Generation: 293, Route table: 0
      Policer: Input: __default_arp_policer__
```

show interfaces statistics (SONET/SDH)

```
user@host> show interfaces statistics detail so-3/0/0 | no-more
Physical interface: so-3/0/0, Enabled, Physical link is Up
  Interface index: 133, SNMP ifIndex: 538, Generation: 283
  Link-level type: PPP, MTU: 4474, Clocking: Internal, SONET mode, Speed: OC192,
 Loopback: None, FCS: 16, Payload scrambler: Enabled
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps Internal: 0x4000
  Link flags     : Keepalives
  Hold-times     : Up 0 ms, Down 0 ms
  Keepalive settings: Interval 10 seconds, Up-count 1, Down-count 3
  Keepalive statistics:
    Input : 13 (last seen 00:00:04 ago)
    Output: 14 (last sent 00:00:02 ago)
  LCP state: Opened
  NCP state: inet: Opened, inet6: Opened, iso: Not-configured, mpls: Not-configured

  CHAP state: Closed
  PAP state: Closed
  CoS queues     : 8 supported, 8 maximum usable queues
  Last flapped   : 2009-11-09 02:52:34 PST (01:12:39 ago)
  Statistics last cleared: 2009-11-09 03:58:54 PST (00:06:19 ago)
  Traffic statistics:
```



```
   Input  bytes  :           2559160294            54761720 bps
   Output bytes  :                10640                  48 bps
   Input  packets:             55633975              148809 pps
   Output packets:                  216                   0 pps
   IPv6 transit statistics:
    Input  bytes  :           647922328
    Output bytes  :                   0
    Input  packets:            14085269
    Output packets:                   0
  Input errors:
    Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0, Bucket drops:
0, Policed discards: 0, L3 incompletes: 0,
    L2 channel errors: 0, L2 mismatch timeouts: 0, HS link CRC errors: 0, HS link
 FIFO overflows: 0
  Output errors:
    Carrier transitions: 0, Errors: 0, Drops: 0, Aged packets: 0, HS link FIFO
underflows: 0, MTU errors: 0
  Egress queues: 8 supported, 4 in use
  Queue counters:       Queued packets  Transmitted packets      Dropped packets

    0 best-effort                    4                    4                    0

    1 expedited-fo                   0                    0                    0

    2 assured-forw                   0                    0                    0

    3 network-cont                 213                  213                    0

  SONET alarms   : None
  SONET defects  : None

  Logical interface so-3/0/0.0 (Index 72) (SNMP ifIndex 578) (Generation 182)
    Flags: Point-To-Point SNMP-Traps 0x4000 Encapsulation: PPP
    Protocol inet, MTU: 4470, Generation: 244, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.30.30.0/30, Local: 10.30.30.2, Broadcast: 10.30.30.3,
Generation: 322
    Protocol inet6, MTU: 4470, Generation: 245, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: ::10.30.30.0/126, Local: ::10.30.30.2
    Generation: 324
      Addresses, Flags: Is-Preferred
        Destination: fe80::/64, Local: fe80::2a0:a5ff:fe61:9264
    Generation: 326
```

show interfaces statistics (Aggregated SONET/SDH—Ingress)

```
user@host> show interfaces statistics detail as0 | no-more
Physical interface: as0, Enabled, Physical link is Up
  Interface index: 132, SNMP ifIndex: 534, Generation: 282
  Link-level type: PPP, MTU: 4474, Speed: OC192, Minimum links needed: 1, Minimum
 bandwidth needed: 0
  Device flags   : Present Running
  Interface flags: SNMP-Traps Internal: 0x4000
  Link flags     : Keepalives
  Keepalive settings: Interval 10 seconds, Up-count 1, Down-count 3
  Last flapped   : 2009-11-09 03:45:53 PST (00:09:38 ago)
  Statistics last cleared: 2009-11-09 03:48:17 PST (00:07:14 ago)
  Traffic statistics:
   Input  bytes  :           2969786332            54761688 bps
   Output bytes  :                11601                   0 bps
```

```
     Input  packets:            64560636              148808 pps
     Output packets:                 225                   0 pps
     IPv6 transit statistics:
      Input  bytes  :         2086013152
      Output bytes  :                  0
      Input  packets:           45348114
      Output packets:                  0
    Input errors:
      Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0, Policed discards:
   0, Resource errors: 0
    Output errors:
      Carrier transitions: 0, Errors: 0, Drops: 0, MTU errors: 0, Resource errors:
   0
    Egress queues: 8 supported, 4 in use
    Queue counters:       Queued packets  Transmitted packets      Dropped packets

      0 best-effort                  3                    3                    0

      1 expedited-fo                 0                    0                    0

      2 assured-forw                 0                    0                    0

      3 network-cont               222                  222                    0


    Logical interface as0.0 (Index 71) (SNMP ifIndex 576) (Generation 179)
      Flags: Point-To-Point SNMP-Traps 0x4000 Encapsulation: PPP
      Statistics        Packets        pps         Bytes          bps
      Bundle:
          Input :      64560550     148808    2969785300     54761688
          Output:           139          0         10344            0
      Link:
        so-3/0/0.0
          Input :      64560550     148808    2969785300     54761688
          Output:           139          0         10344            0
      Protocol inet, MTU: 4470, Generation: 240, Route table: 0
        Addresses, Flags: Is-Preferred Is-Primary
          Destination: 10.30.30.0/30, Local: 10.30.30.2, Broadcast: 10.30.30.3,
  Generation: 316
      Protocol inet6, MTU: 4470, Generation: 241, Route table: 0
        Addresses, Flags: Is-Preferred Is-Primary
          Destination: ::10.30.30.0/126, Local: ::10.30.30.2
      Generation: 318
        Addresses, Flags: Is-Preferred
          Destination: fe80::/64, Local: fe80::2a0:a5ff:fe61:9264
      Generation: 320
```

show interfaces statistics (Aggregated SONET/SDH—Egress)

```
user@host> show interfaces statistics detail as0 | no-more
Physical interface: as0, Enabled, Physical link is Up
  Interface index: 132, SNMP ifIndex: 565, Generation: 323
  Link-level type: PPP, MTU: 4474, Speed: OC192, Minimum links needed: 1, Minimum
 bandwidth needed: 0
  Device flags   : Present Running
  Interface flags: SNMP-Traps Internal: 0x4000
  Link flags     : Keepalives
  Keepalive settings: Interval 10 seconds, Up-count 1, Down-count 3
  Last flapped   : 2009-11-09 03:43:37 PST (00:12:48 ago)
  Statistics last cleared: 2009-11-09 03:48:54 PST (00:07:31 ago)
  Traffic statistics:
```



```
   Input  bytes  :              11198                 392 bps
   Output bytes  :         3101452132            54783448 bps
   Input  packets:                234                   0 pps
   Output packets:           67422937              148868 pps
   IPv6 transit statistics:
    Input  bytes  :              5780
    Output bytes  :        2171015678
    Input  packets:                72
    Output packets:          47195993
  Input errors:
    Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Giants: 0, Policed discards:
 0, Resource errors: 0
  Output errors:
    Carrier transitions: 0, Errors: 0, Drops: 0, MTU errors: 0, Resource errors:
 0
  Egress queues: 8 supported, 4 in use
  Queue counters:       Queued packets  Transmitted packets      Dropped packets

    0 best-effort             67422830             67422830                    0

    1 expedited-fo                   0                    0                    0

    2 assured-forw                   0                    0                    0

    3 network-cont                  90                   90                    0


  Logical interface as0.0 (Index 71) (SNMP ifIndex 548) (Generation 206)
    Flags: Point-To-Point SNMP-Traps 0x4000 Encapsulation: PPP
    Statistics        Packets        pps         Bytes          bps
    Bundle:
        Input :           144          0         10118          392
        Output:      67422847     148868    3101450962     54783448
    Link:
      so-0/1/0.0
        Input :           144          0         10118          392
        Output:      67422847     148868    3101450962     54783448
    Protocol inet, MTU: 4470, Generation: 295, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: 10.30.30.0/30, Local: 10.30.30.1, Broadcast: 10.30.30.3,
Generation: 426
    Protocol inet6, MTU: 4470, Generation: 296, Route table: 0
      Addresses, Flags: Is-Preferred Is-Primary
        Destination: ::/26, Local: ::10.30.30.1
    Generation: 428
      Addresses, Flags: Is-Preferred
        Destination: fe80::/64, Local: fe80::2a0:a5ff:fe63:1d0a
    Generation: 429
```

show interfaces statistics (MX Series Routers)

```
user@host> show interfaces xe-0/0/0 statistics
Physical interface: xe-0/0/0, Enabled, Physical link is Up
  Interface index: 145, SNMP ifIndex: 592
  Link-level type: Ethernet, MTU: 1514, LAN-PHY mode, Speed: 10Gbps, BPDU Error:
 None, Loopback: None, Source filtering: Disabled, Flow control: Enabled
  Pad to minimum frame size: Enabled
  Device flags   : Present Running
  Interface flags: SNMP-Traps Internal: 0x0
  Link flags     : None
  CoS queues     : 8 supported, 8 maximum usable queues
```

```
  Current address: 00:00:5E:00:53:f0, Hardware address: 00:00:5E:00:53:f0
  Last flapped   : 2013-10-26 03:20:40 test (2w3d 03:29 ago)
  Statistics last cleared: Never
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)
  Input errors: 0, Output errors: 0
  Active alarms  : LINK
  Active defects : LINK
  PCS statistics                      Seconds
    Bit errors                           109
    Errored blocks                       109
  Interface transmit statistics: Disabled
```

show interfaces statistics (PTX Series Packet Transport Routers)

```
user@host> show interfaces statistics em0
 Physical interface: em0, Enabled, Physical link is Up
      Interface index: 8, SNMP ifIndex: 0
      Type: Ethernet, Link-level type: Ethernet, MTU: 1514, Speed: 1000mbps
      Device flags   : Present Running
      Interface flags: SNMP-Traps
      Link type      : Full-Duplex
      Current address: 00:00:5E:00:53:1b, Hardware address: 00:00:5E:00:53:1b
      Last flapped   : Never
      Statistics last cleared: Never
 Input packets : 212620
 Output packets: 71
      Input errors: 0, Output errors: 0

      Logical interface em0.0 (Index 3) (SNMP ifIndex 0)
 Flags: SNMP-Traps Encapsulation: ENET2
 Input packets : 212590
 Output packets: 71
 Protocol inet, MTU: 1500
   Flags: Is-Primary
   Addresses, Flags: Is-Default Is-Preferred Is-Primary
     Destination: 192.168.3/24, Local: 192.168.3.30,
     Broadcast: 192.168.3.255
```

show interfaces statistics (ACX Series routers)

```
user@host> show interfaces statistics ge-0/1/7
Physical interface: ge-0/1/7, Enabled, Physical link is Down
  Interface index: 151, SNMP ifIndex: 524
  Link-level type: Ethernet, Media type: Copper, MTU: 1514, Link-mode: Full-duplex,
 Speed: 1000mbps, BPDU Error: None, MAC-REWRITE Error: None, Loopback: Disabled,

  Source filtering: Disabled, Flow control: Enabled, Auto-negotiation: Enabled,
Remote fault: Online
  Device flags   : Present Running Down
  Interface flags: Hardware-Down SNMP-Traps Internal: 0x0
  Link flags     : None
  CoS queues     : 8 supported, 8 maximum usable queues
  Current address: 00:00:5E:00:53:a3, Hardware address: 00:00:5E:00:53:a3
  Last flapped   : 2012-05-11 04:25:28 PDT (2d 20:23 ago)
  Statistics last cleared: 2012-05-13 23:07:23 PDT (01:41:25 ago)
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)
  Input errors: 0, Output errors: 0
  Active alarms  : LINK
```



```
Active defects : LINK
Interface transmit statistics: Disabled
```

show policy

List of Syntax

Syntax on page 1364
Syntax (EX Series Switches) on page 1364

Syntax

show policy
<logical-system (all | *logical-system-name*)>
<*policy-name*>
<*statistics* >

Syntax (EX Series Switches)

show policy
<*policy-name*>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
statistics option introduced in Junos OS Release 16.1 for MX Series routers.

Description

Display information about configured routing policies.

Options

none—List the names of all configured routing policies.

logical-system (all | ***logical-system-name*****)**—(Optional) Perform this operation on all logical systems or on a particular logical system.

policy-name—(Optional) Show the contents of the specified policy.

statistics—(Optional) Use in conjunction with the **test policy** command to show the length of time (in microseconds) required to evaluate a given policy and the number of times it has been executed. This information can be used, for example, to help structure a policy so it is evaluated efficiently. Timers shown are per route; times are not cumulative. Statistics are incremented even when the router is learning (and thus evaluating) routes from peering routers.

Required Privilege Level

view

Related Documentation

- show policy damping on page 1369
- test policy on page 1587

List of Sample Output

show policy on page 1365
show policy policy-name on page 1365
show policy statistics policy-name on page 1365
show policy (Multicast Scoping) on page 1366
show policy (Route Filter and source Address Filter Lists) on page 1366

Output Fields

Table 77 on page 1365 lists the output fields for the **show policy** command. Output fields are listed in the approximate order in which they appear.

Table 77: show policy Output Fields

Field Name	Field Description
policy-name	Name of the policy listed.
term	Name of the user-defined policy term. The term name **unnamed** is used for policy elements that occur outside of user defined terms
from	Match condition for the policy.
then	Action for the policy.

Sample Output

show policy

```
user@host> show policy
Configured policies:
__vrf-export-red-internal__
__vrf-import-red-internal__
red-export
rf-test-policy
multicast-scoping
```

show policy policy-name

```
user@host> show policy vrf-import-red-internal
Policy vrf-import-red-internal:
    from
         203.0.113.0/28  accept
         203.0.113.32/28  accept
    then reject
```

show policy statistics policy-name

```
user@host> show policy statistics iBGP-v4-RR-Import
Policy iBGP-v4-RR-Import:
    [1243328] Term Lab-Infra:
        from [1243328 0]  proto BGP
             [28 0] route filter:
                 10.11.0.0/8 orlonger
                 10.13.0.0/8 orlonger
        then [28 0] accept
    [1243300] Term External:
        from [1243300 1]  proto BGP
         [1243296 0]  community Ext-Com1 [64496:1515 ]
         [1243296 0]  prefix-list-filter Customer-Routes
         [1243296 0]  aspath AS6221
             [1243296 1] route filter:
                 172.16.49.0/12 orlonger
                 172.16.50.0/12 orlonger
                 172.16.51.0/12 orlonger
```



```
                172.16.52.0/12 orlonger
                172.16.56.0/12 orlonger
                172.16.60.0/12 orlonger
        then [1243296 2] community + Ext-Com2 [64496:2000 ] [1243296 0] accept
    [4] Term Final:
        then [4 0] reject
```

show policy (Multicast Scoping)

```
user@host> show policy multicast-scoping
Policy multicast-scoping:
    from
         multicast-scope == 8
  then
    accept
```

show policy (Route Filter and source Address Filter Lists)

```
user@host> show policy rf-test-policy
Policy rf-test-policy:
    Term term1:
        from source-address-filter-list saf-list-1
             source-address filter:
                 192.0.2.0/29 longer
                 192.0.2.64/28 exact
                 192.0.2.128/28 exact
                 192.0.2.160/28 orlonger
    Term term2:
        from route-filter-list rf-list-1
             route filter:
                 198.51.100.0/29 upto 198.51.100.0/30
                 198.51.100.8/29 upto 198.51.100.8/30 accept
    Term unnamed:
        then reject
```


show policy conditions

Syntax

show policy conditions
<*condition-name*>
<detail>
<dynamic>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show policy conditions
<*condition-name*>
<detail>
<dynamic>

Release Information

Command introduced in Junos OS Release 9.0.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display all the configured conditions as well as the routing tables with which the configuration manager is interacting. If the **detail** keyword is included, the output also displays dependent routes for each condition.

Options

none—Display all configured conditions and associated routing tables.

condition-name—(Optional) Display information about the specified condition only.

detail—(Optional) Display the specified level of output.

dynamic—(Optional) Display information about the conditions in the dynamic database.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show policy conditions detail on page 1368

Output Fields

Table 78 on page 1367 lists the output fields for the **show policy conditions** command. Output fields are listed in the approximate order in which they appear.

Table 78: show policy conditions Output Fields

Field Name	Field Description	Level of Output
Condition	Name of configured condition.	All levels
event	Condition type. If the **if-route-exists** option is configured, the event type is: **Existence of a route in a specific routing table**.	All levels
Dependent routes	List of routes dependent on the condition, along with the latest generation number.	**detail**

Table 78: show policy conditions Output Fields *(continued)*

Field Name	Field Description	Level of Output
Condition tables	List of routing tables associated with the condition, along with the latest generation number and number of dependencies.	All levels
If-route-exists conditions	List of conditions configured to look for a route in the specified table.	All levels

Sample Output

show policy conditions detail

```
user@host> show policy conditions detail
Configured conditions:
Condition cond1, event: Existence of a route in a specific routing table
Dependent routes:
 172.16.4.4/32, generation 3
 6.6.6.6/32, generation 3
 10.10.10.10/32, generation 3

Condition cond2, event: Existence of a route in a specific routing table
Dependent routes:
None

Condition tables:
Table inet.0, generation 4, dependencies 3, If-route-exists conditions: cond1
(static) cond2 (static)
```

show policy damping

List of Syntax

Syntax on page 1369
Syntax (EX Series Switch and QFX Series) on page 1369

Syntax

show policy damping
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switch and QFX Series)

show policy damping

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Command introduced in Junos OS Release 11.3 for the QFX Series.
Command introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description

Display information about BGP route flap damping parameters.

Options

none—Display information about BGP route flap damping parameters.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Additional Information

In the output from this command, figure-of-merit values correlate with the probability of future instability of a routing device. Routes with higher figure-of-merit values are suppressed for longer periods of time. The figure-of-merit value decays exponentially over time. A figure-of-merit value of zero is assigned to each new route. The value is increased each time the route is withdrawn or readvertised, or when one of its path attributes changes.

Required Privilege Level

view

Related Documentation

- “Configuring BGP Flap Damping Parameters” in the *Routing Policies, Firewall Filters, and Traffic Policers Feature Guide*
- *clear bgp damping*
- **show route damping on page 1415**

List of Sample Output

show policy damping on page 1370

Output Fields

Table 79 on page 1370 describes the output fields for the **show policy damping** command. Output fields are listed in the approximate order in which they appear.

Table 79: show policy damping Output Fields

Field Name	Field Description
Halflife	Decay half-life, in minutes. The value represents the period during which the accumulated figure-of-merit value is reduced by half if the route remains stable. If a route has flapped, but then becomes stable, the figure-of-merit value for the route decays exponentially. For example, for a route with a figure-of-merit value of 1500, if no incidents occur, its figure-of-merit value is reduced to 750 after 15 minutes and to 375 after another 15 minutes.
Reuse merit	Figure-of-merit value below which a suppressed route can be used again. A suppressed route becomes reusable when its figure-of-merit value decays to a value below a reuse threshold, and the route once again is considered usable and can be installed in the forwarding table and exported from the routing table.
Suppress/cutoff merit	Figure-of-merit value above which a route is suppressed for use or inclusion in advertisements. When a route's figure-of-merit value reaches a particular level, called the cutoff or suppression threshold, the route is suppressed. When a route is suppressed, the routing table no longer installs the route into the forwarding table and no longer exports this route to any of the routing protocols.
Maximum suppress time	Maximum hold-down time, in minutes. The value represents the maximum time that a route can be suppressed no matter how unstable it has been before this period of stability.
Computed values	• **Merit ceiling**—Maximum merit that a flapping route can collect. • **Maximum decay**—Maximum decay half-life, in minutes.

Sample Output

show policy damping

```
user@host> show policy damping
Default damping information:
  Halflife: 15 minutes
  Reuse merit: 750 Suppress/cutoff merit: 3000
  Maximum suppress time: 60 minutes
  Computed values:
    Merit ceiling: 12110
    Maximum decay: 6193
Damping information for "standard-damping":
  Halflife: 10 minutes
  Reuse merit: 4000 Suppress/cutoff merit: 8000
  Maximum suppress time: 30 minutes
  Computed values:
    Merit ceiling: 32120
    Maximum decay: 12453
```


show route

List of Syntax

Syntax on page 1371
Syntax (EX Series Switches) on page 1371

Syntax

show route
<all>
<*destination-prefix*>
<logical-system (all | *logical-system-name*)>
<private>
<te-ipv4-prefix-ip *te-ipv4-prefix-ip*>
<te-ipv4-prefix-node-ip *te-ipv4-prefix-node-ip*>
<te-ipv4-prefix-node-iso *te-ipv4-prefix-node-iso*>

Syntax (EX Series Switches)

show route
<all>
<*destination-prefix*>
<private>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Option **private** introduced in Junos OS Release 9.5.
Option **private** introduced in Junos OS Release 9.5 for EX Series switches.
Command introduced in Junos OS Release 15.1R3 on MX Series routers for enhanced subscriber management.
Option **display-client-data** introduced in Junos OS Release 16.2R1 on MX80, MX104, MX240, MX480, MX960, MX2010, MX2020, vMX Series routers.
Options **te-ipv4-prefix-ip**, **te-ipv4-prefix-node-ip**, and **te-ipv4-prefix-node-iso** introduced in Junos OS Release 17.2R1 on MX Series and PTX Series.

Description

Display the active entries in the routing tables.

Options

none—Display brief information about all active entries in the routing tables.

all—(Optional) Display information about all routing tables, including private, or internal, routing tables.

destination-prefix—(Optional) Display active entries for the specified address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

private—(Optional) Display information only about all private, or internal, routing tables.

display-client-data —(Optional) Display client id and cookie information for routes installed by the routing protocol process client applications.

te-ipv4-prefix-ip *te-ipv4-prefix-ip*—(Optional) Display IPv4 address of the traffic-engineering prefix, without the mask length if present in the routing table.

te-ipv4-prefix-node-ip ***te-ipv4-prefix-node-ip***—(Optional) Display all prefixes that have originated from the traffic-engineering node. You can filter IPv4 node addresses from the traffic-engineered routes in the **lsdist.0** table.

te-ipv4-prefix-node-iso ***te-ipv4-prefix-node-iso***—(Optional) Display all prefixes that have originated from the traffic-engineering node. You can filter IPv4 routes with the specified ISO circuit ID from the **lsdist.0** table.

Required Privilege Level

view

Related Documentation

- *Understanding IS-IS Configuration*
- *Example: Configuring IS-IS*
- *Examples: Configuring Internal BGP Peering*
- *Examples: Configuring External BGP Peering*
- *Examples: Configuring OSPF Routing Policy*
- *Verifying and Managing Junos OS Enhanced Subscriber Management*

List of Sample Output

Output Fields

Table 80 on page 1372 describes the output fields for the **show route** command. Output fields are listed in the approximate order in which they appear.

Table 80: show route Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).
number **destinations**	Number of destinations for which there are routes in the routing table.

Table 80: show route Output Fields *(continued)*

<table>
<tr><th>Field Name</th><th>Field Description</th></tr>
<tr><td>number routes</td><td>Number of routes in the routing table and total number of routes in the following states:

active (routes that are active).
holddown (routes that are in the pending state before being declared inactive). A holddown route was once the active route and is no longer the active route. The route is in the holddown state because a protocol still has interest in the route, meaning that the interest bit is set. A protocol might have its interest bit set on the previously active route because the protocol is still advertising the route. The route will be deleted after all protocols withdraw their advertisement of the route and remove their interest bit. A persistent holddown state often means that the interested protocol is not releasing its interest bit properly.
However, if you have configured advertisement of multiple routes (with the add-path or advertise-inactive statement), the holddown bit is most likely set because BGP is advertising the route as an active route. In this case, you can ignore the holddown state because nothing is wrong.
hidden (routes that are not used because of a routing policy).
</td></tr>
<tr><td>destination-prefix</td><td>Route destination (for example:10.0.0.1/24). Sometimes the route information is presented in another format, such as:

MPLS-label (for example, 80001).
interface-name (for example, ge-1/0/2).
neighbor-address:control-word-status:encapsulation type:vc-id:source (Layer 2 circuit only. For example, 10.1.1.195:NoCtrlWord:1:1:Local/96):

neighbor-address—Address of the neighbor.
control-word-status—Whether the use of the control word has been negotiated for this virtual circuit: NoCtrlWord or CtrlWord.
encapsulation type—Type of encapsulation, represented by a number: (1) Frame Relay DLCI, (2) ATM AAL5 VCC transport, (3) ATM transparent cell transport, (4) Ethernet, (5) VLAN Ethernet, (6) HDLC, (7) PPP, (8) ATM VCC cell transport, (10) ATM VPC cell transport.
vc-id—Virtual circuit identifier.
source—Source of the advertisement: Local or Remote.

</td></tr>
<tr><td>[protocol, preference]</td><td>Protocol from which the route was learned and the preference value for the route.

+—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table.
- —A hyphen indicates the last active route.
*—An asterisk indicates that the route is both the active and the last active route. An asterisk before a to line indicates the best subpath to the route.

In every routing metric except for the BGP LocalPref attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the LocalPref value in the Preference2 field. For example, if the LocalPref value for Route 1 is 100, the Preference2 value is -101. If the LocalPref value for Route 2 is 155, the Preference2 value is -156. Route 2 is preferred because it has a higher LocalPref value and a lower Preference2 value.</td></tr>
<tr><td>weeks:days hours:minutes:seconds</td><td>How long the route been known (for example, 2w4d 13:11:14, or 2 weeks, 4 days, 13 hours, 11 minutes, and 14 seconds).</td></tr>
<tr><td>metric</td><td>Cost value of the indicated route. For routes within an AS, the cost is determined by the IGP and the individual protocol metrics. For external routes, destinations, or routing domains, the cost is determined by a preference value.</td></tr>
</table>

Table 80: show route Output Fields *(continued)*

Field Name	Field Description
localpref	Local preference value included in the route.
from	Interface from which the route was received.
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the local AS number associated with the AS path if more than one AS number is configured on the routing device, or if AS path prepending is configured. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.
encapsulated	Extended next-hop encoding capability enabled for the specified BGP community for routing IPv4 traffic over IPv6 tunnels. When BGP receives routes without the tunnel community, IPv4-0over IPv6 tunnels are not created and BGP routes are resolved without encapsulation.
Route Labels	Stack of labels carried in the BGP route update.
validation-state	(BGP-learned routes) Validation status of the route: • **Invalid**—Indicates that the prefix is found, but either the corresponding AS received from the EBGP peer is not the AS that appears in the database, or the prefix length in the BGP update message is longer than the maximum length permitted in the database. • **Unknown**—Indicates that the prefix is not among the prefixes or prefix ranges in the database. • **Unverified**—Indicates that the origin of the prefix is not verified against the database. This is because the database got populated and the validation is not called for in the BGP import policy, although origin validation is enabled, or the origin validation is not enabled for the BGP peers. • **Valid**—Indicates that the prefix and autonomous system pair are found in the database.
to	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route. If the destination is **Discard**, traffic is dropped.

Table 80: show route Output Fields *(continued)*

Field Name	Field Description
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the interface that is actually used is followed by the word **Selected**. This field can also contain the following information: • **Weight**—Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible. • **Balance**—Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a routing device is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing. • ***lsp-path-name***—Name of the LSP used to reach the next hop. • ***label-action***—MPLS label and operation occurring at the next hop. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label). For VPNs, expect to see multiple **push** operations, corresponding to the inner and outer labels required for VPN routes (in the case of a direct PE-to-PE connection, the VPN route would have the inner label push only).
Private unicast	(Enhanced subscriber management for MX Series routers) Indicates that an access-internal route is managed by enhanced subscriber management. By contrast, access-internal routes *not* managed by enhanced subscriber management are displayed with associated next-hop and media access control (MAC) address information.
balance	Distribution of the load based on the underlying operational interface bandwidth for equal-cost multipaths (ECMP) across the nexthop gateways in percentages.

Sample Output

show route

```
user@host> show route
inet.0: 11 destinations, 12 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:65500:1:10.0.0.20/240
                   *[MVPN/70] 19:53:41, metric2 1
                      Indirect
1:65500:1:10.0.0.40/240
                   *[BGP/170] 19:53:29, localpref 100, from 10.0.0.30
                      AS path: I
                    > to 10.0.24.4 via lt-0/3/0.24, label-switched-path toD
                    [BGP/170] 19:53:26, localpref 100, from 10.0.0.33
                      AS path: I
                    > to 10.0.24.4 via lt-0/3/0.24, label-switched-path toD
1:65500:1:10.0.0.60/240
                   *[BGP/170] 19:53:29, localpref 100, from 10.0.0.30
                      AS path: I
                    > to 10.0.28.8 via lt-0/3/0.28, label-switched-path toF
                    [BGP/170] 19:53:25, localpref 100, from 10.0.0.33
                      AS path: I
                    > to 10.0.28.8 via lt-0/3/0.28, label-switched-path toF
```


show route (VPN)

The following sample output shows a VPN route with composite next hops enabled. The first **Push** operation corresponds to the outer label. The second **Push** operation corresponds to the inner label.

```
user@host> show route 192.0.2.0

13979:665001.inet.0: 871 destinations, 3556 routes (871 active, 0 holddown, 0 
hidden)
+ = Active Route, - = Last Active, * = Both

192.0.2.0/24        [BGP/170] 00:28:32, localpref 100, from 10.9.9.160
                      AS path: 13980 ?, validation-state: unverified
                    > to 10.100.0.42 via ae2.0, Push 16, Push 300368(top)
                    [BGP/170] 00:28:28, localpref 100, from 10.9.9.169
                      AS path: 13980 ?, validation-state: unverified
                    > to 10.100.0.42 via ae2.0, Push 126016, Push 300368(top)
                   #[Multipath/255] 00:28:28, metric2 102
                    > to 10.100.0.42 via ae2.0, Push 16, Push 300368(top)
                      to 10.100.0.42 via ae2.0, Push 16, Push 300368(top)
```

show route (with Destination Prefix)

```
user@host> show route 172.16.0.0/12

inet.0: 10 destinations, 10 routes (9 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.0.0/12      *[Static/5] 2w4d 12:54:27
                    > to 192.168.167.254 via fxp0.0
```

show route destination-prefix detail

```
user@host> show route 198.51.100.0 detail

inet.0: 15 destinations, 20 routes (15 active, 0 holddown, 0 hidden)
198.51.100.0/24 (2 entries, 2 announced)
        *BGP    Preference: 170/-101
         ...
         BGP-Static Preference: 4294967292
               Next hop type: Discard
                 Address: 0x9041ae4
                 Next-hop reference count: 2
                 State: <NoReadvrt Int Ext AlwaysFlash>
        Inactive reason: Route Preference
        Local AS:   200
        Age: 4d 1:40:40
        Validation State: unverified
        Task: RT
        Announcement bits (1): 2-BGP_RT_Background
        AS path: 4 5 6 I
```

show route extensive

```
user@host> show route extensive
v1.mvpn.0: 5 destinations, 8 routes (5 active, 1 holddown, 0 hidden)
1:65500:1:10.0.0.40/240 (1 entry, 1 announced)
    *BGP   Preference: 170/-101
```



```
           PMSI: Flags 0x0: Label[0:0:0]: PIM-SM: Sender 10.0.0.40 Group
203.0.113.1
           Next hop type: Indirect
           Address: 0x92455b8
           Next-hop reference count: 2
           Source: 10.0.0.30
           Protocol next hop: 10.0.0.40
           Indirect next hop: 2 no-forward
           State: <Active Int Ext>
                     Local AS: 64510 Peer AS: 64511
           Age: 3  Metric2: 1
          Validation State: unverified
           Task: BGP_64510.10.0.0.30+179
           Announcement bits (2): 0-PIM.v1 1-mvpn global task
           AS path: I (Originator) Cluster list:  10.0.0.30
           AS path:  Originator ID: 10.0.0.40
           Communities: target:64502:100 encapsulation:0L:14           Import
Accepted
           Localpref: 100
           Router ID: 10.0.0.30
           Primary Routing Table bgp.mvpn.0
           Indirect next hops: 1
                Protocol next hop: 10.0.0.40 Metric: 1
                Indirect next hop: 2 no-forward
                Indirect path forwarding next hops: 1
                      Next hop type: Router
                      Next hop: 10.0.24.4 via lt-0/3/0.24 weight 0x1
                10.0.0.40/32 Originating RIB: inet.3
                      Metric: 1                        Node path count: 1
                      Forwarding nexthops: 1
                         Nexthop: 10.0.24.4 via lt-0/3/0.24
```

show route extensive (ECMP)

```
user@host> show route extensive
*IS-IS Preference: 15
                Level: 1
                Next hop type: Router, Next hop index: 1048577
                Address: 0xXXXXXXXXXX
                Next-hop reference count: YY
                Next hop: 172.16.50.2 via ae1.0 balance 43%, selected
                Session Id: 0x141
                Next hop: 192.0.2.2 via ae0.0 balance 57%
```

show route extensive (Multipath Resolution)

```
user@host> show route extensive
inet.0: 37 destinations, 37 routes (36 active, 0 holddown, 1 hidden)
10.1.1.2/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.1.1.2/32 -> {indirect(1048574)}
        *Static Preference: 5
                Next hop type: Indirect, Next hop index: 0
                Address: 0xb39d1b0
                Next-hop reference count: 2
                Next hop type: Router, Next hop index: 581
                Next hop: 12.1.1.2 via ge-2/0/1.0, selected
                Session Id: 0x144
                Next hop: 13.1.1.2 via ge-2/0/2.0, selected
                Session Id: 0x145
```

```
                Protocol next hop: 10.1.1.1
                Indirect next hop: 0xb2b20f0 1048574 INH Session ID: 0x143
                State: <Active Int Ext>
                Age: 2:53  Metric2: 0
                Validation State: unverified
                Task: RT
                Announcement bits (2): 0-KRT 2-Resolve tree 1
                AS path: I
                Indirect next hops: 1
                        Protocol next hop: 10.1.1.1
                      Indirect next hop: 0xb2b20f0 1048574 INH Session ID: 0x143

                        Indirect path forwarding next hops: 2
                                Next hop type: Router
                                Next hop: 12.1.1.2 via ge-2/0/1.0
                                Session Id: 0x144
                                Next hop: 13.1.1.2 via ge-2/0/2.0
                                Session Id: 0x145
                     10.1.1.1/32 Originating RIB: inet.0
                        Node path count: 1
                          Node flags: 1
                        Forwarding nexthops: 2 (Merged)
                        Nexthop: 12.1.1.2 via ge-2/0/1.0
                         Nexthop: 13.1.1.2 via ge-2/0/2.0

user@host> show route active-path extensive
user@host> show route 141.1.1.1 active-path extensive

inet.0: 1000061 destinations, 1000082 routes (1000061 active, 0 holddown, 0 hidden)
141.1.1.1/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 141.1.1.1/32 -> {indirect(1051215)}
unicast reverse-path: 0
[ae0.0 ae1.0]
Page 0 idx 0, (group Internet-IPv4 type External) Type 1 val 0xbb2e53d8 (adv_entry)
Advertised metrics:
Nexthop: Self
AS path: [500] 410 I
Communities:
Path 141.1.1.1 from 50.0.0.11 Vector len 4. Val: 0
*BGP Preference: 170/-101
Next hop type: Indirect, Next hop index: 0
Address: 0x2e9aacdc
Next-hop reference count: 500000
Source: 50.0.0.11
Next hop type: Router, Next hop index: 0
Next hop: 50.0.12.2 via ae0.0 weight 0x1
Label operation: Push 3851, Push 25, Push 20(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
Load balance label: Label 3851: None; Label 25: None; Label 20: None;
Label element ptr: 0xb5dc1780
Label parent element ptr: 0x18d48080
Label element references: 2
Label element child references: 0
Label element lsp id: 0
Session Id: 0x0
Next hop: 50.0.12.2 via ae0.0 weight 0x1
Label operation: Push 3851, Push 25, Push 22(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
Load balance label: Label 3851: None; Label 25: None; Label 22: None;
Label element ptr: 0xb5dc1700
Label parent element ptr: 0x18d41000
```



```
Label element references: 2
Label element child references: 0
Label element lsp id: 0
Session Id: 0x0
Next hop: 50.0.12.2 via ae0.0 weight 0x1
Label operation: Push 3851, Push 24, Push 48(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
Load balance label: Label 3851: None; Label 24: None; Label 48: None;
Label element ptr: 0x18d40800
Label parent element ptr: 0x18d49780
Label element references: 3
Label element child references: 0
Label element lsp id: 0
Session Id: 0x0
Next hop: 50.0.12.2 via ae0.0 weight 0x1
Label operation: Push 3851, Push 24, Push 49(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
Load balance label: Label 3851: None; Label 24: None; Label 49: None;
Label element ptr: 0xb5dc1680
Label parent element ptr: 0x18d48f00
Label element references: 2
Label element child references: 0
Label element lsp id: 0
Session Id: 0x0
Next hop: 50.0.13.3 via ae1.0 weight 0x1
Label operation: Push 3851, Push 25, Push 21(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
Load balance label: Label 3851: None; Label 25: None; Label 21: None;
Label element ptr: 0xb5dc1600
Label parent element ptr: 0x18d44d80
Label element references: 2
Label element child references: 0
Label element lsp id: 0
Session Id: 0x0
Next hop: 50.0.13.3 via ae1.0 weight 0x1
Label operation: Push 3851, Push 25, Push 25(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
Load balance label: Label 3851: None; Label 25: None; Label 25: None;
Label element ptr: 0xb5dc1580
Label parent element ptr: 0x18d3da80
Label element references: 2
Label element child references: 0
Label element lsp id: 0
Session Id: 0x0
Next hop: 50.0.13.3 via ae1.0 weight 0x1, selected
Label operation: Push 3851, Push 24, Push 68(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
Load balance label: Label 3851: None; Label 24: None; Label 68: None;
Label element ptr: 0x18d41500
Label parent element ptr: 0x18d49000
Label element references: 3
Label element child references: 0
Label element lsp id: 0
Session Id: 0x0
Next hop: 50.0.13.3 via ae1.0 weight 0x1
Label operation: Push 3851, Push 24, Push 69(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl(top)
Load balance label: Label 3851: None; Label 24: None; Label 69: None;
Label element ptr: 0xb5dc1500
Label parent element ptr: 0x18d48300
Label element references: 2
```

```
Label element child references: 0
Label element lsp id: 0
Session Id: 0x0
Protocol next hop: 50.0.0.11
Label operation: Push 3851
Label TTL action: prop-ttl
Load balance label: Label 3851: None;
Indirect next hop: 0x1883e200 1051215 INH Session ID: 0xb0d
State:
Local AS: 500 Peer AS: 500
Age: 1:40:03 Metric2: 2
Validation State: unverified
Task: BGP_500.50.0.0.11
Announcement bits (5): 0-KRT 8-KRT 9-BGP_RT_Background 10-Resolve tree 5 11-Resolve
 tree 8
AS path: 410 I
Accepted
Route Label: 3851
Localpref: 100
Router ID: 50.0.0.11
Indirect next hops: 1
Protocol next hop: 50.0.0.11 Metric: 2
Label operation: Push 3851
Label TTL action: prop-ttl
Load balance label: Label 3851: None;
Indirect next hop: 0x1883e200 1051215 INH Session ID: 0xb0d
Indirect path forwarding next hops (Merged): 8
Next hop type: Router
Next hop: 50.0.12.2 via ae0.0 weight 0x1
Session Id: 0x0
Next hop: 50.0.12.2 via ae0.0 weight 0x1
Session Id: 0x0
Next hop: 50.0.12.2 via ae0.0 weight 0x1
Session Id: 0x0
Next hop: 50.0.12.2 via ae0.0 weight 0x1
Session Id: 0x0
Next hop: 50.0.13.3 via ae1.0 weight 0x1
Session Id: 0x0
Next hop: 50.0.13.3 via ae1.0 weight 0x1
Session Id: 0x0
Next hop: 50.0.13.3 via ae1.0 weight 0x1
Session Id: 0x0
Next hop: 50.0.13.3 via ae1.0 weight 0x1
Session Id: 0x0
50.0.0.11/32 Originating RIB: inet.3
Metric: 1 Node path count: 4
Node flags: 1
Indirect nexthops: 4
Protocol Nexthop: 50.0.0.4 Metric: 1 Push 24
Indirect nexthop: 0x1880f200 1048597 INH Session ID: 0xb0c
Path forwarding nexthops link: 0x36120400
Path inh link: 0x0
Indirect path forwarding nexthops: 2
Nexthop: 50.0.12.2 via ae0.0
Session Id: 0
Nexthop: 50.0.13.3 via ae1.0
Session Id: 0
50.0.0.4/32 Originating RIB: inet.3
Metric: 1 Node path count: 1
Forwarding nexthops: 2
Nexthop: 50.0.12.2 via ae0.0
```



```
Session Id: 0
Nexthop: 50.0.13.3 via ae1.0
Session Id: 0
Protocol Nexthop: 50.0.0.5 Metric: 1 Push 24
Indirect nexthop: 0x18810000 1048596 INH Session ID: 0xb0b
Path forwarding nexthops link: 0x1545be00
Path inh link: 0x0
Indirect path forwarding nexthops: 2
Nexthop: 50.0.12.2 via ae0.0
Session Id: 0
Nexthop: 50.0.13.3 via ae1.0
Session Id: 0
50.0.0.5/32 Originating RIB: inet.3
Metric: 1 Node path count: 1
Forwarding nexthops: 2
Nexthop: 50.0.12.2 via ae0.0
Session Id: 0
Nexthop: 50.0.13.3 via ae1.0
Session Id: 0
Protocol Nexthop: 50.0.0.6 Metric: 1 Push 25
Indirect nexthop: 0x1880e600 1048588 INH Session ID: 0xb0a
Path forwarding nexthops link: 0x3611f440
Path inh link: 0x0
Indirect path forwarding nexthops: 2
Nexthop: 50.0.12.2 via ae0.0
Session Id: 0
Nexthop: 50.0.13.3 via ae1.0
Session Id: 0
50.0.0.6/32 Originating RIB: inet.3
Metric: 1 Node path count: 1
Forwarding nexthops: 2
Nexthop: 50.0.12.2 via ae0.0
Session Id: 0
Nexthop: 50.0.13.3 via ae1.0
Session Id: 0
Protocol Nexthop: 50.0.0.7 Metric: 1 Push 25
Indirect nexthop: 0x1880dc00 1048586 INH Session ID: 0xb09
Path forwarding nexthops link: 0x15466d80
Path inh link: 0x0
Indirect path forwarding nexthops: 2
Nexthop: 50.0.12.2 via ae0.0
Session Id: 0
Nexthop: 50.0.13.3 via ae1.0
Session Id: 0
50.0.0.7/32 Originating RIB: inet.3
Metric: 1 Node path count: 1
Forwarding nexthops: 2
Nexthop: 50.0.12.2 via ae0.0
Session Id: 0
Nexthop: 50.0.13.3 via ae1.0
Session Id: 0
```

show route (Enhanced Subscriber Management)

```
user@host> show route
inet.0: 41 destinations, 41 routes (40 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

198.51.100.11/24    *[Access-internal/12] 00:00:08
                > to #0 10.0.0.1.93.65 via demux0.1073741824
```

```
198.51.100.12/24    *[Access-internal/12] 00:00:08
                Private unicast
```

show route (IPv6 Flow Specification)

```
user@host> show route
inet6.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::10:255:185:19/128
                   *[Direct/0] 05:11:27
                    > via lo0.0
2001:db8::11:11:11:0/120
                   *[BGP/170] 00:28:58, localpref 100
                      AS path: 2000 I, validation-state: unverified
                    > to 2001:db8::13:14:2:2 via ge-1/1/4.0
2001:db8::13:14:2:0/120*[Direct/0] 00:45:07
                    > via ge-1/1/4.0
2001:db8::13:14:2:1/128*[Local/0] 00:45:18
                      Local via ge-1/1/4.0
fe80::2a0:a50f:fc71:71d5/128
                   *[Direct/0] 05:11:27
                    > via lo0.0
fe80::5e5e:abff:feb0:933e/128
                   *[Local/0] 00:45:18
                      Local via ge-1/1/4.0

inet6flow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::11:11:11:10/128,*,proto=6,dstport=80,srcport=65535/term:1
                   *[BGP/170] 00:28:58, localpref 100, from 2001:db8::13:14:2:2
                      AS path: 2000 I, validation-state: unverified
                      Fictitious
2001:db8::11:11:11:30/128,*,icmp6-type=128,len=100,dscp=10/term:2
                   *[BGP/170] 00:20:54, localpref 100, from 2001:db8::13:14:2:2
                      AS path: 2000 I, validation-state: unverified
                      Fictitious
```

show route display-client-data detail

```
user@host> show route 198.51.100.0/24 display-client-data detail
inet.0: 59 destinations, 70 routes (59 active, 0 holddown, 0 hidden)
198.51.100.0/24 (1 entry, 1 announced)
        State: <FlashAll>
        *BGP-Static Preference: 5/-101
                Next hop type: Indirect, Next hop index: 0
                Address: 0xa5c2af8
                Next-hop reference count: 2
                Next hop type: Router, Next hop index: 1641
                Next hop: 192.0.2.1 via ge-2/1/1.0, selected
                Session Id: 0x160
                Protocol next hop: 192.0.2.1
                Indirect next hop: 0xa732cb0 1048621 INH Session ID: 0x17e
                State: <Active Int Ext AlwaysFlash NSR-incapable Programmed>
                Age: 3:13       Metric2: 0
                Validation State: unverified
                Announcement bits (3): 0-KRT 5-LDP 6-Resolve tree 3
```



```
                AS path: I
                Client id: 1, Cookie: 1
```

show route te-ipv4-prefix-ip

```
user@host> show route te-ipv4-prefix-ip 10.10.10.10
lsdist.0: 283 destinations, 283 routes (283 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both


PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:10.10.10.10/32 } ISIS-L1:0
 }/1152
                   *[IS-IS/15] 00:01:01
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0101.0101.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152
                   *[IS-IS/18] 00:01:01
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0202.0202.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152
                   *[IS-IS/18] 00:01:01
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0303.0303.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152
                   *[IS-IS/18] 00:01:01
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0404.0404.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152
                   *[IS-IS/18] 00:01:01
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0505.0505.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152
                   *[IS-IS/18] 00:01:01
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0606.0606.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152
                   *[IS-IS/18] 00:01:01
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0707.0707.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152
                   *[IS-IS/18] 00:01:01
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152
                   *[IS-IS/18] 00:01:01
                      Fictitious
```

show route te-ipv4-prefix-ip extensive

```
user@host>show route te-ipv4-prefix-ip 10.10.10.10 extensive
lsdist.0: 298 destinations, 298 routes (298 active, 0 holddown, 0 hidden)
        *IS-IS  Preference: 15
                Level: 1
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 298
                Next hop:
                State:<Active NotInstall>
                Local AS:   100
                Age: 7:58
```

```
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0x40, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0101.0101.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 298
                Next hop:
       State: <Active NotInstall
       Local AS:   100
                Age: 7:58
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0xe0, Algo: 0>

PREFIX { Node { AS:100 ISO:0100.0202.0202.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 298
                Next hop:
                State: <Active NotInstall>
       Local AS:   100
                Age: 7:58
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0303.0303.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 298
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 7:58
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0404.0404.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 298
                Next hop:
```



```
                State: <Active NotInstall>
                Local AS:   100
                Age: 7:58
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0505.0505.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 298
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 7:58
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0606.0606.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 298
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 7:58
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0707.0707.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 298
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 7:58
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
```

```
                Address: 0xa1a2ac4
                Next-hop reference count: 298
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 7:58
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0x40, Algo: 0
```

show route te-ipv4-prefix-node-iso

```
user@host> show route te-ipv4-prefix-node-iso 0100.0a0a.0a0a.00
lsdist.0: 283 destinations, 283 routes (283 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:10.10.10.10/32 } ISIS-L1:0
 }/1152
                   *[IS-IS/15] 00:05:20
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:1.1.1.1/32 } ISIS-L2:0
}/1152
                   *[IS-IS/18] 00:05:20
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:2.2.2.2/32 } ISIS-L2:0
}/1152
                   *[IS-IS/18] 00:05:20
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:3.3.3.3/32 } ISIS-L2:0
}/1152
                   *[IS-IS/18] 00:05:20
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:4.4.4.4/32 } ISIS-L2:0
}/1152
                   *[IS-IS/18] 00:05:20
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:5.5.5.5/32 } ISIS-L2:0
}/1152
                   *[IS-IS/18] 00:05:20
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:6.6.6.6/32 } ISIS-L2:0
}/1152
                   *[IS-IS/18] 00:05:20
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:7.7.7.7/32 } ISIS-L2:0
}/1152
                   *[IS-IS/18] 00:05:20
                      Fictitious
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152
                   *[IS-IS/18] 00:05:20
                      Fictitious
```

show route te-ipv4-prefix-node-iso extensive

```
user@host> show route te-ipv4-prefix-node-iso 0100.0a0a.0a0a.00 extensive
lsdist.0: 283 destinations, 283 routes (283 active, 0 holddown, 0 hidden)
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:10.10.10.10/32 } ISIS-L1:0
```



```
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 15
                Level: 1
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:47
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0x40, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:1.1.1.1/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:47
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1001, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:2.2.2.2/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:47
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1002, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:3.3.3.3/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:47
                Validation State: unverified
                Task: IS-IS
                AS path: I
```



```
                Prefix SID: 1003, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:4.4.4.4/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:47
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1004, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:5.5.5.5/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:47
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1005, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:6.6.6.6/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:47
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1006, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:7.7.7.7/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:47
```

```
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1007, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:47
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0x40, Algo: 0
```

show route te-ipv4-prefix-node-iso detail

```
user@host> show route te-ipv4-prefix-node-iso 0100.0a0a.0a0a.00 detail
lsdist.0: 283 destinations, 283 routes (283 active, 0 holddown, 0 hidden)
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:10.10.10.10/32 } ISIS-L1:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 15
                Level: 1
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:54
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0x40, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:1.1.1.1/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:54
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1001, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:2.2.2.2/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
```

```
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:54
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1002, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:3.3.3.3/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:54
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1003, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:4.4.4.4/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:54
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1004, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:5.5.5.5/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:54
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1005, Flags: 0xe0, Algo: 0
```



```
PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:6.6.6.6/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:54
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1006, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:7.7.7.7/32 } ISIS-L2:0
}/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:54
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1007, Flags: 0xe0, Algo: 0

PREFIX { Node { AS:100 ISO:0100.0a0a.0a0a.00 } { IPv4:10.10.10.10/32 } ISIS-L2:0
 }/1152 (1 entry, 0 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Fictitious, Next hop index: 0
                Address: 0xa1a2ac4
                Next-hop reference count: 283
                Next hop:
                State: <Active NotInstall>
                Local AS:   100
                Age: 6:54
                Validation State: unverified
                Task: IS-IS
                AS path: I
                Prefix SID: 1000, Flags: 0x40, Algo: 0
```

show route active-path

List of Syntax

Syntax on page 1392
Syntax (EX Series Switches) on page 1392

Syntax

show route active-path
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route active-path
<brief | detail | extensive | terse>

Release Information

Command introduced in Junos OS Release 8.0.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display all active routes for destinations. An active route is a route that is selected as the best path. Inactive routes are not displayed.

Options

none—Display all active routes.

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show route active-path on page 1392
show route active-path brief on page 1393
show route active-path detail on page 1393
show route active-path extensive on page 1394
show route active-path terse on page 1396

Output Fields

For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route active-path

```
user@host> show route active-path

inet.0: 7 destinations, 7 routes (6 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.70.19/32    *[Direct/0] 21:33:52
```

```
                          > via lo0.0
10.255.71.50/32        *[IS-IS/15] 00:18:13, metric 10
                          > to 172.16.100.1 via so-2/1/3.0
172.16.100.1/24           *[Direct/0] 00:18:36
                          > via so-2/1/3.0
172.16.100.1/32           *[Local/0] 00:18:41
                            Local via so-2/1/3.0
192.168.64.0/21        *[Direct/0] 21:33:52
                          > via fxp0.0
192.168.70.19/32       *[Local/0] 21:33:52
                            Local via fxp0.0
```

show route active-path brief

The output for the **show route active-path brief** command is identical to that for the **show route active-path** command. For sample output, see show route active-path on page 1392.

show route active-path detail

```
user@host> show route active-path detail

inet.0: 7 destinations, 7 routes (6 active, 0 holddown, 1 hidden)

10.255.70.19/32 (1 entry, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via lo0.0, selected
                State: <Active Int>
                Local AS:   200
                Age: 21:37:10
                Task: IF
                Announcement bits (3): 2-IS-IS 5-Resolve tree 2 6-Resolve tree 3

                AS path: I

10.255.71.50/32 (1 entry, 1 announced)
        *IS-IS  Preference: 15
                Level: 1
                Next hop type: Router, Next hop index: 397
                Next-hop reference count: 4
                Next hop: 172.16.100.1 via so-2/1/3.0, selected
                State: <Active Int>
                Local AS:   200
                Age: 21:31      Metric: 10
                Task: IS-IS
                Announcement bits (4): 0-KRT 2-IS-IS 5-Resolve tree 2 6-Resolve
tree 3
                AS path: I

172.16.100.0/24 (1 entry, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via so-2/1/3.0, selected
                State: <Active Int>
                Local AS:   200
                Age: 21:54
                Task: IF
                Announcement bits (3): 2-IS-IS 5-Resolve tree 2 6-Resolve tree 3
```



```
                AS path: I

172.16.100.1/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 11
                Interface: so-2/1/3.0
                State: <Active NoReadvrt Int>
                Local AS:   200
                Age: 21:59
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I

192.168.64.0/21 (1 entry, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via fxp0.0, selected
                State: <Active Int>
                Local AS:   200
                Age: 21:37:10
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I

192.168.70.19/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 11
                Interface: fxp0.0
                State: <Active NoReadvrt Int>
                Local AS:   200
                Age: 21:37:10
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I
```

show route active-path extensive

```
user@host> show route active-path extensive

inet.0: 7 destinations, 7 routes (6 active, 0 holddown, 1 hidden)
10.255.70.19/32 (1 entry, 1 announced)
TSI:
IS-IS level 1, LSP fragment 0
IS-IS level 2, LSP fragment 0
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via lo0.0, selected
                State: <Active Int>
                Local AS:   200
                Age: 21:39:47
                Task: IF
                Announcement bits (3): 2-IS-IS 5-Resolve tree 2 6-Resolve tree 3

                AS path: I

10.255.71.50/32 (1 entry, 1 announced)
```

```
TSI:
KRT in-kernel 10.255.71.50/32 -> {172.16.100.1}
IS-IS level 2, LSP fragment 0
        *IS-IS  Preference: 15
                Level: 1
                Next hop type: Router, Next hop index: 397
                Next-hop reference count: 4
                Next hop: 172.16.100.1 via so-2/1/3.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 24:08      Metric: 10
                Task: IS-IS
                Announcement bits (4): 0-KRT 2-IS-IS 5-Resolve tree 2 6-Resolve
tree 3
                AS path: I

172.16.100.1/24 (1 entry, 1 announced)
TSI:
IS-IS level 1, LSP fragment 0
IS-IS level 2, LSP fragment 0
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via so-2/1/3.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 24:31
                Task: IF
                Announcement bits (3): 2-IS-IS 5-Resolve tree 2 6-Resolve tree 3

                AS path: I

172.16.100.1/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 11
                Interface: so-2/1/3.0
                State: ‹Active NoReadvrt Int›
                Local AS:   200
                Age: 24:36
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I

192.168.64.0/21 (1 entry, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 3
                Next hop: via fxp0.0, selected
                State: ‹Active Int›
                Local AS:   200
                Age: 21:39:47
                Task: IF
                Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
                AS path: I

192.168.70.19/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 11
                Interface: fxp0.0
```

```
State: ‹Active NoReadvrt Int›
Local AS:   200
Age: 21:39:47
Task: IF
Announcement bits (2): 5-Resolve tree 2 6-Resolve tree 3
AS path: I
```

show route active-path terse

```
user@host> show route active-path terse

inet.0: 7 destinations, 7 routes (6 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.255.70.19/32    D   0                       >lo0.0
* 10.255.71.50/32    I  15         10            >172.16.100.1.
* 172.16.100.0/24       D   0                       >so-2/1/3.0
* 172.16.100.2/32       L   0                        Local
* 192.168.64.0/21    D   0                       >fxp0.0
* 192.168.70.19/32   L   0                        Local
```

show route advertising-protocol

Syntax

show route advertising-protocol *protocol neighbor-address*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Release Information

Command introduced before Junos OS Release 7.4.

Description

Display the routing information as it has been prepared for advertisement to a particular neighbor of a particular dynamic routing protocol.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

neighbor-address—Address of the neighboring router to which the route entry is being transmitted.

protocol—Protocol transmitting the route:

- **bgp**—Border Gateway Protocol
- **dvmrp**—Distance Vector Multicast Routing Protocol
- **msdp**—Multicast Source Discovery Protocol
- **pim**—Protocol Independent Multicast
- **rip**—Routing Information Protocol
- **ripng**—Routing Information Protocol next generation

Additional Information

Routes displayed are routes that the routing table has exported into the routing protocol and that have been filtered by the associated protocol's **export** routing policy statements. Starting with Junos OS Release 13.3, you can display the routing instance table **foo** for any address family, on a VPN route reflector, or a VPN AS boundary router that is advertising local VPN routes. However, If you do not specify the **table** in the command, the output displays each VRF prefix twice.

Required Privilege Level

view

Related Documentation

- *Example: Configuring the MED Attribute That Determines the Exit Point in an AS*

List of Sample Output

show route advertising-protocol bgp (Layer 3 VPN) on page 1400
show route advertising-protocol bgp detail on page 1400
show route advertising-protocol bgp detail (Labeled Unicast) on page 1400

show route advertising-protocol bgp detail (Layer 2 VPN) on page 1401
show route advertising-protocol bgp detail (Layer 3 VPN) on page 1401
show route advertising-protocol bgp extensive all (Next Hop Self with RIB-out IP Address) on page 1401

Output Fields Table 81 on page 1398 lists the output fields for the **show route advertising-protocol** command. Output fields are listed in the approximate order in which they appear.

Table 81: show route advertising-protocol Output Fields

Field Name	Field Description	Level of Output
routing-table-name	Name of the routing table—for example, inet.0.	All levels
number **destinations**	Number of destinations for which there are routes in the routing table.	All levels
number **routes**	Number of routes in the routing table and total number of routes in the following states: • **active** (routes that are active) • **holddown** (routes that are in the pending state before being declared inactive) • **hidden** (routes that are not used because of a routing policy)	All levels
Prefix	Destination prefix.	**brief** none
destination-prefix **(entry , announced)**	Destination prefix. The **entry** value is the number of routes for this destination, and the **announced** value is the number of routes being announced for this destination.	**detail extensive**
BGP group and type	BGP group name and type (**Internal** or **External**).	**detail extensive**
Route Distinguisher	Unique 64-bit prefix augmenting each IP subnet.	**detail extensive**
Advertised Label	Incoming label advertised by the Label Distribution Protocol (LDP). When an IP packet enters a label-switched path (LSP), the ingress router examines the packet and assigns it a label based on its destination, placing the label in the packet's header. The label transforms the packet from one that is forwarded based on its IP routing information to one that is forwarded based on information associated with the label.	**detail extensive**
Label-Base, range	First label in a block of labels and label block size. A remote PE router uses this first label when sending traffic toward the advertising PE router.	**detail extensive**
VPN Label	Virtual private network (VPN) label. Packets are sent between CE and PE routers by advertising VPN labels. VPN labels transit over either a Resource Reservation Protocol (RSVP) or a Label Distribution Protocol (LDP) label-switched path (LSP) tunnel.	**detail extensive**

Table 81: show route advertising-protocol Output Fields *(continued)*

Field Name	Field Description	Level of Output
Nexthop	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route. If the next-hop advertisement to the peer is **Self**, and the RIB-out next hop is a specific IP address, the RIB-out IP address is included in the extensive output. See show route advertising-protocol bgp extensive all (Next Hop Self with RIB-out IP Address) on page 1401.	All levels
MED	Multiple exit discriminator value included in the route.	**brief**
Lclpref or **Localpref**	Local preference value included in the route.	All levels
Queued	When BGP route prioritization is enabled and a route is present in a priority queue, this shows which priority queue the route is in.	All levels except **brief**
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the local AS number associated with the AS path if configured on the router, or if AS path prepending is configured. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.	All levels
Route Labels	Stack of labels carried in the BGP route update.	**detail extensive**
Cluster list	(For route reflected output only) Cluster ID sent by the route reflector.	**detail extensive**
Originator ID	(For route reflected output only) Address of routing device that originally sent the route to the route reflector.	**detail extensive**
Communities	Community path attribute for the route. See the output field table for the show route detail command for all possible values for this field.	**detail extensive**
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.	**detail extensive**

Table 81: show route advertising-protocol Output Fields *(continued)*

Field Name	Field Description	Level of Output
Attrset AS	Number, local preference, and path of the autonomous system (AS) that originated the route. These values are stored in the **Attrset** attribute at the originating router.	**detail extensive**
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).	**detail extensive**
control flags	Control flags: **none** or **Site Down**.	**detail extensive**
mtu	Maximum transmission unit (MTU) of the Layer 2 circuit.	**detail extensive**

Sample Output

show route advertising-protocol bgp (Layer 3 VPN)

```
user@host> show route advertising-protocol bgp 10.255.14.171
      VPN-A.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
      Prefix            Nexthop                 MED    Lclpref AS path
      10.255.14.172/32  Self                     1        100 I
      VPN-B.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
      Prefix            Nexthop                 MED    Lclpref AS path
      10.255.14.181/32  Self                     2        100 I
```

show route advertising-protocol bgp detail

```
user@host> show route advertising-protocol bgp 111.222.1.3 detail
bgp20.inet.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
111.222.1.11/32 (1 entry, 1 announced)
 BGP group pe-pe type Internal
     Route Distinguisher: 111.255.14.11:69
     Advertised Label: 100000
     next hop: Self
     Localpref: 100
     AS path: 2 I
     Communities: target:69:20
     AIGP 210
111.8.0.0/16 (1 entry, 1 announced)
 BGP group pe-pe type Internal
     Route Distinguisher: 111.255.14.11:69
     Advertised Label: 100000
     Next hop: Self
     Localpref: 100
     AS path: 2 I
     Communities: target:69:20
     AIGP 210
```

show route advertising-protocol bgp detail (Labeled Unicast)

```
user@host>show route advertising bgp 1.1.1.3 detail
inet.0: 69 destinations, 70 routes (69 active, 0 holddown, 0 hidden)
* 1.1.1.8/32 (2 entries, 2 announced)
BGP group ibgp type Internal
Route Labels: 1000123(top) 1000124 1000125 1000126
Nexthop: 1.1.1.4
```

```
MED: 7
Localpref: 100
AS path: [5] I
Cluster ID: 3.3.3.3
Originator ID: 1.1.1.1
Entropy label capable
inet6.0: 26 destinations, 28 routes (26 active, 0 holddown, 0 hidden)
* 100::1/128 (2 entries, 1 announced)
BGP group ibgp type Internal
Labels: 1000123(top) 1000124 1000125 1000126
Nexthop: ::ffff:1.1.1.4
Localpref: 100
AS path: [5] I
Cluster ID: 3.3.3.3
Originator ID: 1.1.1.1
```

show route advertising-protocol bgp detail (Layer 2 VPN)

```
user@host> show route advertising-protocol bgp 192.168.24.1 detail
vpn-a.l2vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
192.168.16.1:1:1:1/96 (1 entry, 1 announced)
 BGP group int type Internal
     Route Distinguisher: 192.168.16.1:1
     Label-base : 32768, range : 3
     Nexthop: Self
     Localpref: 100
     AS path: I
     Communities: target:65412:100
     AIGP 210
     Layer2-info: encaps:VLAN, control flags:, mtu:
```

show route advertising-protocol bgp detail (Layer 3 VPN)

```
user@host> show route advertising-protocol bgp 10.255.14.176 detail
vpna.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
* 10.49.0.0/30 (1 entry, 1 announced)
 BGP group ibgp type Internal
     Route Distinguisher: 10.255.14.174:2
     VPN Label: 101264
     Nexthop: Self
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AIGP 210
     AttrSet AS: 100
         Localpref: 100
         AS path: I
...
```

show route advertising-protocol bgp extensive all (Next Hop Self with RIB-out IP Address)

```
user@host> show route advertising-protocol bgp 200.0.0.2 170.0.1.0/24 extensive all
inet.0: 13 destinations, 19 routes (13 active, 0 holddown, 6 hidden)
  170.0.1.0/24 (2 entries, 1 announced)
 BGP group eBGP-INTEROP type External
     Nexthop: Self (rib-out 10.100.3.2)
     AS path: [4713] 200 I
...
```


show route all

List of Syntax

Syntax on page 1402
Syntax (EX Series Switches) on page 1402

Syntax

show route all
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route all

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display information about all routes in all routing tables, including private, or internal, tables.

Options

none—Display information about all routes in all routing tables, including private, or internal, tables.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

Related Documentation

- show route brief on page 1409
- show route detail on page 1420

List of Sample Output

show route all on page 1402

Output Fields

In Junos OS Release 9.5 and later, only the output fields for the **show route all** command display all routing tables, including private, or hidden, routing tables. The output field table of the show route command does not display entries for private, or hidden, routing tables in Junos OS Release 9.5 and later.

Sample Output

show route all

The following example displays a snippet of output from the **show route** command and then displays the same snippet of output from the **show route all** command:

```
user@host> show route
mpls.0: 7 destinations, 7 routes (5 active, 0 holddown, 2 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
0                  *[MPLS/0] 2d 02:24:39, metric 1
```



```
                       Receive
1                   *[MPLS/0] 2d 02:24:39, metric 1
                       Receive
2                   *[MPLS/0] 2d 02:24:39, metric 1
                       Receive
800017              *[VPLS/7] 1d 14:00:16
                     > via vt-3/2/0.32769, Pop
800018              *[VPLS/7] 1d 14:00:26
                     > via vt-3/2/0.32772, Pop
```

```
user@host> show route all
mpls.0: 7 destinations, 7 routes (5 active, 0 holddown, 2 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
0                   *[MPLS/0] 2d 02:19:12, metric 1
                       Receive
1                   *[MPLS/0] 2d 02:19:12, metric 1
                       Receive
2                   *[MPLS/0] 2d 02:19:12, metric 1
                       Receive
800017              *[VPLS/7] 1d 13:54:49
                     > via vt-3/2/0.32769, Pop
800018              *[VPLS/7] 1d 13:54:59
                     > via vt-3/2/0.32772, Pop
vt-3/2/0.32769       [VPLS/7] 1d 13:54:49
                       Unusable
vt-3/2/0.32772       [VPLS/7] 1d 13:54:59
                       Unusable
```

show route aspath-regex

List of Syntax

Syntax on page 1404
Syntax (EX Series Switches) on page 1404

Syntax

show route aspath-regex *regular-expression*
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route aspath-regex *regular-expression*

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the entries in the routing table that match the specified autonomous system (AS) path regular expression.

Options

regular-expression—Regular expression that matches an entire AS path.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Additional Information

You can specify a regular expression as:

- An individual AS number
- A period wildcard used in place of an AS number
- An AS path regular expression that is enclosed in parentheses

You also can include the operators described in the table of AS path regular expression operators in the *Junos Policy Framework Configuration Guide*. The following list summarizes these operators:

- **{*m*,*n*}**—At least *m* and at most *n* repetitions of the AS path term.
- **{*m*}**—Exactly *m* repetitions of the AS path term.
- **{*m*,}**—*m* or more repetitions of the AS path term.
- *****—Zero or more repetitions of an AS path term.
- ***+***—One or more repetitions of an AS path term.
- ***?***—Zero or one repetition of an AS path term.
- ***aspath_term* | *aspath_term***—Match one of the two AS path terms.

  When you specify more than one AS number or path term, or when you include an operator in the regular expression, enclose the entire regular expression in quotation marks. For example, to match any path that contains AS number 234, specify the following command:

show route aspath-regex ".* 234 .*"

Required Privilege Level view

Related Documentation

- Example: Using AS Path Regular Expressions on page 316

List of Sample Output

show route aspath-regex (Matching a Specific AS Number) on page 1405
show route aspath-regex (Matching Any Path with Two AS Numbers) on page 1405

Output Fields For information about output fields, see the output field table for the show route command.

Sample Output

show route aspath-regex (Matching a Specific AS Number)

```
user@host> show route aspath-regex 65477
inet.0: 46411 destinations, 46411 routes (46409 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

111.222.1.0/25     *[BGP/170] 00:08:48, localpref 100, from 111.222.2.24
                   AS Path: [65477] ({65548 65536}) IGP
                     to 111.222.18.225 via fpa0.0(111.222.18.233)
111.222.1.128/25   *[IS-IS/15] 09:15:37, metric 37, tag 1
                     to 111.222.18.225 via fpa0.0(111.222.18.233)
                    [BGP/170] 00:08:48, localpref 100, from 111.222.2.24
                   AS Path: [65477] ({65548 65536}) IGP
                     to 111.222.18.225 via fpa0.0(111.222.18.233)
...
```

show route aspath-regex (Matching Any Path with Two AS Numbers)

```
user@host> show route aspath-regex ".* 234 3561 .*"

inet.0: 46351 destinations, 46351 routes (46349 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

9.20.0.0/17        *[BGP/170] 01:35:00, localpref 100, from 131.103.20.49
                   AS Path: [666] 234 3561 2685 2686 Incomplete
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
12.10.231.0/24     *[BGP/170] 01:35:00, localpref 100, from 131.103.20.49
                   AS Path: [666] 234 3561 5696 7369 IGP
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
24.64.32.0/19      *[BGP/170] 01:34:59, localpref 100, from 131.103.20.49
                   AS Path: [666] 234 3561 6327 IGP
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
...
```


show route best

List of Syntax

Syntax on page 1406
Syntax (EX Series Switches) on page 1406

Syntax

show route best *destination-prefix*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route best *destination-prefix*
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the route in the routing table that is the best route to the specified address or range of addresses. The best route is the longest matching route.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

destination-prefix—Address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

Related Documentation

- show route brief on page 1409
- show route detail on page 1420

List of Sample Output

show route best on page 1406
show route best detail on page 1407
show route best extensive on page 1408
show route best terse on page 1408

Output Fields

For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route best

`user@host>` **show route best 10.255.70.103**

```
inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
10.255.70.103/32   *[OSPF/10] 1d 13:19:20, metric 2
                    > to 10.31.1.6 via ge-3/1/0.0
                      via so-0/3/0.0

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
10.255.70.103/32   *[RSVP/7] 1d 13:20:13, metric 2
                    > via so-0/3/0.0, label-switched-path green-r1-r3

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
10.0.0.0/8         *[Direct/0] 2d 01:43:34
                    > via fxp2.0
                     [Direct/0] 2d 01:43:34
                    > via fxp1.0
```

show route best detail

```
user@host> show route best 10.255.70.103 detail
inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
10.255.70.103/32 (1 entry, 1 announced)
        *OSPF   Preference: 10
                Next-hop reference count: 9
                Next hop: 10.31.1.6 via ge-3/1/0.0, selected
                Next hop: via so-0/3/0.0
                State: <Active Int>
                Local AS:    69
                Age: 1d 13:20:06        Metric: 2
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete
10.255.70.103/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 5
                Next hop: via so-0/3/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 100016
                State: <Active Int>
                Local AS:    69
                Age: 1d 13:20:59        Metric: 2
                Task: RSVP
                Announcement bits (1): 1-Resolve tree 2
                AS path: I

private1__inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
10.0.0.0/8 (2 entries, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via fxp2.0, selected
                State: <Active Int>
```



```
                Age: 2d 1:44:20
                Task: IF
                AS path: I
         Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via fxp1.0, selected
                State: <NotBest Int>
                Inactive reason: No difference
                Age: 2d 1:44:20
                Task: IF
                AS path: I
```

show route best extensive

The output for the **show route best extensive** command is identical to that for the **show route best detail** command. For sample output, see show route best detail on page 1407.

show route best terse

```
user@host> show route best 10.255.70.103 terse
inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.255.70.103/32   O  10          2            >10.31.1.6
                                                  so-0/3/0.0

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.255.70.103/32   R   7          2            >so-0/3/0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.0.0.0/8         D   0                       >fxp2.0
                     D   0                       >fxp1.0
```


show route brief

List of Syntax Syntax on page 1409
Syntax (EX Series Switches) on page 1409

Syntax show route brief
<*destination-prefix*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches) show route brief
<*destination-prefix*>

Release Information Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description Display brief information about the active entries in the routing tables.

Options **none**—Display all active entries in the routing table.

destination-prefix—(Optional) Display active entries for the specified address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level view

Related Documentation

- show route all on page 1402
- show route best on page 1406

List of Sample Output show route brief on page 1409

Output Fields For information about output fields, see the Output Field table of the show route command.

Sample Output

show route brief

```
user@host> show route brief
inet.0: 10 destinations, 10 routes (9 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 1w5d 20:30:29
                      Discard
10.255.245.51/32   *[Direct/0] 2w4d 13:11:14
                    > via lo0.0
```

```
172.16.0.0/12      *[Static/5] 2w4d 13:11:14
                    > to 192.168.167.254 via fxp0.0
192.168.0.0/18      *[Static/5] 1w5d 20:30:29
                    > to 192.168.167.254 via fxp0.0
192.168.40.0/22    *[Static/5] 2w4d 13:11:14
                    > to 192.168.167.254 via fxp0.0
192.168.64.0/18    *[Static/5] 2w4d 13:11:14
                    > to 192.168.167.254 via fxp0.0
192.168.164.0/22   *[Direct/0] 2w4d 13:11:14
                    > via fxp0.0
192.168.164.51/32  *[Local/0] 2w4d 13:11:14
                      Local via fxp0.0
207.17.136.192/32  *[Static/5] 2w4d 13:11:14
                    > to 192.168.167.254 via fxp0.0
green.inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
100.101.0.0/16     *[Direct/0] 1w5d 20:30:28
                    > via fe-0/0/3.0
100.101.2.3/32     *[Local/0] 1w5d 20:30:28
                      Local via fe-0/0/3.0
172.16.233.5/32       *[OSPF/10] 1w5d 20:30:29, metric 1
                      MultiRecv
```


show route community

List of Syntax

Syntax on page 1411
Syntax (EX Series Switches) on page 1411

Syntax

show route community *as-number:community-value*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route community *as-number:community-value*
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the route entries in each routing table that are members of a Border Gateway Protocol (BGP) community.

Options

as-number:community-value—One or more community identifiers. ***as-number*** is the AS number, and ***community-value*** is the community identifier. When you specify more than one community identifier, enclose the identifiers in double quotation marks. Community identifiers can include wildcards.

For example:

user@host> **show route table inet.0 protocol bgp community "12083:6015" community "12083:65551"**

or

user@host> **show route table inet.0 protocol bgp community [12083:6014 12083:65551]**

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Additional Information

Specifying the community option displays all routes matching the community found within the routing table. The community option does not limit the output to only the routes being advertised to the neighbor after any egress routing policy.

Required Privilege Level

view

Related Documentation

- show route detail on page 1420

List of Sample Output show route community on page 1412

Output Fields For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route community

```
user@host> show route community 234:80
inet.0: 46511 destinations, 46511 routes (46509 active, 0 holddown, 2 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.4.0/8          *[BGP/170] 03:33:07, localpref 100, from 131.103.20.49
                   AS Path: {666} 234 2548 1 IGP
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
172.16.6.0/8          *[BGP/170] 03:33:07, localpref 100, from 131.103.20.49
                   AS Path: {666} 234 2548 568 721 Incomplete
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
172.16.92.0/16         *[BGP/170] 03:33:06, localpref 100, from 131.103.20.49
                   AS Path: {666} 234 2548 1673 1675 1747 IGP
                     to 192.156.169.1 via 192.156.169.14(so-0/0/0)
```

show route community-name

List of Syntax
Syntax on page 1413
Syntax (EX Series Switches) on page 1413

Syntax
show route community-name *community-name*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)
show route community-name *community-name*
<brief | detail | extensive | terse>

Release Information
Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description
Display the route entries in each routing table that are members of a Border Gateway Protocol (BGP) community, specified by a community name.

Options
community-name—Name of the community.

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level
view

List of Sample Output
show route community-name on page 1413

Output Fields
For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route community-name

```
user@host> show route community-name red-com
inet.0: 17 destinations, 17 routes (16 active, 0 holddown, 1 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

instance1.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.245.212/32   *[BGP/170] 00:04:40, localpref 100, from 10.255.245.204
                      AS path: 300 I
```



```
                    > to 172.16.100.1 via ge-1/1/0.0, label-switched-path to_fix
172.16.20.20/32     *[BGP/170] 00:04:40, localpref 100, from 10.255.245.204
                      AS path: I
                    > to 172.16.100.1 via ge-1/1/0.0, label-switched-path to_fix
172.16.100.0/24       *[BGP/170] 00:04:40, localpref 100, from 10.255.245.204
                      AS path: I
                    > to 172.16.100.1 via ge-1/1/0.0, label-switched-path to_fix

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

bgp.l3vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.245.204:10:10.255.245.212/32
                   *[BGP/170] 00:06:40, localpref 100, from 10.255.245.204
                      AS path: 300 I
                    > to 172.16.100.1 via ge-1/1/0.0, label-switched-path to_fix
10.255.245.204:10:172.16.20.20/32
                   *[BGP/170] 00:36:02, localpref 100, from 10.255.245.204
                      AS path: I
                    > to 172.16.100.1 via ge-1/1/0.0, label-switched-path to_fix
10.255.245.204:10:100.1.4.0/24
                   *[BGP/170] 00:36:02, localpref 100, from 10.255.245.204
                      AS path: I
                    > to 172.16.100.1 via ge-1/1/0.0, label-switched-path to_fix

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

instance1.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route damping

List of Syntax

Syntax on page 1415
Syntax (EX Series Switch and QFX Series) on page 1415

Syntax

show route damping (decayed | history | suppressed)
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switch and QFX Series)

show route damping (decayed | history | suppressed)
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Command introduced in Junos OS Release 11.3 for the QFX Series.
Command introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description

Display the BGP routes for which updates might have been reduced because of route flap damping.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to brief.

decayed—Display route damping entries that might no longer be valid, but are not suppressed.

history—Display entries that have already been withdrawn, but have been logged.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

suppressed—Display entries that have been suppressed and are no longer being installed into the forwarding table or exported by routing protocols.

Required Privilege Level

view

Related Documentation

- *clear bgp damping*
- show policy damping on page 1369

List of Sample Output

show route damping decayed detail on page 1418
show route damping history on page 1419
show route damping history detail on page 1419

Output Fields

Table 82 on page 1416 lists the output fields for the **show route damping** command. Output fields are listed in the approximate order in which they appear.

Table 82: show route damping Output Fields

Field Name	Field Description	Level of Output
routing-table-name	Name of the routing table—for example, **inet.0**.	All levels
destinations	Number of destinations for which there are routes in the routing table.	All levels
number **routes**	Number of routes in the routing table and total number of routes in the following states: • **active** • **holddown** (routes that are in a pending state before being declared inactive) • **hidden** (the routes are not used because of a routing policy)	All levels
destination-prefix **(entry, announced)**	Destination prefix. The **entry** value is the number of routes for this destination, and the **announced** value is the number of routes being announced for this destination.	**detail extensive**
[*protocol***,** *preference***]**	Protocol from which the route was learned and the preference value for the route. • **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • **-** —A hyphen indicates the last active route. • *****—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route. In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the LocalPref value for Route 1 is 100, the Preference2 value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value and a lower **Preference2** value.	All levels
Next-hop reference count	Number of references made to the next hop.	**detail extensive**
Source	IP address of the route source.	**detail extensive**
Next hop	Network layer address of the directly reachable neighboring system.	**detail extensive**
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the interface that is actually used is followed by the word **Selected**.	**detail extensive**
Protocol next hop	Network layer address of the remote routing device that advertised the prefix. This address is used to derive a forwarding next hop.	**detail extensive**
Indirect next hop	Index designation used to specify the mapping between protocol next hops, tags, kernel export policy, and the forwarding next hops.	**detail extensive**
State	Flags for this route. For a description of possible values for this field, see the output field table for the show route detail command.	**detail extensive**

Table 82: show route damping Output Fields *(continued)*

Field Name	Field Description	Level of Output
Local AS	AS number of the local routing device.	**detail extensive**
Peer AS	AS number of the peer routing device.	**detail extensive**
Age	How long the route has been known.	**detail extensive**
Metric	Metric for the route.	**detail extensive**
Task	Name of the protocol that has added the route.	**detail extensive**
Announcement bits	List of protocols that announce this route. ***n*-Resolve inet** indicates that the route is used for route resolution for next hops found in the routing table. ***n*** is an index used by Juniper Networks customer support only.	**detail extensive**
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the local AS number associated with the AS path if more than one AS number is configured on the routing device or if AS path prepending is configured. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.	All levels
to	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route.	**brief** none
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the interface that is actually used is followed by the word **Selected**.	**brief** none
Communities	Community path attribute for the route. See the output field table for the show route detail command.	**detail extensive**
Localpref	Local preference value included in the route.	All levels
Router ID	BGP router ID as advertised by the neighbor in the open message.	**detail extensive**

Table 82: show route damping Output Fields *(continued)*

Field Name	Field Description	Level of Output
Merit (last update/now)	Last updated and current figure-of-merit value.	**detail extensive**
damping-parameters	Name that identifies the damping parameters used, which is defined in the damping statement at the **[edit policy-options]** hierarchy level.	**detail extensive**
Last update	Time of most recent change in path attributes.	**detail extensive**
First update	Time of first change in path attributes, which started the route damping process.	**detail extensive**
Flaps	Number of times the route has gone up or down or its path attributes have changed.	**detail extensive**
Suppressed	(**suppressed** keyword only) This route is currently suppressed. A suppressed route does not appear in the forwarding table and routing protocols do not export it.	All levels
Reusable in	(**suppressed** keyword only) Time when a suppressed route will again be available.	All levels
Preference will be	(**suppressed** keyword only) Preference value that will be applied to the route when it is again active.	All levels

Sample Output

show route damping decayed detail

```
user@host> show route damping decayed detail
inet.0: 173319 destinations, 1533668 routes (172625 active, 4 holddown, 108083
hidden)
10.0.111.0/24 (7 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next-hop reference count: 151973
                Source: 172.23.2.129
                Next hop: via so-1/2/0.0
                Next hop: via so-5/1/0.0, selected
                Next hop: via so-6/0/0.0
                Protocol next hop: 172.23.2.129
                Indirect next hop: 89a1a00 264185
                State: <Active Ext>
                Local AS: 64500 Peer AS:   64490
                Age: 3:28       Metric2: 0
                Task: BGP_64490.172.23.2.129+179
                Announcement bits (6): 0-KRT 1-RT 4-KRT 5-BGP.0.0.0.0+179

         6-Resolve tree 2 7-Resolve tree 3
                AS path: 64499 64510 645511 645511 645511 645511 I ()
                Communities: 65551:390 65551:2000 65551:3000 65550:701
                Localpref: 100
                Router ID: 172.23.2.129
                Merit (last update/now): 1934/1790
                damping-parameters: damping-high
```

```
                Last update:       00:03:28 First update:       00:06:40
                Flaps: 2
```

show route damping history

user@host> **show route damping history**

```
inet.0: 173320 destinations, 1533529 routes (172624 active, 6 holddown, 108122
hidden)
+ = Active Route, - = Last Active, * = Both

10.108.0.0/15       [BGP ] 2d 22:47:58, localpref 100
                      AS path: 64220 65541 65542 I
                    > to 192.168.60.85 via so-3/1/0.0
```

show route damping history detail

user@host> **show route damping history detail**

```
inet.0: 173319 destinations, 1533435 routes (172627 active, 2 holddown, 108105
hidden)
10.108.0.0/15 (3 entries, 1 announced)
         BGP                 /-101
                Next-hop reference count: 69058
                Source: 192.168.60.85
                Next hop: 192.168.60.85 via so-3/1/0.0, selected
                State: <Hidden Ext>
                Inactive reason: Unusable path
                Local AS: 64500 Peer AS:  64220
                Age: 2d 22:48:10
                Task: BGP_64220.192.168.60.85+179
                AS path: 64220 65541 65542 I ()
                Communities: 65541:390 65541:2000 65541:3000 65504:3561
                Localpref: 100
                Router ID: 192.168.80.25
                Merit (last update/now): 1000/932
                damping-parameters: set-normal
                Last update:       00:01:05 First update:       00:01:05
                Flaps: 1
```


show route detail

List of Syntax

Syntax on page 1420
Syntax (EX Series Switches) on page 1420

Syntax

show route detail
<*destination-prefix*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route detail
<*destination-prefix*>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Command introduced in Junos OS Release 13.2X51-D15 for the QFX Series.
Comand introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description

Display detailed information about the active entries in the routing tables.

Options

none—Display all active entries in the routing table on all systems.

destination-prefix—(Optional) Display active entries for the specified address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show route detail on page 1431
show route detail (with BGP Multipath) on page 1437
show route label detail (Multipoint LDP Inband Signaling for Point-to-Multipoint LSPs) on page 1438
show route label detail (Multipoint LDP with Multicast-Only Fast Reroute) on page 1438

Output Fields

Table 83 on page 1420 describes the output fields for the **show route detail** command. Output fields are listed in the approximate order in which they appear.

Table 83: show route detail Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).
number **destinations**	Number of destinations for which there are routes in the routing table.

Table 83: show route detail Output Fields *(continued)*

Field Name	Field Description
number routes	Number of routes in the routing table and total number of routes in the following states: • **active** (routes that are active) • **holddown** (routes that are in the pending state before being declared inactive) • **hidden** (routes that are not used because of a routing policy)

route-destination (entry, announced)	Route destination (for example:10.0.0.1/24). The **entry** value is the number of routes for this destination, and the **announced** value is the number of routes being announced for this destination. Sometimes the route destination is presented in another format, such as:
	• *MPLS-label* (for example, 80001).
	• *interface-name* (for example, ge-1/0/2).
	• *neighbor-address*:*control-word-status*:*encapsulation type*:*vc-id*:*source* (Layer 2 circuit only; for example, 10.1.1.195:NoCtrlWord:1:1:Local/96).
	  ◦ *neighbor-address*—Address of the neighbor.
	  ◦ *control-word-status*—Whether the use of the control word has been negotiated for this virtual circuit: **NoCtrlWord** or **CtrlWord**.
	  ◦ *encapsulation type*—Type of encapsulation, represented by a number: (1) Frame Relay DLCI, (2) ATM AAL5 VCC transport, (3) ATM transparent cell transport, (4) Ethernet, (5) VLAN Ethernet, (6) HDLC, (7) PPP, (8) ATM VCC cell transport, (10) ATM VPC cell transport.
	  ◦ *vc-id*—Virtual circuit identifier.
	  ◦ *source*—Source of the advertisement: **Local** or **Remote**.
	  ◦ *source*—Source of the advertisement: **Local** or **Remote**.

label stacking	(Next-to-the-last-hop routing device for MPLS only) Depth of the MPLS label stack, where the label-popping operation is needed to remove one or more labels from the top of the stack. A pair of routes is displayed, because the pop operation is performed only when the stack depth is two or more labels. • **S=0 route** indicates that a packet with an incoming label stack depth of 2 or more exits this routing device with one fewer label (the label-popping operation is performed). • If there is no **S=** information, the route is a normal MPLS route, which has a stack depth of 1 (the label-popping operation is not performed).

Table 83: show route detail Output Fields *(continued)*

Field Name	Field Description

[*protocol*, *preference*]	Protocol from which the route was learned and the preference value for the route.
	• **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table.
	• **-** —A hyphen indicates the last active route.
	• *****—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route.
	
	In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the **LocalPref** value for Route 1 is 100, the **Preference2** value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value.
	
	Preference2 values are signed integers, that is, **Preference2** values can be either positive or negative values. However, Junos OS evaluates **Preference2** values as unsigned integers that are represented by positive values. Based on the **Preference2** values, Junos OS evaluates a preferred route differently in the following scenarios:
	• **Both Signed Preference2 values**
	  ◦ Route A = -101
	  ◦ Route B = -156
	  Where both the **Preference2** values are signed, Junos OS evaluates only the unsigned value of **Preference2** and Route A, which has a lower **Preference2** value is preferred.
	• **Unsigned Preference2 values**
	  Now consider both unsigned **Preference2** values:
	  ◦ Route A = 4294967096
	  ◦ Route B = 200
	  Here, Junos OS considers the lesser **Preference2** value and Route B with a **Preference2** value of 200 is preferred because it is less than 4294967096.
	• **Combination of signed and unsigned Preference2 values**
	  When **Preference2** values of two routes are compared, and for one route the **Preference2** is a signed value, and for the other route it is an unsigned value, Junos OS prefers the route with the positive **Preference2** value over the negative **Preference2** value. For example, consider the following signed and unsigned **Preference2** values:
	  ◦ Route A = -200
	  ◦ Route B = 200
	  In this case, Route B with a **Preference2** value of 200 is preferred although this value is greater than -200, because Junos OS evaluates only the unsigned value of the **Preference2** value.

Level	(IS-IS only). In IS-IS, a single AS can be divided into smaller groups called areas. Routing between areas is organized hierarchically, allowing a domain to be administratively divided into smaller areas. This organization is accomplished by configuring Level 1 and Level 2 intermediate systems. Level 1 systems route within an area. When the destination is outside an area, they route toward a Level 2 system. Level 2 intermediate systems route between areas and toward other ASs.
Route Distinguisher	IP subnet augmented with a 64-bit prefix.
PMSI	Provider multicast service interface (MVPN routing table).

Table 83: show route detail Output Fields *(continued)*

Field Name	Field Description
Next-hop type	Type of next hop. For a description of possible values for this field, see Table 84 on page 1426.
Next-hop reference count	Number of references made to the next hop.
Flood nexthop branches exceed maximum message	Indicates that the number of flood next-hop branches exceeded the system limit of 32 branches, and only a subset of the flood next-hop branches were installed in the kernel.
Source	IP address of the route source.
Next hop	Network layer address of the directly reachable neighboring system.
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the name of the interface that is actually used is followed by the word **Selected**. This field can also contain the following information: • Weight—Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible. • Balance—Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a routing device is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.
Label-switched-path ***lsp-path-name***	Name of the LSP used to reach the next hop.
Label operation	MPLS label and operation occurring at this routing device. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label).
Interface	(Local only) Local interface name.
Protocol next hop	Network layer address of the remote routing device that advertised the prefix. This address is used to derive a forwarding next hop.
Indirect next hop	Index designation used to specify the mapping between protocol next hops, tags, kernel export policy, and the forwarding next hops.
State	State of the route (a route can be in more than one state). See Table 85 on page 1428.
Local AS	AS number of the local routing device.
Age	How long the route has been known.
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.

Table 83: show route detail Output Fields *(continued)*

Field Name	Field Description
Metric*n*	Cost value of the indicated route. For routes within an AS, the cost is determined by IGP and the individual protocol metrics. For external routes, destinations, or routing domains, the cost is determined by a preference value.
MED-plus-IGP	Metric value for BGP path selection to which the IGP cost to the next-hop destination has been added.
TTL-Action	For MPLS LSPs, state of the TTL propagation attribute. Can be enabled or disabled for all RSVP-signaled and LDP-signaled LSPs or for specific VRF routing instances. For sample output, see show route table.
Task	Name of the protocol that has added the route.
Announcement bits	The number of BGP peers or protocols to which Junos OS has announced this route, followed by the list of the recipients of the announcement. Junos OS can also announce the route to the KRT for installing the route into the Packet Forwarding Engine, to a resolve tree, a L2 VC, or even a VPN. For example, ***n*-Resolve inet** indicates that the specified route is used for route resolution for next hops found in the routing table. • ***n***—An index used by Juniper Networks customer support only.
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **Recorded**—The AS path is recorded by the sample process (sampled). • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the number that precedes the AS path. This number represents the number of ASs present in the AS path, when calculated as defined in RFC 4271. This value is used in the AS-path merge process, as defined in RFC 4893. • **[ ]**—If more than one AS number is configured on the routing device, or if AS path prepending is configured, brackets enclose the local AS number associated with the AS path. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.

Table 83: show route detail Output Fields *(continued)*

Field Name	Field Description
validation-state	(BGP-learned routes) Validation status of the route: • **Invalid**—Indicates that the prefix is found, but either the corresponding AS received from the EBGP peer is not the AS that appears in the database, or the prefix length in the BGP update message is longer than the maximum length permitted in the database. • **Unknown**—Indicates that the prefix is not among the prefixes or prefix ranges in the database. • **Unverified**—Indicates that the origin of the prefix is not verified against the database. This is because the database got populated and the validation is not called for in the BGP import policy, although origin validation is enabled, or the origin validation is not enabled for the BGP peers. • **Valid**—Indicates that the prefix and autonomous system pair are found in the database.
ORR Generation-ID	Displays the optimal route reflection (ORR) generation identifier. ISIS and OSPF interior gateway protocol (IGP) updates filed whenever any of the corresponding ORR route has its metric valued changed, or if the ORR route is added or deleted.
FECs bound to route	Point-to-multipoint root address, multicast source address, and multicast group address when multipoint LDP (M-LDP) inband signaling is configured.
Primary Upstream	When multipoint LDP with multicast-only fast reroute (MoFRR) is configured, the primary upstream path. MoFRR transmits a multicast join message from a receiver toward a source on a primary path, while also transmitting a secondary multicast join message from the receiver toward the source on a backup path.
RPF Nexthops	When multipoint LDP with MoFRR is configured, the reverse-path forwarding (RPF) next-hop information. Data packets are received from both the primary path and the secondary paths. The redundant packets are discarded at topology merge points due to the RPF checks.
Label	Multiple MPLS labels are used to control MoFRR stream selection. Each label represents a separate route, but each references the same interface list check. Only the primary label is forwarded while all others are dropped. Multiple interfaces can receive packets using the same label.
weight	Value used to distinguish MoFRR primary and backup routes. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible.
VC Label	MPLS label assigned to the Layer 2 circuit virtual connection.
MTU	Maximum transmission unit (MTU) of the Layer 2 circuit.
VLAN ID	VLAN identifier of the Layer 2 circuit.
Prefixes bound to route	Forwarding equivalent class (FEC) bound to this route. Applicable only to routes installed by LDP.
Communities	Community path attribute for the route. See Table 86 on page 1430 for all possible values for this field.
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).
control flags	Control flags: **none** or **Site Down**.
mtu	Maximum transmission unit (MTU) information.

Table 83: show route detail Output Fields *(continued)*

Field Name	Field Description
Label-Base, range	First label in a block of labels and label block size. A remote PE routing device uses this first label when sending traffic toward the advertising PE routing device.
status vector	Layer 2 VPN and VPLS network layer reachability information (NLRI).
Accepted Multipath	Current active path when BGP multipath is configured.
Accepted LongLivedStale	The LongLivedStale flag indicates that the route was marked LLGR-stale by this router, as part of the operation of LLGR receiver mode. Either this flag or the LongLivedStaleImport flag may be displayed for a route. Neither of these flags are displayed at the same time as the Stale (ordinary GR stale) flag.
Accepted LongLivedStaleImport	The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy. Either this flag or the LongLivedStale flag may be displayed for a route. Neither of these flags are displayed at the same time as the Stale (ordinary GR stale) flag. Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and import into the inet.0 routing table
ImportAccepted LongLivedStaleImport	Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and imported into the inet.0 routing table The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy.
Accepted MultipathContrib	Path currently contributing to BGP multipath.
Localpref	Local preference value included in the route.
Router ID	BGP router ID as advertised by the neighbor in the open message.
Primary Routing Table	In a routing table group, the name of the primary routing table in which the route resides.
Secondary Tables	In a routing table group, the name of one or more secondary tables in which the route resides.

Table 84 on page 1426 describes all possible values for the Next-hop Types output field.

Table 84: Next-hop Types Output Field Values

Next-Hop Type	Description
Broadcast (bcast)	Broadcast next hop.
Deny	Deny next hop.
Discard	Discard next hop.

Table 84: Next-hop Types Output Field Values *(continued)*

Next-Hop Type	Description
Flood	Flood next hop. Consists of components called branches, up to a maximum of 32 branches. Each flood next-hop branch sends a copy of the traffic to the forwarding interface. Used by point-to-multipoint RSVP, point-to-multipoint LDP, point-to-multipoint CCC, and multicast.
Hold	Next hop is waiting to be resolved into a unicast or multicast type.
Indexed (idxd)	Indexed next hop.
Indirect (indr)	Used with applications that have a protocol next hop address that is remote. You are likely to see this next-hop type for internal BGP (IBGP) routes when the BGP next hop is a BGP neighbor that is not directly connected.
Interface	Used for a network address assigned to an interface. Unlike the router next hop, the interface next hop does not reference any specific node on the network.
Local (locl)	Local address on an interface. This next-hop type causes packets with this destination address to be received locally.
Multicast (mcst)	Wire multicast next hop (limited to the LAN).
Multicast discard (mdsc)	Multicast discard.
Multicast group (mgrp)	Multicast group member.
Receive (recv)	Receive.
Reject (rjct)	Discard. An ICMP unreachable message was sent.
Resolve (rslv)	Resolving next hop.
Routed multicast (mcrt)	Regular multicast next hop.
Router	A specific node or set of nodes to which the routing device forwards packets that match the route prefix. To qualify as next-hop type router, the route must meet the following criteria: • Must not be a direct or local subnet for the routing device. • Must have a next hop that is directly connected to the routing device.
Table	Routing table next hop.

Table 84: Next-hop Types Output Field Values *(continued)*

Next-Hop Type	Description
Unicast (ucst)	Unicast.
Unilist (ulst)	List of unicast next hops. A packet sent to this next hop goes to any next hop in the list.

Table 85 on page 1428 describes all possible values for the State output field. A route can be in more than one state (for example, **<Active NoReadvrt Int Ext>**).

Table 85: State Output Field Values

Value	Description
Accounting	Route needs accounting.
Active	Route is active.
Always Compare MED	Path with a lower multiple exit discriminator (MED) is available.
AS path	Shorter AS path is available.
Cisco Non-deterministic MED selection	Cisco nondeterministic MED is enabled, and a path with a lower MED is available.
Clone	Route is a clone.
Cluster list length	Length of cluster list sent by the route reflector.
Delete	Route has been deleted.
Ex	Exterior route.
Ext	BGP route received from an external BGP neighbor.
FlashAll	Forces all protocols to be notified of a change to any route, active or inactive, for a prefix. When not set, protocols are informed of a prefix only when the active route changes.
Hidden	Route not used because of routing policy.
IfCheck	Route needs forwarding RPF check.
IGP metric	Path through next hop with lower IGP metric is available.
Inactive reason	Flags for this route, which was not selected as best for a particular destination.
Initial	Route being added.

Table 85: State Output Field Values *(continued)*

Value	Description
Int	Interior route.
Int Ext	BGP route received from an internal BGP peer or a BGP confederation peer.
Interior > Exterior > Exterior via Interior	Direct, static, IGP, or EBGP path is available.
Local Preference	Path with a higher local preference value is available.
Martian	Route is a martian (ignored because it is obviously invalid).
MartianOK	Route exempt from martian filtering.
Next hop address	Path with lower metric next hop is available.
No difference	Path from neighbor with lower IP address is available.
NoReadvrt	Route not to be advertised.
NotBest	Route not chosen because it does not have the lowest MED.
Not Best in its group	Incoming BGP AS is not the best of a group (only one AS can be the best).
NotInstall	Route not to be installed in the forwarding table.
Number of gateways	Path with a greater number of next hops is available.
Origin	Path with a lower origin code is available.
Pending	Route pending because of a hold-down configured on another route.
Programmed	Route installed programatically by on-box or off-box applications using API.
Release	Route scheduled for release.
RIB preference	Route from a higher-numbered routing table is available.
Route Distinguisher	64-bit prefix added to IP subnets to make them unique.
Route Metric or MED comparison	Route with a lower metric or MED is available.
Route Preference	Route with lower preference value is available
Router ID	Path through a neighbor with lower ID is available.

Table 85: State Output Field Values *(continued)*

Value	Description
Secondary	Route not a primary route.
Unusable path	Path is not usable because of one of the following conditions: • The route is damped. • The route is rejected by an import policy. • The route is unresolved.
Update source	Last tiebreaker is the lowest IP address value.

Table 86 on page 1430 describes the possible values for the Communities output field.

Table 86: Communities Output Field Values

Value	Description
area-number	4 bytes, encoding a 32-bit area number. For AS-external routes, the value is **0**. A nonzero value identifies the route as internal to the OSPF domain, and as within the identified area. Area numbers are relative to a particular OSPF domain.
bandwidth: ***local AS number:link-bandwidth-number***	Link-bandwidth community value used for unequal-cost load balancing. When BGP has several candidate paths available for multipath purposes, it does not perform unequal-cost load balancing according to the link-bandwidth community unless all candidate paths have this attribute.
domain-id	Unique configurable number that identifies the OSPF domain.
domain-id-vendor	Unique configurable number that further identifies the OSPF domain.
link-bandwidth-number	Link-bandwidth number: from **0** through **4,294,967,295** (bytes per second).
local AS number	Local AS number: from **1** through **65,535**.
options	1 byte. Currently this is only used if the route type is **5** or **7**. Setting the least significant bit in the field indicates that the route carries a type 2 metric.
origin	(Used with VPNs) Identifies where the route came from.
ospf-route-type	1 byte, encoded as **1** or **2** for intra-area routes (depending on whether the route came from a type 1 or a type 2 LSA); **3** for summary routes; **5** for external routes (area number must be**0**); **7** for NSSA routes; or **129** for sham link endpoint addresses.
route-type-vendor	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute **0x8000**. The format is ***area-number:ospf-route-type:options***.
rte-type	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute **0x0306**. The format is ***area-number:ospf-route-type:options***.

Table 86: Communities Output Field Values *(continued)*

Value	Description
target	Defines which VPN the route participates in; **target** has the format ***32-bit IP address:16-bit number***. For example, 10.19.0.0:100.
unknown IANA	Incoming IANA codes with a value between **0x1** and **0x7fff**. This code of the BGP extended community attribute is accepted, but it is not recognized.
unknown OSPF vendor community	Incoming IANA codes with a value above **0x8000**. This code of the BGP extended community attribute is accepted, but it is not recognized.

Sample Output

show route detail

```
user@host> show route detail

inet.0: 22 destinations, 23 routes (21 active, 0 holddown, 1 hidden)
10.10.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 29
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 1:31:43
                Task: RT
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

10.31.1.0/30 (2 entries, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 2
                Next hop: via so-0/3/0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:30:17
                Task: IF
                Announcement bits (1): 3-Resolve tree 2
                AS path: I
         OSPF   Preference: 10
                Next-hop reference count: 1
                Next hop: via so-0/3/0.0, selected
                State: <Int>
                Inactive reason: Route Preference
                Local AS:    69
                Age: 1:30:17    Metric: 1
                ORR Generation-ID: 1
      Area: 0.0.0.0
                Task: OSPF
                AS path: I

10.31.1.1/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 7
```

```
                Interface: so-0/3/0.0
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:30:20
                Task: IF
                Announcement bits (1): 3-Resolve tree 2
                AS path: I

...

10.31.2.0/30 (1 entry, 1 announced)
        *OSPF   Preference: 10
                Next-hop reference count: 9
                Next hop: via so-0/3/0.0
                Next hop: 10.31.1.6 via ge-3/1/0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:29:56    Metric: 2
                Area: 0.0.0.0
                ORR Generation-ID: 1
      Task: OSPF
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

...

172.16.233.2/32 (1 entry, 1 announced)
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:45
                Task: PIM Recv
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

...

172.16.233.22/32 (1 entry, 1 announced)
        *IGMP   Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:43
                Task: IGMP
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

10.255.70.103/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 6
                Next hop: 10.31.1.6 via ge-3/1/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 100096
                State: <Active Int>
                Local AS:    69
                Age: 1:25:49    Metric: 2
                Task: RSVP
```



```
                Announcement bits (2): 1-Resolve tree 1 2-Resolve tree 2
                AS path: I

10.255.71.238/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 6
                Next hop: via so-0/3/0.0 weight 0x1, selected
                Label-switched-path green-r1-r2
                State: <Active Int>
                Local AS:    69
                Age: 1:25:49    Metric: 1
                Task: RSVP
                Announcement bits (2): 1-Resolve tree 1 2-Resolve tree 2
                AS path: I

private__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

47.0005.80ff.f800.0000.0108.0001.0102.5507.1052/152 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:31:44
                Task: IF
                AS path: I

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
0   (1 entry, 1 announced)
        *MPLS   Preference: 0
                Next hop type: Receive
                Next-hop reference count: 6
                State: <Active Int>
                Local AS:    69
                Age: 1:31:45    Metric: 1
                Task: MPLS
                Announcement bits (1): 0-KRT
                AS path: I

...

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

  299840 (1 entry, 1 announced)
  TSI:
  KRT in-kernel 299840 /52 -> {indirect(1048575)}
          *RSVP   Preference: 7/2
                  Next hop type: Flood
                  Address: 0x9174a30
                  Next-hop reference count: 4
                  Next hop type: Router, Next hop index: 798
                  Address: 0x9174c28
                  Next-hop reference count: 2
                  Next hop: 172.16.0.2 via lt-1/2/0.9 weight 0x1
                  Label-switched-path R2-to-R4-2p2mp
                  Label operation: Pop
                  Next hop type: Router, Next hop index: 1048574
```



```
                  Address: 0x92544f0
                  Next-hop reference count: 2
                  Next hop: 172.16.0.2 via lt-1/2/0.7 weight 0x1
                  Label-switched-path R2-to-R200-p2mp
                  Label operation: Pop
                  Next hop: 172.16.0.2 via lt-1/2/0.5 weight 0x8001
                  Label operation: Pop
                  State: <Active Int>
                  Age: 1:29       Metric: 1
                  Task: RSVP
                  Announcement bits (1): 0-KRT
                  AS path: I...

800010 (1 entry, 1 announced)
        *VPLS   Preference: 7
                Next-hop reference count: 2
                Next hop: via vt-3/2/0.32769, selected
                Label operation: Pop
                State: <Active Int>
                Age: 1:29:30
                Task: Common L2 VC
                Announcement bits (1): 0-KRT
                AS path: I

vt-3/2/0.32769 (1 entry, 1 announced)
        *VPLS   Preference: 7
                Next-hop reference count: 2
                Next hop: 10.31.1.6 via ge-3/1/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 800012, Push 100096(top)
                Protocol next hop: 10.255.70.103
                Push 800012
                Indirect next hop: 87272e4 1048574
                State: <Active Int>
                Age: 1:29:30    Metric2: 2
                Task: Common L2 VC
                Announcement bits (2): 0-KRT 1-Common L2 VC
                AS path: I
                Communities: target:11111:1 Layer2-info: encaps:VPLS,
                control flags:, mtu: 0

inet6.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

abcd::10:255:71:52/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active Int>
                Local AS:    69
                Age: 1:31:44
                Task: IF
                AS path: I

fe80::280:42ff:fe10:f179/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active NoReadvrt Int>
                Local AS:    69
```



```
                Age: 1:31:44
                Task: IF
                AS path: I

ff02::2/128 (1 entry, 1 announced)
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:45
                Task: PIM Recv6
                Announcement bits (1): 0-KRT
                AS path: I

ff02::d/128 (1 entry, 1 announced)
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:45
                Task: PIM Recv6
                Announcement bits (1): 0-KRT
                AS path: I

ff02::16/128 (1 entry, 1 announced)
        *MLD    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    69
                Age: 1:31:43
                Task: MLD
                Announcement bits (1): 0-KRT
                AS path: I

private.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

fe80::280:42ff:fe10:f179/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.16385, selected
                State: <Active NoReadvrt Int>
                Age: 1:31:44
                Task: IF
                AS path: I

green.l2vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

10.255.70.103:1:3:1/96 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.255.70.103:1
                Next-hop reference count: 7
                Source: 10.255.70.103
                Protocol next hop: 10.255.70.103
                Indirect next hop: 2 no-forward
                State: <Secondary Active Int Ext>
                Local AS:    69 Peer AS:    69
                Age: 1:25:49    Metric2: 1
                AIGP 210
                Task: BGP_69.10.255.70.103+179
                Announcement bits (1): 0-green-l2vpn
```

```
                AS path: I
                Communities: target:11111:1 Layer2-info: encaps:VPLS,
                control flags:, mtu: 0
                Label-base: 800008, range: 8
                Localpref: 100
                Router ID: 10.255.70.103
                Primary Routing Table bgp.l2vpn.0

10.255.71.52:1:1:1/96 (1 entry, 1 announced)
        *L2VPN  Preference: 170/-1
                Next-hop reference count: 5
                Protocol next hop: 10.255.71.52
                Indirect next hop: 0 -
                State: <Active Int Ext>
                Age: 1:31:40    Metric2: 1
                Task: green-l2vpn
                Announcement bits (1): 1-BGP.0.0.0.0+179
                AS path: I
                Communities: Layer2-info: encaps:VPLS, control flags:Site-Down,
                mtu: 0
                Label-base: 800016, range: 8, status-vector: 0x9F

10.255.71.52:1:5:1/96 (1 entry, 1 announced)
        *L2VPN  Preference: 170/-101
                Next-hop reference count: 5
                Protocol next hop: 10.255.71.52
                Indirect next hop: 0 -
                State: <Active Int Ext>
                Age: 1:31:40    Metric2: 1
                Task: green-l2vpn
                Announcement bits (1): 1-BGP.0.0.0.0+179
                AS path: I
                Communities: Layer2-info: encaps:VPLS, control flags:, mtu: 0
                Label-base: 800008, range: 8, status-vector: 0x9F

...

l2circuit.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
10.245.255.63:CtrlWord:4:3:Local/96 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via so-1/1/2.0 weight 1, selected
                Label-switched-path my-lsp
                Label operation: Push 100000[0]
                Protocol next hop: 10.245.255.63 Indirect next hop: 86af000 296
                State: <Active Int>
                Local AS:    99
                Age: 10:21
                Task: l2 circuit
                Announcement bits (1): 0-LDP
                AS path: I
                VC Label 100000, MTU 1500, VLAN ID 512

inet.0: 45 destinations, 47 routes (44 active, 0 holddown, 1 hidden)
1.1.1.3/32 (1 entry, 1 announced)
        *IS-IS  Preference: 18
                Level: 2
                Next hop type: Router, Next hop index: 580
                Address: 0x9db6ed0
                Next-hop reference count: 8
                Next hop: 10.1.1.6 via lt-1/0/10.5, selected
                Session Id: 0x18a
```

```
                State: <Active Int>
                Local AS:     2
                Age: 1:32       Metric: 10
                Validation State: unverified
                ORR Generation-ID: 1
                Task: IS-IS
                Announcement bits (3): 0-KRT 5-Resolve tree 4 6-Resolve_IGP_FRR
task
                AS path: I

inet.0: 61 destinations, 77 routes (61 active, 1 holddown, 0 hidden)
1.1.1.1/32 (2 entries, 1 announced)
        *OSPF   Preference: 10
                Next hop type: Router, Next hop index: 673
                Address: 0xc008830
                Next-hop reference count: 3
                Next hop: 10.1.1.1 via ge-0/0/2.0, selected
                Session Id: 0x1b7
                State: <Active Int>
                Local AS:     1
                Age: 3:06:59    Metric: 100
                Validation State: unverified
                ORR Generation-ID: 1
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (2): 1-KRT 9-Resolve tree 2
                AS path: I
```

show route detail (with BGP Multipath)

```
user@host> show route detail

10.1.1.8/30 (2 entries, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 262142
                Address: 0x901a010
                Next-hop reference count: 2
                Source: 10.1.1.2
                Next hop: 10.1.1.2 via ge-0/3/0.1, selected
                Next hop: 10.1.1.6 via ge-0/3/0.5
                State: <Active Ext>
                Local AS:     1 Peer AS:     2
                Age: 5:04:43
               Validation State: unverified
                Task: BGP_2.10.1.1.2+59955
                Announcement bits (1): 0-KRT
                AS path: 2 I
                Accepted Multipath
                Localpref: 100
                Router ID: 172.16.1.2
         BGP    Preference: 170/-101
                Next hop type: Router, Next hop index: 678
                Address: 0x8f97520
                Next-hop reference count: 9
                Source: 10.1.1.6
                Next hop: 10.1.1.6 via ge-0/3/0.5, selected
                State: <NotBest Ext>
                Inactive reason: Not Best in its group - Active preferred
                Local AS:     1 Peer AS:     2
                Age: 5:04:43
             Validation State: unverified
```

```
                Task: BGP_2.10.1.1.6+58198
                AS path: 2 I
                Accepted MultipathContrib
                Localpref: 100
                Router ID: 172.16.1.3
```

show route label detail (Multipoint LDP Inband Signaling for Point-to-Multipoint LSPs)

```
user@host> show route label 299872 detail
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
299872 (1 entry, 1 announced)
        *LDP    Preference: 9
                Next hop type: Flood
                Next-hop reference count: 3
                Address: 0x9097d90
                Next hop: via vt-0/1/0.1
                Next-hop index: 661
                Label operation: Pop
                Address: 0x9172130
                Next hop: via so-0/0/3.0
                Next-hop index: 654
                Label operation: Swap 299872
                State: **Active Int>
                Local AS:  1001
                Age: 8:20       Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
               FECs bound to route: P2MP root-addr 10.255.72.166, grp 232.1.1.1,
 src 192.168.142.2
```

show route label detail (Multipoint LDP with Multicast-Only Fast Reroute)

```
user@host> show route label 301568 detail

mpls.0: 18 destinations, 18 routes (18 active, 0 holddown, 0 hidden)
301568 (1 entry, 1 announced)
        *LDP    Preference: 9
                Next hop type: Flood
                Address: 0x2735208
                Next-hop reference count: 3
                Next hop type: Router, Next hop index: 1397
                Address: 0x2735d2c
                Next-hop reference count: 3
                Next hop: 1.3.8.2 via ge-1/2/22.0
                Label operation: Pop
                Load balance label:  None;
                Next hop type: Router, Next hop index: 1395
                Address: 0x2736290
                Next-hop reference count: 3
                Next hop: 1.3.4.2 via ge-1/2/18.0
                Label operation: Pop
                Load balance label:  None;
                State: <Active Int AckRequest MulticastRPF>
                Local AS:    10
                Age: 54:05      Metric: 1
                Validation State: unverified
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
```



```
                FECs bound to route: P2MP root-addr 172.16.1.1, grp: 232.1.1.1,
src: 192.168.219.11
                Primary Upstream : 172.16.1.3:0--172.16.1.2:0
                   RPF Nexthops  :
                        ge-1/2/15.0, 1.2.94.1, Label: 301568, weight: 0x1
                        ge-1/2/14.0, 1.2.3.1, Label: 301568, weight: 0x1
                Backup Upstream : 172.16.1.3:0--172.16.1.6:0
                   RPF Nexthops :
                        ge-1/2/20.0, 1.2.96.1, Label: 301584, weight: 0xfffe
                        ge-1/2/19.0, 1.3.6.1, Label: 301584, weight: 0xfffe
```


show route exact

List of Syntax

Syntax on page 1440
Syntax (EX Series Switches) on page 1440

Syntax

show route exact *destination-prefix*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route exact *destination-prefix*
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display only the routes that exactly match the specified address or range of addresses.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

destination-prefix—Address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show route exact on page 1440
show route exact detail on page 1441
show route exact extensive on page 1441
show route exact terse on page 1441

Output Fields

For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route exact

```
user@host> show route exact 207.17.136.0/24

inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
207.17.136.0/24    *[Static/5] 2d 03:30:22
                    > to 192.168.71.254 via fxp0.0
```


show route exact detail

```
user@host> show route exact 207.17.136.0/24 detail

inet.0: 24 destinations, 25 routes (23 active, 0 holddown, 1 hidden)
Restart Complete
207.17.136.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 29
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2d 3:30:26
                Task: RT
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I
```

show route exact extensive

```
user@host> show route exact 207.17.136.0/24 extensive
inet.0: 22 destinations, 23 routes (21 active, 0 holddown, 1 hidden)
207.17.136.0/24 (1 entry, 1 announced)
TSI:
KRT in-kernel 207.17.136.0/24 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 29
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 1:25:18
                Task: RT
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I
```

show route exact terse

```
user@host> show route exact 207.17.136.0/24 terse

inet.0: 22 destinations, 23 routes (21 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both
A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 207.17.136.0/24    S   5                        >192.168.71.254
```

show route export

List of Syntax

Syntax on page 1442
Syntax (EX Series Switches) on page 1442

Syntax

show route export
<brief | detail>
<instance <*instance-name*> | *routing-table-name*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route export
<brief | detail>
<instance <*instance-name*> | *routing-table-name*>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display policy-based route export information. Policy-based export simplifies the process of exchanging route information between routing instances.

Options

none—(Same as **brief**.) Display standard information about policy-based export for all instances and routing tables on all systems.

brief | detail—(Optional) Display the specified level of output.

instance <*instance-name*>—(Optional) Display a particular routing instance for which policy-based export is currently enabled.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

routing-table-name—(Optional) Display information about policy-based export for all routing tables whose name begins with this string (for example, inet.0 and inet6.0 are both displayed when you run the **show route export inet** command).

Required Privilege Level

view

List of Sample Output

show route export on page 1443
show route export detail on page 1443
show route export instance detail on page 1444

Output Fields

Table 87 on page 1443 lists the output fields for the **show route export** command. Output fields are listed in the approximate order in which they appear.

Table 87: show route export Output Fields

Field Name	Field Description	Level of Output
Table or *table-name*	Name of the routing tables that either import or export routes.	All levels
Routes	Number of routes exported from this table into other tables. If a particular route is exported to different tables, the counter will only increment by one.	**brief** none
Export	Whether the table is currently exporting routes to other tables: **Y** or **N** (Yes or No).	**brief** none
Import	Tables currently importing routes from the originator table. (Not displayed for tables that are not exporting any routes.)	**detail**
Flags	(**instance** keyword only) Flags for this feature on this instance: • **config auto-policy**—The policy was deduced from the configured IGP export policies. • **cleanup**—Configuration information for this instance is no longer valid. • **config**—The instance was explicitly configured.	**detail**
Options	(**instance** keyword only) Configured option displays the type of routing tables the feature handles: • **unicast**—Indicates *instance*.inet.0. • **multicast**—Indicates *instance*.inet.2. • **unicast multicast**—Indicates *instance*.inet.0 and *instance*.inet.2.	**detail**
Import policy	(**instance** keyword only) Policy that **route export** uses to construct the import-export matrix. Not displayed if the instance type is **vrf**.	**detail**
Instance	(**instance** keyword only) Name of the routing instance.	**detail**
Type	(**instance** keyword only) Type of routing instance: **forwarding**, **non-forwarding**, or **vrf**.	**detail**

Sample Output

show route export

```
user@host> show route export
Table                             Export          Routes
inet.0                            N                    0
black.inet.0                      Y                    3
red.inet.0                        Y                    4
```

show route export detail

```
user@host>  show route export detail
inet.0                                          Routes:        0
black.inet.0                                    Routes:        3
  Import: [ inet.0 ]
red.inet.0                                      Routes:        4
  Import: [ inet.0 ]
```


show route export instance detail

```
user@host> show route export instance detail
Instance: master                        Type: forwarding
  Flags: <config auto-policy> Options: <unicast multicast>
  Import policy: [ (ospf-master-from-red || isis-master-from-black) ]
Instance: black                         Type: non-forwarding
Instance: red                           Type: non-forwarding
```


show route extensive

List of Syntax

Syntax on page 1445
Syntax (EX Series Switches) on page 1445

Syntax

show route extensive
<*destination-prefix*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route extensive
<*destination-prefix*>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display extensive information about the active entries in the routing tables.

Options

none—Display all active entries in the routing table.

destination-prefix—(Optional) Display active entries for the specified address or range of addresses.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show route extensive on page 1452
show route extensive (Access Route) on page 1459
show route extensive (BGP PIC Edge) on page 1459
show route extensive (FRR and LFA) on page 1460
show route extensive (IS-IS) on page 1461
show route extensive (Route Reflector) on page 1461
show route label detail (Multipoint LDP Inband Signaling for Point-to-Multipoint LSPs) on page 1461
show route label detail (Multipoint LDP with Multicast-Only Fast Reroute) on page 1462

Output Fields

Table 88 on page 1445 describes the output fields for the **show route extensive** command. Output fields are listed in the approximate order in which they appear.

Table 88: show route extensive Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).
number **destinations**	Number of destinations for which there are routes in the routing table.

Table 88: show route extensive Output Fields *(continued)*

<table>
<tr><th>Field Name</th><th>Field Description</th></tr>
<tr><td>number routes</td><td>Number of routes in the routing table and total number of routes in the following states:
active (routes that are active).
holddown (routes that are in the pending state before being declared inactive).
hidden (routes that are not used because of a routing policy).</td></tr>
<tr><td>route-destination (entry, announced)</td><td>Route destination (for example: 10.0.0.1/24). The entry value is the number of route for this destination, and the announced value is the number of routes being announced for this destination. Sometimes the route destination is presented in another format, such as:
MPLS-label (for example, 80001).
interface-name (for example, ge-1/0/2).
neighbor-address:control-word-status:encapsulation type:vc-id:source (Layer 2 circuit only; for example, 10.1.1.195:NoCtrlWord:1:1:Local/96).
neighbor-address—Address of the neighbor.
control-word-status—Whether the use of the control word has been negotiated for this virtual circuit: NoCtrlWord or CtrlWord.
encapsulation type—Type of encapsulation, represented by a number: (1) Frame Relay DLCI, (2) ATM AAL5 VCC transport, (3) ATM transparent cell transport, (4) Ethernet, (5) VLAN Ethernet, (6) HDLC, (7) PPP, (8) ATM VCC cell transport, (10) ATM VPC cell transport.
vc-id—Virtual circuit identifier.
source—Source of the advertisement: Local or Remote.</td></tr>
<tr><td>TSI</td><td>Protocol header information.</td></tr>
<tr><td>label stacking</td><td>(Next-to-the-last-hop routing device for MPLS only) Depth of the MPLS label stack, where the label-popping operation is needed to remove one or more labels from the top of the stack. A pair of routes is displayed, because the pop operation is performed only when the stack depth is two or more labels.
S=0 route indicates that a packet with an incoming label stack depth of two or more exits this router with one fewer label (the label-popping operation is performed).
If there is no S= information, the route is a normal MPLS route, which has a stack depth of 1 (the label-popping operation is not performed).</td></tr>
<tr><td>[protocol, preference]</td><td>Protocol from which the route was learned and the preference value for the route.
+—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table.
- —A hyphen indicates the last active route.
*—An asterisk indicates that the route is both the active and the last active route. An asterisk before a to line indicates the best subpath to the route.
In every routing metric except for the BGP LocalPref attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the LocalPref value in the Preference2 field. For example, if the LocalPref value for Route 1 is 100, the Preference2 value is -101. If the LocalPref value for Route 2 is 155, the Preference2 value is -156. Route 2 is preferred because it has a higher LocalPref value and a lower Preference2 value.</td></tr>
</table>

Table 88: show route extensive Output Fields *(continued)*

Field Name	Field Description
Level	(IS-IS only). In IS-IS, a single autonomous system (AS) can be divided into smaller groups called areas. Routing between areas is organized hierarchically, allowing a domain to be administratively divided into smaller areas. This organization is accomplished by configuring Level 1 and Level 2 intermediate systems. Level 1 systems route within an area. When the destination is outside an area, they route toward a Level 2 system. Level 2 intermediate systems route between areas and toward other ASs.
Route Distinguisher	IP subnet augmented with a 64-bit prefix.
PMSI	Provider multicast service interface (MVPN routing table).
Next-hop type	Type of next hop. For a description of possible values for this field, see the Output Field table in the show route detail command.
Next-hop reference count	Number of references made to the next hop.
Flood nexthop branches exceed maximum message	Indicates that the number of flood next-hop branches exceeded the system limit of 32 branches, and only a subset of the flood next-hop branches were installed in the kernel.
Source	IP address of the route source.
Next hop	Network layer address of the directly reachable neighboring system.
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the name of the interface that is actually used is followed by the word **Selected**. This field can also contain the following information: • Weight—Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible. • Balance—Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a routing device is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.
Label-switched-path ***lsp-path-name***	Name of the LSP used to reach the next hop.
Label operation	MPLS label and operation occurring at this routing device. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label).
Offset	Whether the metric has been increased or decreased by an offset value.
Interface	(Local only) Local interface name.
Protocol next hop	Network layer address of the remote routing device that advertised the prefix. This address is used to recursively derive a forwarding next hop.

Table 88: show route extensive Output Fields *(continued)*

Field Name	Field Description
label-operation	MPLS label and operation occurring at this routing device. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label).
Indirect next hops	When present, a list of nodes that are used to resolve the path to the next-hop destination, in the order that they are resolved. When BGP PIC Edge is enabled, the output lines that contain **Indirect next hop: weight** follow next hops that the software can use to repair paths where a link failure occurs. The next-hop weight has one of the following values: • 0x1 indicates active next hops. • 0x4000 indicates passive next hops.
State	State of the route (a route can be in more than one state). See the Output Field table in the show route detail command.
Session ID	The BFD session ID number that represents the protection using MPLS fast reroute (FRR) and loop-free alternate (LFA).
Weight	Weight for the backup path. If the weight of an indirect next hop is larger than zero, the weight value is shown. For sample output, see show route table.

Table 88: show route extensive Output Fields *(continued)*

Field Name	Field Description
Inactive reason	If the route is inactive, the reason for its current state is indicated. Typical reasons include: • **Active preferred**—Currently active route was selected over this route. • **Always compare MED**—Path with a lower multiple exit discriminator (MED) is available. • **AS path**—Shorter AS path is available. • **Cisco Non-deterministic MED selection**—Cisco nondeterministic MED is enabled and a path with a lower MED is available. • **Cluster list length**—Path with a shorter cluster list length is available. • **Forwarding use only**—Path is only available for forwarding purposes. • **IGP metric**—Path through the next hop with a lower IGP metric is available. • **IGP metric type**—Path with a lower OSPF link-state advertisement type is available. • **Interior > Exterior > Exterior via Interior**—Direct, static, IGP, or EBGP path is available. • **Local preference**—Path with a higher local preference value is available. • **Next hop address**—Path with a lower metric next hop is available. • **No difference**—Path from a neighbor with a lower IP address is available. • **Not Best in its group**—Occurs when multiple peers of the same external AS advertise the same prefix and are grouped together in the selection process. When this reason is displayed, an additional reason is provided (typically one of the other reasons listed). • **Number of gateways**—Path with a higher number of next hops is available. • **Origin**—Path with a lower origin code is available. • **OSPF version**—Path does not support the indicated OSPF version. • **RIB preference**—Route from a higher-numbered routing table is available. • **Route destinguisher**—64-bit prefix added to IP subnets to make them unique. • **Route metric or MED comparison**—Route with a lower metric or MED is available. • **Route preference**—Route with a lower preference value is available. • **Router ID**—Path through a neighbor with a lower ID is available. • **Unusable path**—Path is not usable because of one of the following conditions: the route is damped, the route is rejected by an import policy, or the route is unresolved. • **Update source**—Last tiebreaker is the lowest IP address value.
Local AS	Autonomous system (AS) number of the local routing device.
Age	How long the route has been known.
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.
Metric	Cost value of the indicated route. For routes within an AS, the cost is determined by IGP and the individual protocol metrics. For external routes, destinations, or routing domains, the cost is determined by a preference value.
MED-plus-IGP	Metric value for BGP path selection to which the IGP cost to the next-hop destination has been added.
TTL-Action	For MPLS LSPs, state of the TTL propagation attribute. Can be enabled or disabled for all RSVP-signaled and LDP-signaled LSPs or for specific VRF routing instances. For sample output, see show route table.

Table 88: show route extensive Output Fields *(continued)*

Field Name	Field Description
Task	Name of the protocol that has added the route.
Announcement bits	List of protocols that are consumers of the route. Using the following output as an example, **Announcement bits (3): 0-KRT 5-Resolve tree 2 8-BGP RT Background** there are (3) announcement bits to reflect the three clients (protocols) that have state for this route: Kernel (0-KRT), 5 (resolution tree process 2), and 8 (BGP). The notation ***n*-Resolve inet** indicates that the route is used for route resolution for next hops found in the routing table. ***n*** is an index used by Juniper Networks customer support only.
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **Recorded**—The AS path is recorded by the sample process (sampled). • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the local AS number associated with the AS path if more than one AS number is configured on the routing device, or if AS path prepending is configured. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.
validation-state	(BGP-learned routes) Validation status of the route: • **Invalid**—Indicates that the prefix is found, but either the corresponding AS received from the EBGP peer is not the AS that appears in the database, or the prefix length in the BGP update message is longer than the maximum length permitted in the database. • **Unknown**—Indicates that the prefix is not among the prefixes or prefix ranges in the database. • **Unverified**—Indicates that origin validation is not enabled for the BGP peers. • **Valid**—Indicates that the prefix and autonomous system pair are found in the database.
FECs bound to route	Point-to-multipoint root address, multicast source address, and multicast group address when multipoint LDP (M-LDP) inband signaling is configured.
AS path: I <Originator>	(For route reflected output only) Originator ID attribute set by the route reflector.

Table 88: show route extensive Output Fields *(continued)*

Field Name	Field Description
route status	Indicates the status of a BGP route: • **Accepted**—The specified BGP route is imported by the default BGP policy. • **Import**—The route is imported into a Layer 3 VPN routing instance. • **Import-Protect**—A remote instance egress that is protected. • **Multipath**—A BGP multipath active route. • **MultipathContrib**—The route is not active but contributes to the BGP multipath. • **Protect**—An egress route that is protected. • **Stale**—A route that is marked stale due to graceful restart.
Primary Upstream	When multipoint LDP with multicast-only fast reroute (MoFRR) is configured, the primary upstream path. MoFRR transmits a multicast join message from a receiver toward a source on a primary path, while also transmitting a secondary multicast join message from the receiver toward the source on a backup path.
RPF Nexthops	When multipoint LDP with MoFRR is configured, the reverse-path forwarding (RPF) next-hop information. Data packets are received from both the primary path and the secondary paths. The redundant packets are discarded at topology merge points due to the RPF checks.
Label	Multiple MPLS labels are used to control MoFRR stream selection. Each label represents a separate route, but each references the same interface list check. Only the primary label is forwarded while all others are dropped. Multiple interfaces can receive packets using the same label.
weight	Value used to distinguish MoFRR primary and backup routes. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible.
VC Label	MPLS label assigned to the Layer 2 circuit virtual connection.
MTU	Maximum transmission unit (MTU) of the Layer 2 circuit.
VLAN ID	VLAN identifier of the Layer 2 circuit.
Cluster list	(For route reflected output only) Cluster ID sent by the route reflector.
Originator ID	(For route reflected output only) Address of router that originally sent the route to the route reflector.
Prefixes bound to route	Forwarding Equivalent Class (FEC) bound to this route. Applicable only to routes installed by LDP.
Communities	Community path attribute for the route. See the Output Field table in the show route detail command for all possible values for this field.
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).
control flags	Control flags: **none** or Site Down.
mtu	Maximum transmission unit (MTU) information.
Label-Base, range	First label in a block of labels and label block size. A remote PE routing device uses this first label when sending traffic toward the advertising PE routing device.

Table 88: show route extensive Output Fields *(continued)*

Field Name	Field Description
status vector	Layer 2 VPN and VPLS network layer reachability information (NLRI).
Localpref	Local preference value included in the route.
Router ID	BGP router ID as advertised by the neighbor in the open message.
Primary Routing Table	In a routing table group, the name of the primary routing table in which the route resides.
Secondary Tables	In a routing table group, the name of one or more secondary tables in which the route resides.
Originating RIB	Name of the routing table whose active route was used to determine the forwarding next-hop entry in the resolution database. For example, in the case of inet.0 resolving through inet.0 and inet.3, this field indicates which routing table, inet.0 or inet.3, provided the best path for a particular prefix.
Node path count	Number of nodes in the path.
Forwarding nexthops	Number of forwarding next hops. The forwarding next hop is the network layer address of the directly reachable neighboring system (if applicable) and the interface used to reach it.

Sample Output

show route extensive

```
user@host> show route extensive
inet.0: 22 destinations, 23 routes (21 active, 0 holddown, 1 hidden)
203.0.113.10/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 203.0.113.10/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 29
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    64496
                Age: 1:34:06
                Task: RT
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

203.0.113.30/30 (2 entries, 1 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 2
                Next hop: via so-0/3/0.0, selected
                State: <Active Int>
                Local AS:    64496
                Age: 1:32:40
                Task: IF
                Announcement bits (1): 3-Resolve tree 2
                AS path: I
         OSPF   Preference: 10
                Next-hop reference count: 1
                Next hop: via so-0/3/0.0, selected
```

```
                State: <Int>
                Inactive reason: Route Preference
                Local AS:    64496
                Age: 1:32:40    Metric: 1
                Area: 0.0.0.0
                Task: OSPF
                AS path: I

203.0.113.103/32 (1 entry, 1 announced)
        *Local  Preference: 0
                Next hop type: Local
                Next-hop reference count: 7
                Interface: so-0/3/0.0
                State: <Active NoReadvrt Int>
                Local AS:    644969
                Age: 1:32:43
                Task: IF
                Announcement bits (1): 3-Resolve tree 2
                AS path: I

...

203.0.113.203/30 (1 entry, 1 announced)
TSI:
KRT in-kernel 203.0.113.203/30 -> {203.0.113.216}
        *OSPF   Preference: 10
                Next-hop reference count: 9
                Next hop: via so-0/3/0.0
                Next hop: 203.0.113.216 via ge-3/1/0.0, selected
                State: <Active Int>
                Local AS:    64496
                Age: 1:32:19    Metric: 2
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

...

198.51.100.2/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 198.51.100.2/32 -> {}
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    64496
                Age: 1:34:08
                Task: PIM Recv
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

...

198.51.100.22/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 198.51.100.22/32 -> {}
        *IGMP   Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    64496
                Age: 1:34:06
```



```
                Task: IGMP
                Announcement bits (2): 0-KRT 3-Resolve tree 2
                AS path: I

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

203.0.113.103/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 6
                Next hop: 203.0.113.216 via ge-3/1/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 100096
                State: <Active Int>
                Local AS:    64496
                Age: 1:28:12    Metric: 2
                Task: RSVP
                Announcement bits (2): 1-Resolve tree 1 2-Resolve tree 2
                AS path: I

203.0.113.238/32 (1 entry, 1 announced)
        State: <FlashAll>
        *RSVP   Preference: 7
                Next-hop reference count: 6
                Next hop: via so-0/3/0.0 weight 0x1, selected
                Label-switched-path green-r1-r2
                State: <Active Int>
                Local AS:    64496
                Age: 1:28:12    Metric: 1
                Task: RSVP
                Announcement bits (2): 1-Resolve tree 1 2-Resolve tree 2
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

...

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

47.0005.80ff.f800.0000.0108.0001.0102.5507.1052/152 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active Int>
                Local AS:    64496
                Age: 1:34:07
                Task: IF
                AS path: I

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

0   (1 entry, 1 announced)
TSI:
KRT in-kernel 0      /36 -> {}
        *MPLS   Preference: 0
                Next hop type: Receive
                Next-hop reference count: 6
                State: <Active Int>
                Local AS:    64496
                Age: 1:34:08    Metric: 1
```



```
                Task: MPLS
                Announcement bits (1): 0-KRT
                AS path: I

...

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
  299840 (1 entry, 1 announced)
  TSI:
  KRT in-kernel 299840 /52 -> {indirect(1048575)}
          *RSVP   Preference: 7/2
                  Next hop type: Flood
                  Address: 0x9174a30
                  Next-hop reference count: 4
                  Next hop type: Router, Next hop index: 798
                  Address: 0x9174c28
                  Next-hop reference count: 2
                  Next hop: 198.51.100.2 via lt-1/2/0.9 weight 0x1
                  Label-switched-path R2-to-R4-2p2mp
                  Label operation: Pop
                  Next hop type: Router, Next hop index: 1048574
                  Address: 0x92544f0
                  Next-hop reference count: 2
                  Next hop: 198.51.100.2 via lt-1/2/0.7 weight 0x1
                  Label-switched-path R2-to-R200-p2mp
                  Label operation: Pop
                  Next hop: 198.51.100.2 via lt-1/2/0.5 weight 0x8001
                  Label operation: Pop
                  State: <Active Int>
                  Age: 1:29       Metric: 1
                  Task: RSVP
                  Announcement bits (1): 0-KRT
                  AS path: I...

800010 (1 entry, 1 announced)

TSI:
KRT in-kernel 800010 /36 -> {vt-3/2/0.32769}
        *VPLS   Preference: 7
                Next-hop reference count: 2
                Next hop: via vt-3/2/0.32769, selected
                Label operation: Pop
                State: <Active Int>
                Age: 1:31:53
                Task: Common L2 VC
                Announcement bits (1): 0-KRT
                AS path: I

vt-3/2/0.32769 (1 entry, 1 announced)
TSI:
KRT in-kernel vt-3/2/0.32769.0      /16 -> {indirect(1048574)}
        *VPLS   Preference: 7
                Next-hop reference count: 2
                Next hop: 203.0.113.216 via ge-3/1/0.0 weight 0x1, selected
                Label-switched-path green-r1-r3
                Label operation: Push 800012, Push 100096(top)
                Protocol next hop: 203.0.113.103
                Push 800012
                Indirect next hop: 87272e4 1048574
                State: <Active Int>
                Age: 1:31:53    Metric2: 2
```

```
                Task: Common L2 VC
                Announcement bits (2): 0-KRT 1-Common L2 VC
                AS path: I
                Communities: target:11111:1 Layer2-info: encaps:VPLS,
                control flags:, mtu: 0
                Indirect next hops: 1
                        Protocol next hop: 203.0.113.103 Metric: 2
                        Push 800012
                        Indirect next hop: 87272e4 1048574
                        Indirect path forwarding next hops: 1
                               Next hop: 203.0.113.216 via ge-3/1/0.0 weight 0x1

                        203.0.113.103/32 Originating RIB: inet.3
                          Metric: 2                      Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 203.0.113.216 via ge-3/1/0.0

inet6.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

2001:db8::10:255:71:52/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active Int>
                Local AS:    64496
                Age: 1:34:07
                Task: IF
                AS path: I

fe80::280:42ff:fe10:f179/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Active NoReadvrt Int>
                Local AS:    64496
                Age: 1:34:07
                Task: IF
                AS path: I

ff02::2/128 (1 entry, 1 announced)
TSI:
KRT in-kernel ff02::2/128 -> {}
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    64496
                Age: 1:34:08
                Task: PIM Recv6
                Announcement bits (1): 0-KRT
                AS path: I

ff02::d/128 (1 entry, 1 announced)
TSI:
KRT in-kernel ff02::d/128 -> {}
        *PIM    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    64496
                Age: 1:34:08
```



```
                Task: PIM Recv6
                Announcement bits (1): 0-KRT
                AS path: I

ff02::16/128 (1 entry, 1 announced)
TSI:
KRT in-kernel ff02::16/128 -> {}
        *MLD    Preference: 0
                Next-hop reference count: 18
                State: <Active NoReadvrt Int>
                Local AS:    64496
                Age: 1:34:06
                Task: MLD
                Announcement bits (1): 0-KRT
                AS path: I

private.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

fe80::280:42ff:fe10:f179/128 (1 entry, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.16385, selected
                State: <Active NoReadvrt Int>
                Age: 1:34:07
                Task: IF
                AS path: I

green.l2vpn.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)

203.0.113.103:1:3:1/96 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 203.0.113.103:1
                Next-hop reference count: 7
                Source: 203.0.113.103
                Protocol next hop: 203.0.113.103
                Indirect next hop: 2 no-forward
                State: <Secondary Active Int Ext>
                Local AS:    64496 Peer AS:    64496
                Age: 1:28:12    Metric2: 1
                Task: BGP_69.203.0.113.103+179
                Announcement bits (1): 0-green-l2vpn
                AS path: I
                Communities: target:11111:1 Layer2-info: encaps:VPLS,
                control flags:, mtu: 0
                Label-base: 800008, range: 8
                Localpref: 100
                Router ID: 203.0.113.103
                Primary Routing Table bgp.l2vpn.0

203.0.113.152:1:1:1/96 (1 entry, 1 announced)
TSI:
Page 0 idx 0 Type 1 val 8699540
        *L2VPN  Preference: 170/-1
                Next-hop reference count: 5
                Protocol next hop: 203.0.113.152
                Indirect next hop: 0 -
                State: <Active Int Ext>
                Age: 1:34:03    Metric2: 1
                Task: green-l2vpn
                Announcement bits (1): 1-BGP.0.0.0.0+179
```

```
                AS path: I
                Communities: Layer2-info: encaps:VPLS, control flags:Site-Down,
                mtu: 0
                Label-base: 800016, range: 8, status-vector: 0x9F

203.0.113.152:1:5:1/96 (1 entry, 1 announced)
TSI:
Page 0 idx 0 Type 1 val 8699528
        *L2VPN  Preference: 170/-101
                Next-hop reference count: 5
                Protocol next hop: 203.0.113.152
                Indirect next hop: 0 -
                State: <Active Int Ext>
                Age: 1:34:03    Metric2: 1
                Task: green-l2vpn
                Announcement bits (1): 1-BGP.0.0.0.0+179
                AS path: I
                Communities: Layer2-info: encaps:VPLS, control flags:, mtu: 0
                Label-base: 800008, range: 8, status-vector: 0x9F

...

l2circuit.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

TSI:

203.0.113.163:CtrlWord:4:3:Local/96 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via so-1/1/2.0 weight 1, selected
                Label-switched-path my-lsp
                Label operation: Push 100000[0]
                Protocol next hop: 203.0.113.163 Indirect next hop: 86af000 296
                State: <Active Int>
                Local AS:    64499
                Age: 10:21
                Task: l2 circuit
                Announcement bits (1): 0-LDP
                AS path: I
                VC Label 100000, MTU 1500, VLAN ID 512


203.0.113.55/24 (1 entry, 1 announced)
TSI:
KRT queued (pending) add
  198.51.100.0/24 -> {Push 300112}
        *BGP    Preference: 170/-101
                Next hop type: Router
                Address: 0x925c208
                Next-hop reference count: 2
                Source: 203.0.113.9
                Next hop: 203.0.113.9 via ge-1/2/0.15, selected
                Label operation: Push 300112
                Label TTL action: prop-ttl
                State: <Active Ext>
                Local AS:  64509 Peer AS: 65539
                Age: 1w0d 23:06:56
                AIGP: 25
                Task: BGP_65539.203.0.113.9+56732
                Announcement bits (1): 0-KRT
                AS path: 65539 64508 I
                Accepted
```



```
                Route Label: 300112
                Localpref: 100
                Router ID: 213.0.113.99
```

show route extensive (Access Route)

```
user@host> show route 203.0.113.102 extensive
inet.0: 39256 destinations, 39258 routes (39255 active, 0 holddown, 1 hidden)
203.0.113.102/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 203.0.113.102/32 -> {192.0.2.2}
OSPF area : 0.0.0.0, LSA ID : 203.0.113.102, LSA type : Extern
        *Access Preference: 13
                Next-hop reference count: 78472
                Next hop: 192.0.2.2 via fe-0/0/0.0, selected
                State: <Active Int>
       Age: 12
                Task: RPD Unix Domain Server./var/run/rpd_serv.local
                Announcement bits (2): 0-KRT 1-OSPFv2
                AS path: I


user@host> show route 2001:db8:4641:1::/48 extensive

inet6.0: 75 destinations, 81 routes (75 active, 0 holddown, 0 hidden)
  2001:db8:4641:1::/48 (1 entry, 1 announced)
  TSI:
  KRT in-kernel 2001:db8:4641:1::/48 -> {#0 0.13.1.0.0.1}
          *Access Preference: 13
                  Next hop type: Router, Next hop index: 74548
                  Address: 0x1638c1d8
                  Next-hop reference count: 6
                  Next hop: #0 0.13.1.0.0.1 via demux0.1073753267, selected
                  Session Id: 0x0
                  State: <Active Int>
                  Age: 4:17
                  Validation State: unverified
                  Task: RPD Unix Domain Server./var/run/rpd_serv.local
                  Announcement bits (2): 0-KRT 4-Resolve tree 2
                  AS path: I
2001:db8:4641:1::/128 (1 entry, 1 announced)
  TSI:
  KRT in-kernel 2001:db8:4641:1::/128 -> {#0 0.13.1.0.0.1}
          *Access-internal Preference: 12
                  Next hop type: Router, Next hop index: 74548
                  Address: 0x1638c1d8
                  Next-hop reference count: 6
                  Next hop: #0 0.13.1.0.0.1 via demux0.1073753267, selected
                  Session Id: 0x0
                  State: <Active Int>
                  Age: 4:17
                  Validation State: unverified
                  Task: RPD Unix Domain Server./var/run/rpd_serv.local
                  Announcement bits (2): 0-KRT 4-Resolve tree 2
                  AS path: I
```

show route extensive (BGP PIC Edge)

```
user@host> show route 198.51.100.6 extensive
ed.inet.0: 6 destinations, 9 routes (6 active, 0 holddown, 0 hidden)
  198.51.100.6/32 (3 entries, 2 announced)
```

```
          State: <CalcForwarding>
  TSI:
  KRT in-kernel 198.51.100.6/32 -> {indirect(1048574), indirect(1048577)}
  Page 0 idx 0 Type 1 val 9219e30
      Nexthop: Self
      AS path: [2] 3 I
      Communities: target:2:1
  Path 198.51.100.6 from 198.51.100.4 Vector len 4.  Val: 0
..
          #Multipath Preference: 255
                  Next hop type: Indirect
                  Address: 0x93f4010
                  Next-hop reference count: 2
..
                  Protocol next hop: 198.51.1001.4
                  Push 299824
                  Indirect next hop: 944c000 1048574 INH Session ID: 0x3
                  Indirect next hop: weight 0x1
                  Protocol next hop: 198.51.100.5
                  Push 299824
                  Indirect next hop: 944c1d8 1048577 INH Session ID: 0x4
                  Indirect next hop: weight 0x4000
                  State: <ForwardingOnly Int Ext)>
                  Inactive reason: Forwarding use only
                  Age: 25         Metric2: 15
                  Validation State: unverified
                  Task: RT
                  Announcement bits (1): 0-KRT
                  AS path: 3 I
                  Communities: target:2:1
```

show route extensive (FRR and LFA)

```
user@host> show route 203.0.113.20 extensive
inet.0: 46 destinations, 49 routes (45 active, 0 holddown, 1 hidden)
203.0.113.20/24 (2 entries, 1 announced)
        State: FlashAll
TSI:
KRT in-kernel 203.0.113.20/24 -> {Push 299776, Push 299792}
        *RSVP   Preference: 7/1
                Next hop type: Router, Next hop index: 1048574
                Address: 0xbbbc010
                Next-hop reference count: 5
                Next hop: 203.0.113.112 via ge-2/1/8.0 weight 0x1, selected
                Label-switched-path europa-d-to-europa-e
                Label operation: Push 299776
                Label TTL action: prop-ttl
                Session Id: 0x201
                Next hop: 203.0.113.122 via ge-2/1/4.0 weight 0x4001
                Label-switched-path europa-d-to-europa-e
                Label operation: Push 299792
                Label TTL action: prop-ttl
                Session Id: 0x202
                State: Active Int
                Local AS:   64500
                Age: 5:31  Metric: 2
                Task: RSVP
                Announcement bits (1): 0-KRT
                AS path: I
         OSPF   Preference: 10
                Next hop type: Router, Next hop index: 615
```



```
                Address: 0xb9d78c4
                Next-hop reference count: 7
                Next hop: 203.0.113.112 via ge-2/1/8.0, selected
                Session Id: 0x201
                State: Int
                Inactive reason: Route Preference
                Local AS:   64500
                Age: 5:35  Metric: 3
                Area: 0.0.0.0
                Task: OSPF
                AS path: I
```

show route extensive (IS-IS)

```
user@host> show route extensive
IS-IS Preference: 15
                Level: 1
                Next hop type: Router, Next hop index: 1048577
                Address: 0xXXXXXXXXXX
                Next-hop reference count: YY
                Next hop: 203.0.113.22 via ae1.0 balance 43%, selected
                Session Id: 0x141
                Next hop: 203.0.113.22 via ae0.0 balance 57%
```

show route extensive (Route Reflector)

```
user@host> show route extensive
203.0.113.0/8 (1 entry, 1 announced)

TSI:
KRT in-kernel 203.0.113.0/8 -> {indirect(40)}
        *BGP    Preference: 170/-101
                Source: 192.168.4.214
                Protocol next hop: 198.51.100.192 Indirect next hop: 84ac908 40
                State: <Active Int Ext>
                Local AS: 65548 Peer AS: 65548
                Age: 3:09       Metric: 0       Metric2: 0
                Task: BGP_65548.192.168.4.214+1033
                Announcement bits (2): 0-KRT 4-Resolve inet.0
                AS path: 65544 64507 I <Originator>
                Cluster list:  198.51.100.1
                Originator ID: 203.0.113.88
                Communities: 7777:7777
                Localpref: 100
                Router ID: 203.0.113.4
                Indirect next hops: 1
                        Protocol next hop: 203.0.113.192 Metric: 0
                        Indirect next hop: 84ac908 40
                        Indirect path forwarding next hops: 0
                                Next hop type: Discard
```

show route label detail (Multipoint LDP Inband Signaling for Point-to-Multipoint LSPs)

```
user@host> show route label 299872 detail
mpls.0: 13 destinations, 13 routes (13 active, 0 holddown, 0 hidden)
299872 (1 entry, 1 announced)
        *LDP    Preference: 9
                Next hop type: Flood
                Next-hop reference count: 3
                Address: 0x9097d90
```

```
                Next hop: via vt-0/1/0.1
                Next-hop index: 661
                Label operation: Pop
                Address: 0x9172130
                Next hop: via so-0/0/3.0
                Next-hop index: 654
                Label operation: Swap 299872
                State: **Active Int>
                Local AS:  64511
                Age: 8:20       Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
               FECs bound to route: P2MP root-addr 203.0.113.166, grp 203.0.1.1,
 src 192.168.142.2
```

show route label detail (Multipoint LDP with Multicast-Only Fast Reroute)

user@host> **show route label 301568 detail**

```
mpls.0: 18 destinations, 18 routes (18 active, 0 holddown, 0 hidden)
301568 (1 entry, 1 announced)
        *LDP    Preference: 9
                Next hop type: Flood
                Address: 0x2735208
                Next-hop reference count: 3
                Next hop type: Router, Next hop index: 1397
                Address: 0x2735d2c
                Next-hop reference count: 3
                Next hop: 203.0.113.82 via ge-1/2/22.0
                Label operation: Pop
                Load balance label:  None;
                Next hop type: Router, Next hop index: 1395
                Address: 0x2736290
                Next-hop reference count: 3
                Next hop: 203.0.113.2 via ge-1/2/18.0
                Label operation: Pop
                Load balance label:  None;
                State: <Active Int AckRequest MulticastRPF>
                Local AS:    64500
                Age: 54:05      Metric: 1
                Validation State: unverified
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
               FECs bound to route: P2MP root-addr 198.51.100.1, grp: 232.1.1.1,
 src: 192.168.219.11
                Primary Upstream : 198.51.100.3:0--198.51.100.2:0
                   RPF Nexthops  :
                        ge-1/2/15.0, 1.2.94.1, Label: 301568, weight: 0x1
                        ge-1/2/14.0, 1.2.3.1, Label: 301568, weight: 0x1
                Backup Upstream : 198.51.100.3:0--198.51.100.6:0
                   RPF Nexthops :
                       ge-1/2/20.0, 198.51.100.96, Label: 301584, weight: 0xfffe

                       ge-1/2/19.0, 198.51.100.36, Label: 301584, weight: 0xfffe
```

show route flow validation

List of Syntax

Syntax on page 1463
Syntax (EX Series Switches) on page 1463

Syntax

show route flow validation
<brief | detail>
<*ip-prefix*>
<table *table-name*>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route flow validation
<brief | detail>
<*ip-prefix*>
<table *table-name*>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display flow route information.

Options

none—Display flow route information.

brief | detail—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to brief.

ip-prefix—(Optional) IP address for the flow route.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

table *table-name*—(Optional) Display flow route information for all routing tables whose name begins with this string (for example, inet.0 and inet6.0 are both displayed when you run the **show route flow validation inet** command).

Required Privilege Level

view

List of Sample Output

show route flow validation on page 1464
show route flow validation (IPv6) on page 1464

Output Fields

Table 89 on page 1463 lists the output fields for the **show route flow validation** command. Output fields are listed in the approximate order in which they appear.

Table 89: show route flow validation Output Fields

Field Name	Field Description	Level of Output
routing-table-name	Name of the routing table (for example, inet.0).	All levels

Table 89: show route flow validation Output Fields *(continued)*

Field Name	Field Description	Level of Output
prefix	Route address.	All levels
Active unicast route	Active route in the routing table.	All levels
Dependent flow destinations	Number of flows for which there are routes in the routing table.	All levels
Origin	Source of the route flow.	All levels
Neighbor AS	Autonomous system identifier of the neighbor.	All levels
Flow destination	Number of entries and number of destinations that match the route flow.	All levels
Unicast best match	Destination that is the best match for the route flow.	All levels
Flags	Information about the route flow.	All levels

Sample Output

show route flow validation

```
user@host> show route flow validation
inet.0:
10.0.5.0/24Active unicast route
Dependent flow destinations: 1
Origin: 192.168.224.218, Neighbor AS: 64501
Flow destination (3 entries, 1 match origin)
Unicast best match: 10.0.5.0/24
Flags: SubtreeApex Consistent
```

show route flow validation (IPv6)

```
user@host> show route flow validation
inet6.0:
2001:db8::11:11:11:0/120
                Active unicast route
                    Dependent flow destinations: 2
                    Origin: 2001:db8::13:14:2:2, Neighbor AS: 2000
2001:db8::11:11:11:10/128
                Flow destination (1 entries, 1 match origin, next-as)
                    Unicast best match: 2001:db8::11:11:11:0/120
                    Flags: Consistent
2001:db8::11:11:11:30/128
                Flow destination (1 entries, 1 match origin, next-as)
                    Unicast best match: 2001:db8::11:11:11:0/120
                    Flags: Consistent
```

show route forwarding-table

List of Syntax

Syntax on page 1465
Syntax (MX Series Routers) on page 1465
Syntax (TX Matrix and TX Matrix Plus Routers) on page 1465

Syntax

show route forwarding-table
<detail | extensive | summary>
<all>
<ccc *interface-name*>
<destination *destination-prefix*>
<family *family* | matching *matching*>
<interface-name *interface-name*>
<label *name*>
<matching *matching*>
<multicast>
<table (default | *logical-system-name/routing-instance-name* | *routing-instance-name*)>
<vlan (all | *vlan-name*)>
<vpn *vpn*>

Syntax (MX Series Routers)

show route forwarding-table
<detail | extensive | summary>
<all>
<bridge-domain (all | *domain-name*)>
<ccc *interface-name*>
<destination *destination-prefix*>
<family *family* | matching *matching*>
<interface-name *interface-name*>
<label *name*>
<learning-vlan-id *learning-vlan-id*>
<matching *matching*>
<multicast>
<table (default | *logical-system-name/routing-instance-name* | *routing-instance-name*)>
<vlan (all | *vlan-name*)>
<vpn *vpn*>

Syntax (TX Matrix and TX Matrix Plus Routers)

show route forwarding-table
<detail | extensive | summary>
<all>
<ccc *interface-name*>
<destination *destination-prefix*>
<family *family* | matching *matching*>
<interface-name *interface-name*>
<matching *matching*>
<label *name*>
<lcc *number*>
<multicast>
<table *routing-instance-name*>
<vpn *vpn*>

Release Information

Command introduced before Junos OS Release 7.4.
Option **bridge-domain** introduced in Junos OS Release 7.5

Option **learning-vlan-id** introduced in Junos OS Release 8.4
Options **all** and **vlan** introduced in Junos OS Release 9.6.
Command introduced in Junos OS Release 11.3 for the QFX Series.
Command introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description Display the Routing Engine's forwarding table, including the network-layer prefixes and their next hops. This command is used to help verify that the routing protocol process has relayed the correction information to the forwarding table. The Routing Engine constructs and maintains one or more routing tables. From the routing tables, the Routing Engine derives a table of active routes, called the forwarding table.

NOTE: The Routing Engine copies the forwarding table to the Packet Forwarding Engine, the part of the router that is responsible for forwarding packets. To display the entries in the Packet Forwarding Engine's forwarding table, use the **show pfe route** command.

Options **none**—Display the routes in the forwarding tables. By default, the **show route forwarding-table** command does not display information about private, or internal, forwarding tables.

detail | extensive | summary—(Optional) Display the specified level of output.

all—(Optional) Display routing table entries for all forwarding tables, including private, or internal, tables.

bridge-domain (all | *bridge-domain-name*)—(MX Series routers only) (Optional) Display route entries for all bridge domains or the specified bridge domain.

ccc *interface-name*—(Optional) Display route entries for the specified circuit cross-connect interface.

destination *destination-prefix*—(Optional) Destination prefix.

family *family*—(Optional) Display routing table entries for the specified family: **bridge (ccc | destination | detail | extensive | interface-name | label | learning-vlan-id | matching | multicast | summary | table | vlan | vpn)**, **ethernet-switching**, **evpn**, **fibre-channel**, **fmembers**, **inet**, **inet6**, **iso**, **mcsnoop-inet**, **mcsnoop-inet6**, **mpls**, **satellite-inet**, **satellite-inet6**, **satellite-vpls**, **tnp**, **unix**, **vpls**, or **vlan-classification**.

interface-name *interface-name*—(Optional) Display routing table entries for the specified interface.

label *name*—(Optional) Display route entries for the specified label.

lcc *number*—(TX Matrix and TX matrix Plus routers only) (Optional) On a routing matrix composed of a TX Matrix router and T640 routers, display information for the specified T640 router (or line-card chassis) connected to the TX Matrix router. On a routing matrix composed of the TX Matrix Plus router and T1600 or T4000 routers, display information for the specified router (line-card chassis) connected to the TX Matrix Plus router.

Replace *number* with the following values depending on the LCC configuration:

- 0 through 3, when T640 routers are connected to a TX Matrix router in a routing matrix.
- 0 through 3, when T1600 routers are connected to a TX Matrix Plus router in a routing matrix.
- 0 through 7, when T1600 routers are connected to a TX Matrix Plus router with 3D SIBs in a routing matrix.
- 0, 2, 4, or 6, when T4000 routers are connected to a TX Matrix Plus router with 3D SIBs in a routing matrix.

learning-vlan-id *learning-vlan-id*—(MX Series routers only) (Optional) Display learned information for all VLANs or for the specified VLAN.

matching *matching*—(Optional) Display routing table entries matching the specified prefix or prefix length.

multicast—(Optional) Display routing table entries for multicast routes.

table —(Optional) Display route entries for all the routing tables in the main routing instance or for the specified routing instance. If your device supports logical systems, you can also display route entries for the specified logical system and routing instance. To view the routing instances on your device, use the show route instance command.

vlan (all | *vlan-name*)—(Optional) Display information for all VLANs or for the specified VLAN.

vpn *vpn*—(Optional) Display routing table entries for a specified VPN.

Required Privilege Level
view

List of Sample Output

show route forwarding-table table logical-system-name/routing-instance-name on page 1479
show route forwarding-table vpn on page 1480

Output Fields Table 90 on page 1468 lists the output fields for the **show route forwarding-table** command. Output fields are listed in the approximate order in which they appear. Field names might be abbreviated (as shown in parentheses) when no level of output is specified, or when the **detail** keyword is used instead of the **extensive** keyword.

Table 90: show route forwarding-table Output Fields

Field Name	Field Description	Level of Output
Logical system	Name of the logical system. This field is displayed if you specify the **table** ***logical-system-name/routing-instance-name*** option on a device that is configured for and supports logical systems.	All levels
Routing table	Name of the routing table (for example, inet, inet6, mpls).	All levels

Table 90: show route forwarding-table Output Fields *(continued)*

Field Name	Field Description	Level of Output
Enabled protocols	The features and protocols that have been enabled for a given routing table. This field can contain the following values: • BUM hashing—BUM hashing is enabled. • MAC Stats—Mac Statistics is enabled. • Bridging—Routing instance is a normal layer 2 bridge. • No VLAN—No VLANs are associated with the bridge domain. • All VLANs—The **vlan-id all** statement has been enabled for this bridge domain. • Single VLAN—Single VLAN ID is associated with the bridge domain. • MAC action drop—New MACs will be dropped when the MAC address limit is reached. • Dual VLAN—Dual VLAN tags are associated with the bridge domain • No local switching—No local switching is enabled for this routing instance.. • Learning disabled—Layer 2 learning is disabled for this routing instance. • MAC limit reached—The maximum number of MAC addresses that was configured for this routing instance has been reached. • VPLS—The VPLS protocol is enabled. • No IRB l2-copy—The no-irb-layer-2-copy feature is enabled for this routing instance. • ACKed by all peers—All peers have acknowledged this routing instance. • BUM Pruning—BUM pruning is enabled on the VPLS instance. • Def BD VXLAN—VXLAN is enabled for the default bridge domain. • EVPN—EVPN protocol is enabled for this routing instance. • Def BD OVSDB—Open vSwitch Database (OVSDB) is enabled on the default bridge domain. • Def BD Ingress replication—VXLAN ingress node replication is enabled on the default bridge domain. • L2 backhaul—Layer 2 backhaul is enabled. • FRR optimize—Fast reroute optimization • MAC pinning—MAC pinning is enabled for this bridge domain. • MAC Aging Timer—The MAC table aging time is set per routing instance. • EVPN VXLAN—This routing instance supports EVPN with VXLAN encapsulation. • PBBN—This routing instance is configured as a provider backbone bridged network. • PBN—This routing instance is configured as a provider bridge network. • ETREE—The ETREE protocol is enabled on this EVPN routing instance. • ARP/NDP suppression—EVPN ARP NDP suppression is enabled in this routing instance. • Def BD EVPN VXLAN—EVPN VXLAN is enabled for the default bridge domain. • MPLS control word—Control word is enabled for this MPLS routing instance.	All levels
Address family	Address family (for example, **IP**, **IPv6**, **ISO**, **MPLS**, and **VPLS**).	All levels
Destination	Destination of the route.	**detail extensive**

Table 90: show route forwarding-table Output Fields *(continued)*

Field Name	Field Description	Level of Output
Route Type (Type)	How the route was placed into the forwarding table. When the **detail** keyword is used, the route type might be abbreviated (as shown in parentheses): • **cloned** (**clon**)—(TCP or multicast only) Cloned route. • **destination** (**dest**)—Remote addresses directly reachable through an interface. • **destination down** (**iddn**)—Destination route for which the interface is unreachable. • **interface cloned** (**ifcl**)—Cloned route for which the interface is unreachable. • **route down** (**ifdn**)—Interface route for which the interface is unreachable. • **ignore** (**ignr**)—Ignore this route. • **interface** (**intf**)—Installed as a result of configuring an interface. • **permanent** (**perm**)—Routes installed by the kernel when the routing table is initialized. • **user**—Routes installed by the routing protocol process or as a result of the configuration.	All levels
Route Reference (RtRef)	Number of routes to reference.	**detail extensive**
Flags	Route type flags: • **none**—No flags are enabled. • **accounting**—Route has accounting enabled. • **cached**—Cache route. • **incoming-iface** ***interface-number***—Check against incoming interface. • **prefix load balance**—Load balancing is enabled for this prefix. • **rt nh decoupled**—Route has been decoupled from the next hop to the destination. • **sent to PFE**—Route has been sent to the Packet Forwarding Engine. • **static**—Static route.	**extensive**
Next hop	IP address of the next hop to the destination.	**detail extensive**

Table 90: show route forwarding-table Output Fields *(continued)*

Field Name	Field Description	Level of Output
Next hop Type (Type)	Next-hop type. When the **detail** keyword is used, the next-hop type might be abbreviated (as indicated in parentheses): • **broadcast (bcst)**—Broadcast. • **deny**—Deny. • **discard (dscd)** —Discard. • **hold**—Next hop is waiting to be resolved into a unicast or multicast type. • **indexed (idxd)**—Indexed next hop. • **indirect (indr)**—Indirect next hop. • **local (locl)**—Local address on an interface. • **routed multicast (mcrt)**—Regular multicast next hop. • **multicast (mcst)**—Wire multicast next hop (limited to the LAN). • **multicast discard (mdsc)**—Multicast discard. • **multicast group (mgrp)**—Multicast group member. • **receive (recv)**—Receive. • **reject (rjct)**—Discard. An ICMP unreachable message was sent. • **resolve (rslv)**—Resolving the next hop. • **unicast (ucst)**—Unicast. • **unilist (ulst)**—List of unicast next hops. A packet sent to this next hop goes to any next hop in the list.	**detail extensive**
Index	Software index of the next hop that is used to route the traffic for a given prefix.	**detail extensive** none
Route interface-index	Logical interface index from which the route is learned. For example, for interface routes, this is the logical interface index of the route itself. For static routes, this field is zero. For routes learned through routing protocols, this is the logical interface index from which the route is learned.	**extensive**
Reference (NhRef)	Number of routes that refer to this next hop.	**detail extensive** none
Next-hop interface (Netif)	Interface used to reach the next hop.	**detail extensive** none
Weight	Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible (see the **Balance** field description).	**extensive**
Balance	Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a router is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.	**extensive**
RPF interface	List of interfaces from which the prefix can be accepted. Reverse path forwarding (RPF) information is displayed only when **rpf-check** is configured on the interface.	**extensive**

Sample Output

show route forwarding-table

```
user@host> show route forwarding-table
Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    46     4
0.0.0.0/32         perm     0                    dscd    44     1
172.16.1.0/24         ifdn     0                    rslv   608     1 ge-2/0/1.0
172.16.1.0/32        iddn     0 172.16.1.0           recv   606     1 ge-2/0/1.0
172.16.1.1/32         user     0                    rjct    46     4
172.16.1.1/32         intf     0 172.16.1.1             locl   607     2
172.16.1.1/32         iddn     0 172.16.1.1             locl   607     2
172.16.1.255/32       iddn     0 ff:ff:ff:ff:ff:ff  bcst   605     1 ge-2/0/1.0
10.0.0.0/24        intf     0                    rslv   616     1 ge-2/0/0.0
10.0.0.0/32        dest     0 10.0.0.0           recv   614     1 ge-2/0/0.0
10.0.0.1/32        intf     0 10.0.0.1           locl   615     2
10.0.0.1/32        dest     0 10.0.0.1           locl   615     2
10.0.0.255/32      dest     0 10.0.0.255         bcst   613     1 ge-2/0/0.0
10.1.1.0/24        ifdn     0                    rslv   612     1 ge-2/0/1.0
10.1.1.0/32        iddn     0 10.1.1.0           recv   610     1 ge-2/0/1.0
10.1.1.1/32        user     0                    rjct    46     4
10.1.1.1/32        intf     0 10.1.1.1           locl   611     2
10.1.1.1/32        iddn     0 10.1.1.1           locl   611     2
10.1.1.255/32      iddn     0 ff:ff:ff:ff:ff:ff  bcst   609     1 ge-2/0/1.0
10.206.0.0/16      user     0 10.209.63.254      ucst   419    20 fxp0.0
10.209.0.0/16      user     1 0:12:1e:ca:98:0    ucst   419    20 fxp0.0
10.209.0.0/18      intf     0                    rslv   418     1 fxp0.0
10.209.0.0/32      dest     0 10.209.0.0         recv   416     1 fxp0.0
10.209.2.131/32    intf     0 10.209.2.131       locl   417     2
10.209.2.131/32    dest     0 10.209.2.131       locl   417     2
10.209.17.55/32    dest     0 0:30:48:5b:78:d2   ucst   435     1 fxp0.0
10.209.63.42/32    dest     0 0:23:7d:58:92:ca   ucst   434     1 fxp0.0
10.209.63.254/32   dest     0 0:12:1e:ca:98:0    ucst   419    20 fxp0.0
10.209.63.255/32   dest     0 10.209.63.255      bcst   415     1 fxp0.0
10.227.0.0/16      user     0 10.209.63.254      ucst   419    20 fxp0.0

...

Routing table: iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                   rjct    27     1
47.0005.80ff.f800.0000.0108.0003.0102.5524.5220.00
intf     0                   locl    28     1

Routing table: inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                   rjct     6     1
ff00::/8           perm     0                   mdsc     4     1
ff02::1/128        perm     0 ff02::1           mcst     3     1

Routing table: ccc
MPLS:
Interface.Label    Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct 16     1
100004(top)fe-0/0/1.0
```


show route forwarding-table detail

```
user@host> show route forwarding-table detail
Routing table: inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            user     2 0:90:69:8e:b1:1b   ucst   132     4 fxp0.0
default            perm     0                    rjct    14     1
10.1.1.0/24        intf     0 ff.3.0.21          ucst   322     1 so-5/3/0.0
10.1.1.0/32        dest     0 10.1.1.0           recv   324     1 so-5/3/0.0
10.1.1.1/32        intf     0 10.1.1.1           locl   321     1
10.1.1.255/32      dest     0 10.1.1.255         bcst   323     1 so-5/3/0.0
10.21.21.0/24      intf     0 ff.3.0.21          ucst   326     1 so-5/3/0.0
10.21.21.0/32      dest     0 10.21.21.0         recv   328     1 so-5/3/0.0
10.21.21.1/32      intf     0 10.21.21.1         locl   325     1
10.21.21.255/32    dest     0 10.21.21.255       bcst   327     1 so-5/3/0.0
127.0.0.1/32       intf     0 127.0.0.1          locl   320     1
172.17.28.19/32    clon     1 192.168.4.254      ucst   132     4 fxp0.0
172.17.28.44/32    clon     1 192.168.4.254      ucst   132     4 fxp0.0

...

Routing table: private1__.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    46     1
10.0.0.0/8         intf     0                    rslv   136     1 fxp1.0
10.0.0.0/32        dest     0 10.0.0.0           recv   134     1 fxp1.0
10.0.0.4/32        intf     0 10.0.0.4           locl   135     2
10.0.0.4/32        dest     0 10.0.0.4           locl   135     2

...

Routing table: iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    38     1

Routing table: inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    22     1
ff00::/8           perm     0                    mdsc    21     1
ff02::1/128        perm     0 ff02::1            mcst    17     1

...

Routing table: mpls
MPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct 28     1
```

show route forwarding-table destination extensive (Weights and Balances)

```
user@host> show route forwarding-table destination 3.4.2.1 extensive
Routing table: inet [Index 0]
Internet:

Destination:  3.4.2.1/32
  Route type: user
```

```
  Route reference: 0                  Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: unilist              Index: 262143   Reference: 1
  Nexthop: 172.16.4.4
  Next-hop type: unicast              Index: 335      Reference: 2
  Next-hop interface: so-1/1/0.0      Weight: 22      Balance: 3
  Nexthop: 145.12.1.2
  Next-hop type: unicast              Index: 337      Reference: 2
  Next-hop interface: so-0/1/2.0      Weight: 33      Balance: 33
```

show route forwarding-table extensive

```
user@host> show route forwarding-table extensive
Routing table: inet [Index 0]
Internet:

Destination:  default
  Route type: user
  Route reference: 2                  Route interface-index: 0
  Flags: sent to PFE
  Nexthop: 00:00:5E:00:53:1b
  Next-hop type: unicast              Index: 132      Reference: 4
  Next-hop interface: fxp0.0

Destination:  default
  Route type: permanent
  Route reference: 0                  Route interface-index: 0
  Flags: none
  Next-hop type: reject               Index: 14       Reference: 1

Destination:  127.0.0.1/32
  Route type: interface
  Route reference: 0                  Route interface-index: 0
  Flags: sent to PFE
  Nexthop: 127.0.0.1
  Next-hop type: local                Index: 320      Reference: 1

...

Routing table: private1__.inet [Index 1]
Internet:

Destination:  default
  Route type: permanent
  Route reference: 0                  Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject               Index: 46       Reference: 1

Destination:  10.0.0.0/8
  Route type: interface
  Route reference: 0                  Route interface-index: 3
  Flags: sent to PFE
  Next-hop type: resolve              Index: 136      Reference: 1
  Next-hop interface: fxp1.0

...

Routing table: iso [Index 0]
ISO:

Destination:  default
```



```
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 38       Reference: 1

Routing table: inet6 [Index 0]
Internet6:

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 22       Reference: 1

Destination:  ff00::/8
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: multicast discard      Index: 21       Reference: 1

...

Routing table: private1__.inet6 [Index 1]
Internet6:

Destination:  default
  Route type: permanent
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Next-hop type: reject                 Index: 54       Reference: 1

Destination:  fe80::2a0:a5ff:fe3d:375/128
  Route type: interface
  Route reference: 0                    Route interface-index: 0
  Flags: sent to PFE
  Nexthop: fe80::2a0:a5ff:fe3d:375
  Next-hop type: local                  Index: 75       Reference: 1

...
```

show route forwarding-table extensive (RPF)

The next example is based on the following configuration, which enables an RPF check on all routes that are learned from this interface, including the interface route:

```
so-1/1/0 {
    unit 0 {
        family inet {
            rpf-check;
            address 192.0.2.2/30;
        }
    }
}
```

```
user@host> show route forwarding-table extensive
Routing table: inet [Index 0]
Internet:
...
...
Destination:  192.0.2.3/32
```

```
Route type: destination
Route reference: 0                    Route interface-index: 67
Flags: sent to PFE
Nexthop: 192.0.2.3
Next-hop type: broadcast              Index: 328      Reference: 1
Next-hop interface: so-1/1/0.0
RPF interface: so-1/1/0.0
```

show route forwarding-table family mpls

```
user@host> show route forwarding-table family mpls
Routing table: mpls
MPLS:
Destination        Type RtRef Next hop          Type Index NhRef Netif
default            perm     0                   rjct    19     1
0                  user     0                   recv    18     3
1                  user     0                   recv    18     3
2                  user     0                   recv    18     3
100000             user     0 10.31.1.6         swap  100001     fe-1/1/0.0
800002             user     0                   Pop              vt-0/3/0.32770

vt-0/3/0.32770 (VPLS)
                   user     0                   indr   351     4
                                                Push 800000, Push 100002(top)
so-0/0/0.0
```

show route forwarding-table family mpls ccc ge-0/0/1.1004

```
user@host>show route forwarding-table mpls ccc ge-0/0/1.1004
Routing table: default.mpls
MPLS:
Destination        Type RtRef Next hop          Type  Index    NhRef Netif
ge-0/0/1.1004   (CCC) user     0                    ulst   1048577     2
                                                    comp      754         3
                                                   comp      755         3
                                                comp      756         3

Routing table: __mpls-oam__.mpls
MPLS:
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      556     1
```

show route forwarding-table family vpls

```
user@host> show route forwarding-table family vpls
Routing table: green.vpls
VPLS:
Destination        Type RtRef Next hop          Type Index NhRef Netif
default            dynm     0                   flood   353     1
default            perm     0                   rjct    298     1
fe-0/1/0.0         dynm     0                   flood   355     1
00:00:5E:00:53:1f/48             <<<<<Remote CE

                   dynm     0                   indr   351     4
                                                Push 800000, Push 100002(top)
so-0/0/0.0
00:00:5E:00:53:1f/48             <<<<<<Local CE

                   dynm     0                   ucst   354     2 fe-0/1/0.0
```


show route forwarding-table vpls (Broadcast, unknown unicast, and multicast (BUM) hashing is enabled)

```
user@host> show route forwarding-table vpls
Routing table: green.vpls
VPLS:
Enabled protocols: BUM hashing
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      519     1
lsi.1048832        intf     0                    indr  1048574     4
                              172.16.3.2           Push 262145      621     2
ge-3/0/0.0
00:00:5E:00:53:01/48 user     0                  ucst      590     5 ge-2/3/9.0
0x30003/51         user     0                    comp      627     2
ge-2/3/9.0         intf     0                    ucst      590     5 ge-2/3/9.0
ge-3/1/3.0         intf     0                    ucst      619     4 ge-3/1/3.0
0x30002/51         user     0                    comp      600     2
0x30001/51         user     0                    comp      597     2
```

show route forwarding-table vpls (Broadcast, unknown unicast, and multicast (BUM) hashing is enabled with MAC Statistics)

```
user@host> show route forwarding-table vpls
Routing table: green.vpls
VPLS:
Enabled protocols: BUM hashing, MAC Stats
Destination        Type RtRef Next hop           Type Index    NhRef Netif
default            perm     0                    dscd      519     1
lsi.1048834        intf     0                    indr  1048574     4
                              172.16.3.2           Push 262145      592     2
ge-3/0/0.0
00:19:e2:25:d0:01/48 user     0                  ucst      590     5 ge-2/3/9.0
0x30003/51         user     0                    comp      630     2
ge-2/3/9.0         intf     0                    ucst      590     5 ge-2/3/9.0
ge-3/1/3.0         intf     0                    ucst      591     4 ge-3/1/3.0
0x30002/51         user     0                    comp      627     2
0x30001/51         user     0                    comp      624     2
```

show route forwarding-table family vpls extensive

```
user@host> show route forwarding-table family vpls extensive
Routing table: green.vpls [Index 2]
VPLS:

Destination:  default
  Route type: dynamic
  Route reference: 0                   Route interface-index: 72
  Flags: sent to PFE
  Next-hop type: flood                 Index: 289      Reference: 1
  Next-hop type: unicast               Index: 291      Reference: 3
  Next-hop interface: fe-0/1/3.0
  Next-hop type: unicast               Index: 290      Reference: 3
  Next-hop interface: fe-0/1/2.0

Destination:  default
  Route type: permanent
  Route reference: 0                   Route interface-index: 0
  Flags: none
  Next-hop type: discard               Index: 341      Reference: 1

Destination:  fe-0/1/2.0
```

```
    Route type: dynamic
    Route reference: 0                   Route interface-index: 69
    Flags: sent to PFE
    Next-hop type: flood                 Index: 293      Reference: 1
    Next-hop type: indirect              Index: 363      Reference: 4
    Next-hop type: Push 800016
    Next-hop interface: at-1/0/1.0
    Next-hop type: indirect              Index: 301      Reference: 5
    Next hop: 10.31.3.2
    Next-hop type: Push 800000
    Next-hop interface: fe-0/1/1.0
    Next-hop type: unicast               Index: 291      Reference: 3
    Next-hop interface: fe-0/1/3.0

  Destination:  fe-0/1/3.0
    Route type: dynamic
    Route reference: 0                   Route interface-index: 70
    Flags: sent to PFE
    Next-hop type: flood                 Index: 292      Reference: 1
    Next-hop type: indirect              Index: 363      Reference: 4
    Next-hop type: Push 800016
    Next-hop interface: at-1/0/1.0
    Next-hop type: indirect              Index: 301      Reference: 5
    Next hop: 10.31.3.2
    Next-hop type: Push 800000
    Next-hop interface: fe-0/1/1.0
    Next-hop type: unicast               Index: 290      Reference: 3
    Next-hop interface: fe-0/1/2.0

  Destination:  00:00:5E:00:53:01/48
    Route type: dynamic
    Route reference: 0                   Route interface-index: 70
    Flags: sent to PFE, prefix load balance
    Next-hop type: unicast               Index: 291      Reference: 3
    Next-hop interface: fe-0/1/3.0
    Route used as destination:
      Packet count:        6640    Byte count:      675786
    Route used as source
      Packet count:        6894    Byte count:      696424

  Destination:  00:00:5E:00:53:04/48
    Route type: dynamic
    Route reference: 0                   Route interface-index: 69
    Flags: sent to PFE, prefix load balance
    Next-hop type: unicast               Index: 290      Reference: 3
    Next-hop interface: fe-0/1/2.0
    Route used as destination:
      Packet count:          96    Byte count:        8079
    Route used as source:
      Packet count:         296    Byte count:       24955

  Destination:  00:00:5E:00:53:05/48
    Route type: dynamic
    Route reference: 0                   Route interface-index: 74
    Flags: sent to PFE, prefix load balance
    Next-hop type: indirect              Index: 301      Reference: 5
    Next hop: 10.31.3.2
    Next-hop type: Push 800000
    Next-hop interface: fe-0/1/1.0
```


show route forwarding-table table default

```
user@host> show route forwarding-table table default
Routing table: default.inet
Internet:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    36     2
0.0.0.0/32         perm     0                    dscd    34     1
10.0.60.0/30       user     0 10.0.60.13         ucst   713     5 fe-0/1/3.0
10.0.60.12/30      intf     0                    rslv   688     1 fe-0/1/3.0
10.0.60.12/32      dest     0 10.0.60.12         recv   686     1 fe-0/1/3.0
10.0.60.13/32      dest     0 0:5:85:8b:bc:22    ucst   713     5 fe-0/1/3.0
10.0.60.14/32      intf     0 10.0.60.14         locl   687     2
10.0.60.14/32      dest     0 10.0.60.14         locl   687     2
10.0.60.15/32      dest     0 10.0.60.15         bcst   685     1 fe-0/1/3.0
10.0.67.12/30      user     0 10.0.60.13         ucst   713     5 fe-0/1/3.0
10.0.80.0/30       ifdn     0 ff.3.0.21          ucst   676     1 so-0/0/1.0
10.0.80.0/32       dest     0 10.0.80.0          recv   678     1 so-0/0/1.0
10.0.80.2/32       user     0                    rjct    36     2
10.0.80.2/32       intf     0 10.0.80.2          locl   675     1
10.0.80.3/32       dest     0 10.0.80.3          bcst   677     1 so-0/0/1.0
10.0.90.12/30      intf     0                    rslv   684     1 fe-0/1/0.0
10.0.90.12/32      dest     0 10.0.90.12         recv   682     1 fe-0/1/0.0
10.0.90.14/32      intf     0 10.0.90.14         locl   683     2
10.0.90.14/32      dest     0 10.0.90.14         locl   683     2
10.0.90.15/32      dest     0 10.0.90.15         bcst   681     1 fe-0/1/0.0
10.5.0.0/16        user     0 192.168.187.126    ucst   324    15 fxp0.0
10.10.0.0/16       user     0 192.168.187.126    ucst   324    15 fxp0.0
10.13.10.0/23      user     0 192.168.187.126    ucst   324    15 fxp0.0
10.84.0.0/16       user     0 192.168.187.126    ucst   324    15 fxp0.0
10.150.0.0/16      user     0 192.168.187.126    ucst   324    15 fxp0.0
10.157.64.0/19     user     0 192.168.187.126    ucst   324    15 fxp0.0
10.209.0.0/16      user     0 192.168.187.126    ucst   324    15 fxp0.0

...

Routing table: default.iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    60     1

Routing table: default.inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct    44     1
::/128             perm     0                    dscd    42     1
ff00::/8           perm     0                    mdsc    43     1
ff02::1/128        perm     0 ff02::1            mcst    39     1

Routing table: default.mpls
MPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    dscd    50     1
```

show route forwarding-table table logical-system-name/routing-instance-name

```
user@host> show route forwarding-table table R4/vpn-red
Logical system: R4
Routing table: vpn-red.inet
Internet:
```

```
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   563     1
0.0.0.0/32         perm     0                    dscd   561     2
172.16.0.1/32         user     0                    dscd   561     2
172.16.2.0/24         intf     0                    rslv   771     1 ge-1/2/0.3
172.16.2.0/32        dest     0 172.16.2.0           recv   769     1 ge-1/2/0.3
172.16.2.1/32         intf     0 172.16.2.1            locl   770     2
172.16.2.1/32         dest     0 172.16.2.1            locl   770     2
172.16.2.2/32         dest     0 0.4.80.3.0.1b.c0.d5.e4.bd.0.1b.c0.d5.e4.bc.8.0
                                                 ucst   789     1 ge-1/2/0.3
172.16.2.255/32      dest     0 172.16.2.255         bcst   768     1 ge-1/2/0.3
172.16.233.0/4        perm     1                    mdsc   562     1
172.16.233.1/32       perm     0 172.16.233.1          mcst   558     1
255.255.255.255/32 perm     0                    bcst   559     1

Logical system: R4
Routing table: vpn-red.iso
ISO:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   608     1

Logical system: R4
Routing table: vpn-red.inet6
Internet6:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    rjct   708     1
::/128             perm     0                    dscd   706     1
ff00::/8           perm     0                    mdsc   707     1
ff02::1/128        perm     0 ff02::1            mcst   704     1

Logical system: R4
Routing table: vpn-red.mpls
MPLS:
Destination        Type RtRef Next hop           Type Index NhRef Netif
default            perm     0                    dscd   638
```

show route forwarding-table vpn

```
user@host> show route forwarding-table vpn VPN-A
      Routing table:: VPN-A.inet
Internet:
      Destination        Type RtRef Nexthop           Type Index NhRef Netif
      default            perm     0                   rjct     4     4
      10.39.10.20/30     intf     0 ff.3.0.21         ucst    40     1
      so-0/0/0.0
      10.39.10.21/32     intf     0 10.39.10.21       locl    36     1
      10.255.14.172/32   user     0                   ucst    69     2
      so-0/0/0.0
      10.255.14.175/32   user     0                   indr    81     3
                                                      Push 100004, Push
      100004(top) so-1/0/0.0
      172.16.233.0/4        perm     2                   mdsc     5     3
      172.16.233.1/32       perm     0 172.16.233.1         mcst     1     8
      172.16.233.5/32       user     1 172.16.233.5         mcst     1     8
      255.255.255.255/32 perm     0                   bcst     2     3
```

On QFX5200, the results for this command look like this:

```
show route forwarding-table family mpls
```

```
Routing table: default.mpls
MPLS:
Destination Type RtRef Next hop Type Index NhRef Netif
default perm 0 dscd 65 1
0 user 0 recv 64 4
1 user 0 recv 64 4
2 user 0 recv 64 4
13 user 0 recv 64 4
300384 user 0 9.1.1.1 Pop 1711 2 xe-0/0/34.0
300384(S=0) user 0 9.1.1.1 Pop 1712 2 xe-0/0/34.0
300400 user 0 ulst 131071 2
                              10.1.1.2 Pop 1713 1 xe-0/0/38.0
                              172.16.11.2 Pop 1714 1 xe-0/0/40.0
300400(S=0) user 0 ulst 131072 2
                              10.1.1.2 Pop 1715 1 xe-0/0/38.0
                              172.16.11.2 Pop 1716 1 xe-0/0/40.0

Routing table: __mpls-oam__.mpls
MPLS:
Destination Type RtRef Next hop Type Index NhRef Netif
default perm 0 dscd 1681 1
```


show route hidden

Syntax

show route hidden
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Release Information

Command introduced before Junos OS Release 7.4.

Description

Display only hidden route information. A hidden route is unusable, even if it is the best path.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

logical-system (all | ***logical-system-name*****)**—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

Related Documentation

- *Understanding Hidden Routes*

List of Sample Output

show route hidden on page 1482
show route hidden detail on page 1483
show route hidden extensive on page 1483
show route hidden terse on page 1483

Output Fields

For information about output fields, see the output field table for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route hidden

```
user@host> show route hidden
inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
127.0.0.1/32        [Direct/0] 04:26:38
                    > via lo0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
10.5.5.5/32         [BGP/170] 03:44:10, localpref 100, from 10.4.4.4
                      AS path: 100 I
```



```
                      Unusable
10.12.1.0/24        [BGP/170] 03:44:10, localpref 100, from 10.4.4.4
                      AS path: 100 I
                      Unusable
10.12.80.4/30       [BGP/170] 03:44:10, localpref 100, from 10.4.4.4
                      AS path: I
                      Unusable
...
```

show route hidden detail

user@host> **show route hidden detail**

```
inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
127.0.0.1/32 (1 entry, 0 announced)
         Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Hidden Martian Int>
                Local AS:     1
                Age: 4:27:37
                Task: IF
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete

10.5.5.5/32 (1 entry, 0 announced)
         BGP    Preference: 170/-101
                Route Distinguisher: 10.4.4.4:4
                Next hop type: Unusable
                Next-hop reference count: 6
                State: <Secondary Hidden Int Ext>
                Local AS:     1 Peer AS:     1
                Age: 3:45:09
                Task: BGP_1.10.4.4.4+2493
                AS path: 100 I
                Communities: target:1:999
                VPN Label: 100064
                Localpref: 100
                Router ID: 10.4.4.4
                Primary Routing Table bgp.l3vpn.0

...
```

show route hidden extensive

The output for the **show route hidden extensive** command is identical to that of the **show route hidden detail** command. For sample output, see show route hidden detail on page 1483.

show route hidden terse

user@host> **show route hidden terse**


```
inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  127.0.0.1/32       D   0                       >lo0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.5.5.5/32        B 170        100             Unusable        100 I
  10.12.1.0/24       B 170        100             Unusable        100 I
  10.12.80.4/30      B 170        100             Unusable        I

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

bgp.l3vpn.0: 3 destinations, 3 routes (0 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.4.4.4:4:10.5.5.5/32
                     B 170        100             Unusable        100 I
  10.4.4.4:4:10.12.1.0/24
                     B 170        100             Unusable        100 I
  10.4.4.4:4:10.12.80.4/30
                     B 170        100             Unusable        I

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```


show route inactive-path

List of Syntax

Syntax on page 1485
Syntax (EX Series Switches) on page 1485

Syntax

show route inactive-path
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route inactive-path
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display routes for destinations that have no active route. An inactive route is a route that was not selected as the best path.

Options

none—Display all inactive routes.

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

Related Documentation

- show route active-path on page 1392

List of Sample Output

show route inactive-path on page 1485
show route inactive-path detail on page 1486
show route inactive-path extensive on page 1487
show route inactive-path terse on page 1487

Output Fields

For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route inactive-path

```
user@host> show route inactive-path

inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
```

```
Restart Complete
+ = Active Route, - = Last Active, * = Both

10.12.100.12/30     [OSPF/10] 03:57:28, metric 1
                    > via so-0/3/0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.0/8          [Direct/0] 04:39:56
                    > via fxp1.0

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

10.12.80.0/30       [BGP/170] 04:38:17, localpref 100
                      AS path: 100 I
                    > to 10.12.80.1 via ge-6/3/2.0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

bgp.l3vpn.0: 3 destinations, 3 routes (0 active, 0 holddown, 3 hidden)
Restart Complete

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route inactive-path detail

```
user@host> show route inactive-path detail

inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete

10.12.100.12/30 (2 entries, 1 announced)
         OSPF   Preference: 10
                Next-hop reference count: 1
                Next hop: via so-0/3/0.0, selected
                State: <Int>
                Inactive reason: Route Preference
                Local AS:     1
                Age: 3:58:24    Metric: 1
                Area: 0.0.0.0
                Task: OSPF
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

10.0.0.0/8 (2 entries, 0 announced)
         Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via fxp1.0, selected
                State: <NotBest Int>
```

```
                Inactive reason: No difference
                Age: 4:40:52
                Task: IF
                AS path: I

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete

10.12.80.0/30 (2 entries, 1 announced)
         BGP    Preference: 170/-101
                Next-hop reference count: 6
                Source: 10.12.80.1
                Next hop: 10.12.80.1 via ge-6/3/2.0, selected
                State: <Ext>
                Inactive reason: Route Preference
                Peer AS:   100
                Age: 4:39:13
                Task: BGP_100.10.12.80.1+179
                AS path: 100 I
                Localpref: 100
                Router ID: 10.0.0.0
```

show route inactive-path extensive

The output for the **show route inactive-path extensive** command is identical to that of the **show route inactive-path detail** command. For sample output, see show route inactive-path detail on page 1486.

show route inactive-path terse

user@host> **show route inactive-path terse**

```
inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.12.100.12/30    O  10          1            >so-0/3/0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.0.0.0/8         D   0                       >fxp1.0

red.inet.0: 6 destinations, 8 routes (4 active, 0 holddown, 3 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  10.12.80.0/30      B 170        100            >10.12.80.1      100 I

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete
```



```
bgp.l3vpn.0: 3 destinations, 3 routes (0 active, 0 holddown, 3 hidden)
Restart Complete

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```


show route inactive-prefix

List of Syntax

Syntax on page 1489
Syntax (EX Series Switches) on page 1489

Syntax

show route inactive-prefix
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route inactive-prefix
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display inactive route destinations in each routing table.

Options

none—Display all inactive route destination.

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to brief.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show route inactive-prefix on page 1489
show route inactive-prefix detail on page 1490
show route inactive-prefix extensive on page 1490
show route inactive-prefix terse on page 1490

Output Fields

For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route inactive-prefix

```
user@host> show route inactive-prefix

inet.0: 14 destinations, 14 routes (13 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

127.0.0.1/32        [Direct/0] 00:04:54
                    > via lo0.0
```

show route inactive-prefix detail

```
user@host> show route inactive-prefix detail

inet.0: 14 destinations, 14 routes (13 active, 0 holddown, 1 hidden)
127.0.0.1/32 (1 entry, 0 announced)
         Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via lo0.0, selected
                State: <Hidden Martian Int>
                Age: 4:51
                Task: IF
                AS path: I00:04:54
                    > via lo0.0
```

show route inactive-prefix extensive

The output for the **show route inactive-prefix extensive** command is identical to that of the **show route inactive-path detail** command. For sample output, see show route inactive-prefix detail on page 1490.

show route inactive-prefix terse

```
user@host> show route inactive-prefix terse

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  127.0.0.1/32       D   0                       >lo0.0
```


show route instance

List of Syntax

Syntax on page 1491
Syntax (EX Series Switches and QFX Series) on page 1491

Syntax

show route instance
<brief | detail | summary>
<*instance-name*>
<logical-system (all | *logical-system-name*)>
<operational>

Syntax (EX Series Switches and QFX Series)

show route instance
<brief | detail | summary>
<*instance-name*>
<operational>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Command introduced in Junos OS Release 11.3 for the QFX Series.
Command introduced in Junos OS Release 14.1X53-D20 for the OCX Series.

Description

Display routing instance information.

Options

none—(Same as **brief**) Display standard information about all routing instances.

brief | detail | summary—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**. (These options are not available with the **operational** keyword.)

instance-name—(Optional) Display information for all routing instances whose name begins with this string (for example, **cust1**, **cust11**, and **cust111** are all displayed when you run the **show route instance cust1** command).

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

operational—(Optional) Display operational routing instances.

Required Privilege Level

view

Related Documentation

- Example: Transporting IPv6 Traffic Across IPv4 Using Filter-Based Tunneling on page 888
- *Example: Configuring the Helper Capability Mode for OSPFv3 Graceful Restart*

List of Sample Output

show route instance on page 1493
show route instance detail (Graceful Restart Complete) on page 1493
show route instance detail (Graceful Restart Incomplete) on page 1495

show route instance detail (VPLS Routing Instance) on page 1497
show route instance operational on page 1497
show route instance summary on page 1497

Output Fields Table 91 on page 1492 lists the output fields for the **show route instance** command. Output fields are listed in the approximate order in which they appear.

Table 91: show route instance Output Fields

Field Name	Field Description	Level of Output
Instance or *instance-name*	Name of the routing instance.	All levels
Operational Routing Instances	(**operational** keyword only) Names of all operational routing instances.	—
Type	Type of routing instance: **forwarding**, **l2vpn**, **no-forwarding**, **vpls**, **virtual-router**, or **vrf**.	All levels
State	State of the routing instance: **active** or **inactive**.	**brief detail** none
Interfaces	Name of interfaces belonging to this routing instance.	**brief detail** none
Restart State	Status of graceful restart for this instance: **Pending** or **Complete**.	**detail**
Path selection timeout	Maximum amount of time, in seconds, remaining until graceful restart is declared complete. The default is **300**.	**detail**
Tables	Tables (and number of routes) associated with this routing instance.	**brief detail** none
Route-distinguisher	Unique route distinguisher associated with this routing instance.	**detail**
Vrf-import	VPN routing and forwarding instance import policy name.	**detail**
Vrf-export	VPN routing and forwarding instance export policy name.	**detail**
Vrf-import-target	VPN routing and forwarding instance import target community name.	**detail**
Vrf-export-target	VPN routing and forwarding instance export target community name.	**detail**
Vrf-edge-protection-id	Context identifier configured for edge-protection.	**detail**
Fast-reroute-priority	Fast reroute priority setting for a VPLS routing instance: **high**, **medium**, or **low**. The default is **low**.	**detail**
Restart State	Restart state: • **Pending:***protocol-name*—List of protocols that have not yet completed graceful restart for this routing table. • **Complete**—All protocols have restarted for this routing table.	**detail**

Table 91: show route instance Output Fields *(continued)*

Field Name	Field Description	Level of Output
Primary rib	Primary table for this routing instance.	**brief** none **summary**
Active/holddown/hidden	Number of active, hold-down, and hidden routes.	All levels

Sample Output

show route instance

```
user@host> show route instance
Instance             Type
         Primary RIB                                     Active/holddown/hidden
master               forwarding
         inet.0                                          16/0/1
         iso.0                                           1/0/0
         mpls.0                                          0/0/0
         inet6.0                                         2/0/0
         l2circuit.0                                     0/0/0
__juniper_private1__ forwarding
         __juniper_private1__.inet.0                     12/0/0
         __juniper_private1__.inet6.0                    1/0/0
```

show route instance detail (Graceful Restart Complete)

```
user@host> show route instance detail
master:
  Router ID: 10.255.14.176
  Type: forwarding        State: Active
  Restart State: Complete Path selection timeout: 300
  Tables:
    inet.0                 : 17 routes (15 active, 0 holddown, 1 hidden)
    Restart Complete
    inet.3                 : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Complete
    iso.0                  : 1 routes (1 active, 0 holddown, 0 hidden)
    Restart Complete
    mpls.0                 : 19 routes (19 active, 0 holddown, 0 hidden)
    Restart Complete
    bgp.l3vpn.0            : 10 routes (10 active, 0 holddown, 0 hidden)
    Restart Complete
    inet6.0                : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Complete
    bgp.l2vpn.0            : 1 routes (1 active, 0 holddown, 0 hidden)
    Restart Complete
BGP-INET:
  Router ID: 10.69.103.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.103
  Route-distinguisher: 10.255.14.176:103
  Vrf-import: [ BGP-INET-import ]
  Vrf-export: [ BGP-INET-export ]
  Tables:
    BGP-INET.inet.0        : 4 routes (4 active, 0 holddown, 0 hidden)
```

```
    Restart Complete
BGP-L:
  Router ID: 10.69.104.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.104
  Route-distinguisher: 10.255.14.176:104
  Vrf-import: [ BGP-L-import ]
  Vrf-export: [ BGP-L-export ]
  Tables:
    BGP-L.inet.0           : 4 routes (4 active, 0 holddown, 0 hidden)
    Restart Complete
    BGP-L.mpls.0           : 3 routes (3 active, 0 holddown, 0 hidden)
    Restart Complete
L2VPN:
  Router ID: 0.0.0.0
  Type: l2vpn             State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.512
  Route-distinguisher: 10.255.14.176:512
  Vrf-import: [ L2VPN-import ]
  Vrf-export: [ L2VPN-export ]
  Tables:
    L2VPN.l2vpn.0          : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Complete
LDP:
  Router ID: 10.69.105.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.105
  Route-distinguisher: 10.255.14.176:105
  Vrf-import: [ LDP-import ]
  Vrf-export: [ LDP-export ]
  Tables:
    LDP.inet.0             : 5 routes (4 active, 0 holddown, 0 hidden)
    Restart Complete
OSPF:
  Router ID: 10.69.101.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.101
  Route-distinguisher: 10.255.14.176:101
  Vrf-import: [ OSPF-import ]
  Vrf-export: [ OSPF-export ]
  Vrf-import-target: [ target:11111
  Tables:
    OSPF.inet.0            : 8 routes (7 active, 0 holddown, 0 hidden)
    Restart Complete
RIP:
  Router ID: 10.69.102.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.102
  Route-distinguisher: 10.255.14.176:102
  Vrf-import: [ RIP-import ]
  Vrf-export: [ RIP-export ]
```



```
  Tables:
    RIP.inet.0            : 6 routes (6 active, 0 holddown, 0 hidden)
    Restart Complete
STATIC:
  Router ID: 10.69.100.1
  Type: vrf               State: Active
  Restart State: Complete Path selection timeout: 300
  Interfaces:
    t3-0/0/0.100
  Route-distinguisher: 10.255.14.176:100
  Vrf-import: [ STATIC-import ]
  Vrf-export: [ STATIC-export ]
  Tables:
    STATIC.inet.0         : 4 routes (4 active, 0 holddown, 0 hidden)
    Restart Complete
```

show route instance detail (Graceful Restart Incomplete)

```
user@host> show route instance detail
master:
  Router ID: 10.255.14.176
  Type: forwarding        State: Active
  Restart State: Pending  Path selection timeout: 300
  Tables:
    inet.0                : 17 routes (15 active, 1 holddown, 1 hidden)
    Restart Pending: OSPF LDP
    inet.3                : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Pending: OSPF LDP
    iso.0                 : 1 routes (1 active, 0 holddown, 0 hidden)
    Restart Complete
    mpls.0                : 23 routes (23 active, 0 holddown, 0 hidden)
    Restart Pending: LDP VPN
    bgp.l3vpn.0           : 10 routes (10 active, 0 holddown, 0 hidden)
    Restart Pending: BGP VPN
    inet6.0               : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Complete
    bgp.l2vpn.0           : 1 routes (1 active, 0 holddown, 0 hidden)
    Restart Pending: BGP VPN
BGP-INET:
  Router ID: 10.69.103.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.103
  Route-distinguisher: 10.255.14.176:103
  Vrf-import: [ BGP-INET-import ]
  Vrf-export: [ BGP-INET-export ]
  Tables:
    BGP-INET.inet.0       : 6 routes (5 active, 0 holddown, 0 hidden)
    Restart Pending: VPN
BGP-L:
  Router ID: 10.69.104.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.104
  Route-distinguisher: 10.255.14.176:104
  Vrf-import: [ BGP-L-import ]
  Vrf-export: [ BGP-L-export ]
  Tables:
    BGP-L.inet.0          : 6 routes (5 active, 0 holddown, 0 hidden)
```



```
    Restart Pending: VPN
    BGP-L.mpls.0          : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Pending: VPN
L2VPN:
  Router ID: 0.0.0.0
  Type: l2vpn             State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.512
  Route-distinguisher: 10.255.14.176:512
  Vrf-import: [ L2VPN-import ]
  Vrf-export: [ L2VPN-export ]
  Tables:
    L2VPN.l2vpn.0         : 2 routes (2 active, 0 holddown, 0 hidden)
    Restart Pending: VPN L2VPN
LDP:
  Router ID: 10.69.105.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.105
  Route-distinguisher: 10.255.14.176:105
  Vrf-import: [ LDP-import ]
  Vrf-export: [ LDP-export ]
  Tables:
    LDP.inet.0            : 5 routes (4 active, 1 holddown, 0 hidden)
    Restart Pending: OSPF LDP VPN
OSPF:
  Router ID: 10.69.101.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.101
  Route-distinguisher: 10.255.14.176:101
  Vrf-import: [ OSPF-import ]
  Vrf-export: [ OSPF-export ]
  Tables:
    OSPF.inet.0           : 8 routes (7 active, 1 holddown, 0 hidden)
    Restart Pending: OSPF VPN
RIP:
  Router ID: 10.69.102.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.102
  Route-distinguisher: 10.255.14.176:102
  Vrf-import: [ RIP-import ]
  Vrf-export: [ RIP-export ]
  Tables:
    RIP.inet.0            : 8 routes (6 active, 2 holddown, 0 hidden)
    Restart Pending: RIP VPN
STATIC:
  Router ID: 10.69.100.1
  Type: vrf               State: Active
  Restart State: Pending  Path selection timeout: 300
  Interfaces:
    t3-0/0/0.100
  Route-distinguisher: 10.255.14.176:100
  Vrf-import: [ STATIC-import ]
  Vrf-export: [ STATIC-export ]
  Tables:
```



```
    STATIC.inet.0          : 4 routes (4 active, 0 holddown, 0 hidden)
    Restart Pending: VPN
```

show route instance detail (VPLS Routing Instance)

```
user@host> show route instance detail test-vpls
test-vpls:
  Router ID: 0.0.0.0
  Type: vpls             State: Active
  Interfaces:
    lsi.1048833
    lsi.1048832
    fe-0/1/0.513
  Route-distinguisher: 10.255.37.65:1
  Vrf-import: [ __vrf-import-test-vpls-internal__ ]
  Vrf-export: [ __vrf-export-test-vpls-internal__ ]
  Vrf-import-target: [ target:300:1 ]
  Vrf-export-target: [ target:300:1 ]
  Vrf-edge-protection-id:  166.1.3.1  Fast-reroute-priority: high
  Tables:
    test-vpls.l2vpn.0          : 3 routes (3 active, 0 holddown, 0 hidden)
```

show route instance operational

```
user@host> show route instance operational
Operational Routing Instances:

master
default
```

show route instance summary

```
user@host> show route instance summary
Instance           Type          Primary rib           Active/holddown/hidden
master             forwarding
                                 inet.0                15/0/1
                                 iso.0                 1/0/0
                                 mpls.0                35/0/0
                                 l3vpn.0               0/0/0
                                 inet6.0               2/0/0
                                 l2vpn.0               0/0/0
                                 l2circuit.0           0/0/0
BGP-INET           vrf
                                 BGP-INET.inet.0       5/0/0
                                 BGP-INET.iso.0        0/0/0
                                 BGP-INET.inet6.0      0/0/0
BGP-L              vrf
                                 BGP-L.inet.0          5/0/0
                                 BGP-L.iso.0           0/0/0
                                 BGP-L.mpls.0          4/0/0
                                 BGP-L.inet6.0         0/0/0
L2VPN              l2vpn
                                 L2VPN.inet.0          0/0/0
                                 L2VPN.iso.0           0/0/0
                                 L2VPN.inet6.0         0/0/0
                                 L2VPN.l2vpn.0         2/0/0
LDP                vrf
                                 LDP.inet.0            4/0/0
                                 LDP.iso.0             0/0/0
                                 LDP.mpls.0            0/0/0
```

```
                                    LDP.inet6.0            0/0/0
                                    LDP.l2circuit.0        0/0/0
OSPF              vrf
                                    OSPF.inet.0            7/0/0
                                    OSPF.iso.0             0/0/0
                                    OSPF.inet6.0           0/0/0
RIP               vrf
                                    RIP.inet.0             6/0/0
                                    RIP.iso.0              0/0/0
                                    RIP.inet6.0            0/0/0
STATIC            vrf
                                    STATIC.inet.0          4/0/0
                                    STATIC.iso.0           0/0/0
                                    STATIC.inet6.0         0/0/0
```


show route next-hop

List of Syntax

Syntax on page 1499
Syntax (EX Series Switches) on page 1499

Syntax

show route next-hop *next-hop*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route next-hop *next-hop*
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the entries in the routing table that are being sent to the specified next-hop address.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of ouput.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

next-hop—Next-hop address.

Required Privilege Level

view

List of Sample Output

show route next-hop on page 1499
show route next-hop detail on page 1500
show route next-hop extensive on page 1502
show route next-hop terse on page 1503

Output Fields

For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route next-hop

```
user@host> show route next-hop 192.168.71.254

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both

10.10.0.0/16       *[Static/5] 06:26:25
                    > to 192.168.71.254 via fxp0.0
10.209.0.0/16      *[Static/5] 06:26:25
```



```
                        > to 192.168.71.254 via fxp0.0
172.16.0.0/12          *[Static/5] 06:26:25
                        > to 192.168.71.254 via fxp0.0
192.168.0.0/16         *[Static/5] 06:26:25
                        > to 192.168.71.254 via fxp0.0
192.168.102.0/23       *[Static/5] 06:26:25
                        > to 192.168.71.254 via fxp0.0
207.17.136.0/24        *[Static/5] 06:26:25
                        > to 192.168.71.254 via fxp0.0
207.17.136.192/32      *[Static/5] 06:26:25
                        > to 192.168.71.254 via fxp0.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route next-hop detail

```
user@host> show route next-hop 192.168.71.254 detail

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
Restart Complete
10.10.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

10.209.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

172.16.0.0/12 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
```



```
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

192.168.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

192.168.102.0/23 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

207.17.136.0/24 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

207.17.136.192/32 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 36
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:     1
                Age: 6:27:41
                Task: RT
                Announcement bits (3): 0-KRT 3-Resolve tree 1 5-Resolve tree 2
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete
```

```
inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route next-hop extensive

```
user@host> show route next-hop 192.168.71.254 extensive

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
10.10.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.10.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.209.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.209.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

172.16.0.0/12 (1 entry, 1 announced)
TSI:
KRT in-kernel 172.16.0.0/12 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

192.168.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 192.168.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
```



```
192.168.102.0/23 (1 entry, 1 announced)
TSI:
KRT in-kernel 192.168.102.0/23 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

207.17.136.0/24 (1 entry, 1 announced)
TSI:
KRT in-kernel 207.17.136.0/24 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

207.17.136.192/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 207.17.136.192/32 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:02:28
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

inet6.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

green.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

red.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route next-hop terse

user@host> **show route next-hop 192.168.71.254 terse**

```
inet.0: 25 destinations, 26 routes (24 active, 0 holddown, 1 hidden)
Restart Complete
+ = Active Route, - = Last Active, * = Both
```



```
A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.10.0.0/16       S   5                       >192.168.71.254
* 10.209.0.0/16      S   5                       >192.168.71.254
* 172.16.0.0/12      S   5                       >192.168.71.254
* 192.168.0.0/16     S   5                       >192.168.71.254
* 192.168.102.0/23   S   5                       >192.168.71.254
* 207.17.136.0/24    S   5                       >192.168.71.254
* 207.17.136.192/32  S   5                       >192.168.71.254

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

red.inet.0: 4 destinations, 5 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Restart Complete

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Restart Complete
private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route no-community

List of Syntax

Syntax on page 1505
Syntax (EX Series Switches) on page 1505

Syntax

show route no-community
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route no-community
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the route entries in each routing table that are not associated with any community.

Options

none—(Same as **brief**) Display the route entries in each routing table that are not associated with any community.

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | ***logical-system-name*****)**—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show route no-community on page 1505
show route no-community detail on page 1506
show route no-community extensive on page 1506
show route no-community terse on page 1507

Output Fields

For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route no-community

```
user@host> show route no-community
inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.10.0.0/16       *[Static/5] 00:36:27
                    > to 192.168.71.254 via fxp0.0
10.209.0.0/16      *[Static/5] 00:36:27
                    > to 192.168.71.254 via fxp0.0
10.255.71.52/32    *[Direct/0] 00:36:27
```

```
                    > via lo0.0
10.255.71.63/32    *[OSPF/10] 00:04:39, metric 1
                    > to 35.1.1.2 via ge-3/1/0.0
10.255.71.64/32    *[OSPF/10] 00:00:08, metric 2
                    > to 35.1.1.2 via ge-3/1/0.0
10.255.71.240/32   *[OSPF/10] 00:05:04, metric 2
                      via so-0/1/2.0
                    > via so-0/3/2.0
10.255.71.241/32   *[OSPF/10] 00:05:14, metric 1
                    > via so-0/1/2.0
10.255.71.242/32   *[OSPF/10] 00:05:19, metric 1
                    > via so-0/3/2.0
172.16.12.0/24        *[OSPF/10] 00:05:14, metric 2
                    > via so-0/3/2.0
172.16.14.0/24        *[OSPF/10] 00:00:08, metric 3
                    > to 35.1.1.2 via ge-3/1/0.0
                      via so-0/1/2.0
                      via so-0/3/2.0
172.16.16.0/24        *[OSPF/10] 00:05:14, metric 2
                    > via so-0/1/2.0
.....
```

show route no-community detail

```
user@host> show route no-community detail

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
10.10.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Age: 38:08
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.209.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Age: 38:08
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
....
```

show route no-community extensive

```
user@host> show route no-community extensive

inet.0: 18 destinations, 18 routes (17 active, 0 holddown, 1 hidden)
10.10.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.10.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
```



```
                Local AS:    69
                Age: 2:03:33
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.209.0.0/16 (1 entry, 1 announced)
TSI:
KRT in-kernel 10.209.0.0/16 -> {192.168.71.254}
        *Static Preference: 5
                Next-hop reference count: 22
                Next hop: 192.168.71.254 via fxp0.0, selected
                State: <Active NoReadvrt Int Ext>
                Local AS:    69
                Age: 2:03:33
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I
```

show route no-community terse

user@host> **show route no-community terse**

```
inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.10.0.0/16       S   5                       >192.168.71.254
* 10.209.0.0/16      S   5                       >192.168.71.254
* 10.255.71.52/32    D   0                       >lo0.0
* 10.255.71.63/32    O  10          1            >35.1.1.2
* 10.255.71.64/32    O  10          2            >35.1.1.2
* 10.255.71.240/32   O  10          2             so-0/1/2.0
                                                 >so-0/3/2.0
* 10.255.71.241/32   O  10          1            >so-0/1/2.0
* 10.255.71.242/32   O  10          1            >so-0/3/2.0
* 172.16.12.0/24        O  10          2            >so-0/3/2.0
* 172.16.14.0/24        O  10          3            >35.1.1.2
                                                  so-0/1/2.0
                                                  so-0/3/2.0
* 172.16.16.0/24        O  10          2            >so-0/1/2.0
...
```

show route output

List of Syntax

Syntax on page 1508
Syntax (EX Series Switches) on page 1508

Syntax

show route output (address *ip-address* | interface *interface-name*)
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route output (address *ip-address* | interface *interface-name*)
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the entries in the routing table learned through static routes and interior gateway protocols that are to be sent out the interface with either the specified IP address or specified name.

To view routes advertised to a neighbor or received from a neighbor for the BGP protocol, use the **show route advertising-protocol bgp** and **show route receive-protocol bgp** commands instead.

Options

address ***ip-address***—Display entries in the routing table that are to be sent out the interface with the specified IP address.

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

interface ***interface-name***—Display entries in the routing table that are to be sent out the interface with the specified name.

logical-system (all | ***logical-system-name*****)**—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show route output address on page 1509
show route output address detail on page 1509
show route output address extensive on page 1510
show route output address terse on page 1510
show route output interface on page 1510
show route output interface detail on page 1511
show route output interface extensive on page 1511
show route output interface terse on page 1511

Output Fields For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route output address

user@host> **show route output address 172.16.36.1/24**

```
inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.36.0/24        *[Direct/0] 00:19:56
                    > via so-0/1/2.0
                    [OSPF/10] 00:19:55, metric 1
                    > via so-0/1/2.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route output address detail

user@host> **show route output address 172.16.36.1 detail**

```
inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
172.16.36.0/24 (2 entries, 0 announced)
        *Direct Preference: 0
                Next hop type: Interface
                Next-hop reference count: 1
                Next hop: via so-0/1/2.0, selected
                State: <Active Int>
                Age: 23:00
                Task: IF
                AS path: I
         OSPF   Preference: 10
                Next-hop reference count: 1
                Next hop: via so-0/1/2.0, selected
                State: <Int>
                Inactive reason: Route Preference
                Age: 22:59      Metric: 1
                Area: 0.0.0.0
                Task: OSPF
                AS path: I

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

```
private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route output address extensive

The output for the **show route output address extensive** command is identical to that of the **show route output address detail** command. For sample output, see show route output address detail on page 1509.

show route output address terse

```
user@host> show route output address 172.16.36.1 terse

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 172.16.36.0/24        D    0                      >so-0/1/2.0
                     O  10          1            >so-0/1/2.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route output interface

```
user@host> show route output interface so-0/1/2.0

inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.255.71.240/32   *[OSPF/10] 00:13:00, metric 2
                      via so-0/1/2.0
                    > via so-0/3/2.0
10.255.71.241/32   *[OSPF/10] 00:13:10, metric 1
                    > via so-0/1/2.0
172.16.14.0/24        *[OSPF/10] 00:05:11, metric 3
                      to 35.1.1.2 via ge-3/1/0.0
                    > via so-0/1/2.0
                      via so-0/3/2.0
172.16.16.0/24        *[OSPF/10] 00:13:10, metric 2
                    > via so-0/1/2.0
172.16.36.0/24        *[Direct/0] 00:13:21
                    > via so-0/1/2.0
                    [OSPF/10] 00:13:20, metric 1
                    > via so-0/1/2.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
```



```
inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route output interface detail

user@host> **show route output interface so-0/1/2.0 detail**

```
inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
10.255.71.240/32 (1 entry, 1 announced)
        *OSPF   Preference: 10
                Next-hop reference count: 2
                Next hop: via so-0/1/2.0
                Next hop: via so-0/3/2.0, selected
                State: <Active Int>
                Age: 14:52      Metric: 2
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (1): 0-KRT
                AS path: I

10.255.71.241/32 (1 entry, 1 announced)
        *OSPF   Preference: 10
                Next-hop reference count: 4
                Next hop: via so-0/1/2.0, selected
                State: <Active Int>
                Age: 15:02      Metric: 1
                Area: 0.0.0.0
                Task: OSPF
                Announcement bits (1): 0-KRT
                AS path: I
...
```

show route output interface extensive

The output for the **show route output interface extensive** command is identical to that of the **show route output interface detail** command. For sample output, see show route output interface detail on page 1511.

show route output interface terse

user@host> **show route output interface so-0/1/2.0 terse**

```
inet.0: 28 destinations, 30 routes (27 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* 10.255.71.240/32   O  10          2              so-0/1/2.0
                                                  >so-0/3/2.0
* 10.255.71.241/32   O  10          1             >so-0/1/2.0
* 172.16.14.0/24        O  10          3              35.1.1.2
                                                  >so-0/1/2.0
                                                   so-0/3/2.0
* 172.16.16.0/24        O  10          2              >so-0/1/2.0
* 172.16.36.0/24        D   0                         >so-0/1/2.0
                     O  10          1             >so-0/1/2.0

private1__.inet.0: 2 destinations, 3 routes (2 active, 0 holddown, 0 hidden)
```



```
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

private1__.inet6.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
```

show route protocol

List of Syntax

Syntax on page 1513
Syntax (EX Series Switches) on page 1513

Syntax

show route protocol *protocol*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route protocol *protocol*
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
ospf2 and **ospf3** options introduced in Junos OS Release 9.2.
ospf2 and **ospf3** options introduced in Junos OS Release 9.2 for EX Series switches.
flow option introduced in Junos OS Release 10.0.
flow option introduced in Junos OS Release 10.0 for EX Series switches.

Description

Display the route entries in the routing table that were learned from a particular protocol.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output. If you do not specify a level of output, the system defaults to **brief**.

logical-system (all | ***logical-system-name*****)**—(Optional) Perform this operation on all logical systems or on a particular logical system.

protocol—Protocol from which the route was learned:

- **access**—Access route for use by DHCP application
- **access-internal**—Access-internal route for use by DHCP application
- **aggregate**—Locally generated aggregate route
- **arp**—Route learned through the Address Resolution Protocol
- **atmvpn**—Asynchronous Transfer Mode virtual private network
- **bgp**—Border Gateway Protocol
- **ccc**—Circuit cross-connect
- **direct**—Directly connected route
- **dvmrp**—Distance Vector Multicast Routing Protocol
- **esis**—End System-to-Intermediate System
- **flow**—Locally defined flow-specification route
- **frr**—Precomputed protection route or backup route used when a link goes down
- **isis**—Intermediate System-to-Intermediate System

- **ldp**—Label Distribution Protocol
- **l2circuit**—Layer 2 circuit
- **l2vpn**—Layer 2 virtual private network
- **local**—Local address
- **mpls**—Multiprotocol Label Switching
- **msdp**—Multicast Source Discovery Protocol
- **ospf**—Open Shortest Path First versions 2 and 3
- **ospf2**—Open Shortest Path First versions 2 only
- **ospf3**—Open Shortest Path First version 3 only
- **pim**—Protocol Independent Multicast
- **rip**—Routing Information Protocol
- **ripng**—Routing Information Protocol next generation
- **rsvp**—Resource Reservation Protocol
- **rtarget**—Local route target virtual private network
- **static**—Statically defined route
- **tunnel**—Dynamic tunnel
- **vpn**—Virtual private network

NOTE: EX Series switches run a subset of these protocols. See the switch CLI for details.

Required Privilege Level view

List of Sample Output

show route protocol access on page 1515
show route protocol access-internal extensive on page 1515
show route protocol arp on page 1515
show route protocol bgp on page 1516
show route protocol bgp detail on page 1516
show route protocol bgp detail (Labeled Unicast) on page 1516
show route protocol bgp extensive on page 1517
show route protocol bgp terse on page 1518
show route protocol direct on page 1518
show route protocol frr on page 1519
show route protocol l2circuit detail on page 1519
show route protocol l2vpn extensive on page 1520
show route protocol ldp on page 1521

show route protocol ldp extensive on page 1521
show route protocol ospf (Layer 3 VPN) on page 1522
show route protocol ospf detail on page 1523
show route protocol rip on page 1523
show route protocol rip detail on page 1523
show route protocol ripng table inet6 on page 1524
show route protocol static detail on page 1524

Output Fields For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

show route protocol access

```
user@host> show route protocol access
inet.0: 30380 destinations, 30382 routes (30379 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

13.160.0.3/32      *[Access/13] 00:00:09
                    > to 13.160.0.2 via fe-0/0/0.0
13.160.0.4/32      *[Access/13] 00:00:09
                    > to 13.160.0.2 via fe-0/0/0.0
13.160.0.5/32      *[Access/13] 00:00:09
                    > to 13.160.0.2 via fe-0/0/0.0
```

show route protocol access-internal extensive

```
user@host> show route protocol access-internal 13.160.0.19 extensive
inet.0: 100020 destinations, 100022 routes (100019 active, 0 holddown, 1 hidden)
13.160.0.19/32 (1 entry, 1 announced)
TSI:
KRT in-kernel 13.160.0.19/32 -> {13.160.0.2}
        *Access-internal Preference: 12
                Next-hop reference count: 200000
                Next hop: 13.160.0.2 via fe-0/0/0.0, selected
                State: <Active Int>
       Age: 36
                Task: RPD Unix Domain Server./var/run/rpd_serv.local
                Announcement bits (1): 0-KRT
                AS path: I
```

show route protocol arp

```
user@host> show route protocol arp
inet.0: 43 destinations, 43 routes (42 active, 0 holddown, 1 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

cust1.inet.0: 1033 destinations, 2043 routes (1033 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

20.20.1.3/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.4/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
```



```
20.20.1.5/32        [ARP/4294967293] 00:04:32, from 20.20.1.1
                      Unusable
20.20.1.6/32        [ARP/4294967293] 00:04:34, from 20.20.1.1
                      Unusable
20.20.1.7/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.8/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.9/32        [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.10/32       [ARP/4294967293] 00:04:35, from 20.20.1.1
                      Unusable
20.20.1.11/32       [ARP/4294967293] 00:04:33, from 20.20.1.1
                      Unusable
20.20.1.12/32       [ARP/4294967293] 00:04:33, from 20.20.1.1
                      Unusable
20.20.1.13/32       [ARP/4294967293] 00:04:33, from 20.20.1.1
                      Unusable
...
```

show route protocol bgp

```
user@host> show route protocol bgp 192.168.64.0/21
inet.0: 335832 destinations, 335833 routes (335383 active, 0 holddown, 450 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.64.0/21       *[BGP/170] 6d 10:41:16, localpref 100, from 192.168.69.71
                      AS path: 10458 14203 2914 4788 4788 I
                    > to 192.168.167.254 via fxp0.0
```

show route protocol bgp detail

```
user@host> show route protocol bgp 66.117.63.0/24 detail
inet.0: 335805 destinations, 335806 routes (335356 active, 0 holddown, 450 hidden)
66.117.63.0/24  (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 1006436
                Source: 192.168.69.71
                Next hop type: Router, Next hop index: 324
                Next hop: 192.168.167.254 via fxp0.0, selected
                Protocol next hop: 192.168.69.71
                Indirect next hop: 8e166c0 342
                State: <Active Ext>
                Local AS:    69 Peer AS: 10458
                Age: 6d 10:42:42        Metric2: 0
                Task: BGP_10458.192.168.69.71+179
                Announcement bits (3): 0-KRT 2-BGP RT Background 3-Resolve tree
1
                AS path: 10458 14203 2914 4788 4788 I
                Communities: 2914:410 2914:2403 2914:3400
                Accepted
                Localpref: 100
                Router ID: 207.17.136.192
```

show route protocol bgp detail (Labeled Unicast)

```
user@host> show route protocol bgp 1.1.1.8/32 detail
inet.0: 45 destinations, 46 routes (45 active, 0 holddown, 0 hidden)
1.1.1.8/32 (2 entries, 2 announced)
```



```
State:
*BGP Preference: 1/-101
Next hop type: Indirect, Next hop index: 0
Address: 0xc007f30
Next-hop reference count: 2
Source: 1.1.1.1
Next hop type: Router, Next hop index: 614
Next hop: 20.1.1.2 via ge-0/0/1.0, selected
Label-switched-path lsp1
Label operation: Push 1000126, Push 1000125, Push 1000124, Push 1000123, Push
299872(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl, prop-ttl, prop-ttl(top)
Load balance label: Label 1000126: None; Label 1000125: None; Label 1000124: None;
 Label 1000123: None; Label 299872: None;
Label element ptr: 0xc007860
Label parent element ptr: 0xc0089a0
Label element references: 1
Label element child references: 0
Label element lsp id: 0
Session Id: 0x140
Protocol next hop: 1.1.1.4
Label operation: Push 1000126, Push 1000125, Push 1000124, Push 1000123(top)
Label TTL action: prop-ttl, prop-ttl, prop-ttl, prop-ttl
Load balance label: Label 1000126: None; Label 1000125: None; Label 1000124: None;
 Label 1000123: None;
Indirect next hop: 0xae8d300 1048576 INH Session ID: 0x142
State:
Local AS: 5 Peer AS: 5
Age: 22:43 Metric2: 2
Validation State: unverified
Task: BGP_5.1.1.1.1
Announcement bits (2): 0-KRT 7-Resolve tree 2
AS path: I
Accepted
Route Labels: 1000123(top) 1000124 1000125 1000126
Localpref: 100
Router ID: 1.1.1.1
```

show route protocol bgp extensive

user@host> **show route protocol bgp 192.168.64.0/21 extensive**

```
inet.0: 335827 destinations, 335828 routes (335378 active, 0 holddown, 450 hidden)
192.168.64.0/21 (1 entry, 1 announced)
TSI:
KRT in-kernel 1.9.0.0/16 -> {indirect(342)}
Page 0 idx 1 Type 1 val db31a80
    Nexthop: Self
    AS path: [69] 10458 14203 2914 4788 4788 I
    Communities: 2914:410 2914:2403 2914:3400
Path 1.9.0.0 from 192.168.69.71 Vector len 4.  Val: 1
        *BGP    Preference: 170/-101
                Next hop type: Indirect
                Next-hop reference count: 1006502
                Source: 192.168.69.71
                Next hop type: Router, Next hop index: 324
                Next hop: 192.168.167.254 via fxp0.0, selected
                Protocol next hop: 192.168.69.71
                Indirect next hop: 8e166c0 342
                State: <Active Ext>
                Local AS:    69 Peer AS: 10458
```



```
                Age: 6d 10:44:45        Metric2: 0
                Task: BGP_10458.192.168.69.71+179
                Announcement bits (3): 0-KRT 2-BGP RT Background 3-Resolve tree
1
                AS path: 10458 14203 2914 4788 4788 I
                Communities: 2914:410 2914:2403 2914:3400
                Accepted
                Localpref: 100
                Router ID: 207.17.136.192
                Indirect next hops: 1
                        Protocol next hop: 192.168.69.71
                        Indirect next hop: 8e166c0 342
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 192.168.167.254 via fxp0.0
                        192.168.0.0/16 Originating RIB: inet.0
                          Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 192.168.167.254 via fxp0.0
```

show route protocol bgp terse

```
user@host> show route protocol bgp 192.168.64.0/21 terse

inet.0: 24 destinations, 32 routes (23 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

A Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
  192.168.64.0/21    B 170        100            >172.16.100.1    10023 21 I
```

show route protocol direct

```
user@host> show route protocol direct

inet.0: 335843 destinations, 335844 routes (335394 active, 0 holddown, 450 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.8.0/24         *[Direct/0] 17w0d 10:31:49
                    > via fe-1/3/1.0
10.255.165.1/32    *[Direct/0] 25w4d 04:13:18
                    > via lo0.0
172.16.30.0/24      *[Direct/0] 17w0d 23:06:26
                    > via fe-1/3/2.0
192.168.164.0/22   *[Direct/0] 25w4d 04:13:20
                    > via fxp0.0

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

47.0005.80ff.f800.0000.0108.0001.0102.5516.5001/152
                   *[Direct/0] 25w4d 04:13:21
                    > via lo0.0

inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2001:db8::10:255:165:1/128
                   *[Direct/0] 25w4d 04:13:21
                    > via lo0.0
fe80::2a0:a5ff:fe12:ad7/128
```

```
                    *[Direct/0] 25w4d 04:13:21
                     > via lo0.0
```

show route protocol frr

```
user@host> show route protocol frr
inet.0: 43 destinations, 43 routes (42 active, 0 holddown, 1 hidden)

inet.3: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

cust1.inet.0: 1033 destinations, 2043 routes (1033 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

20.20.1.3/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.3 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.4/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.4 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.5/32       *[FRR/200] 00:05:35, from 20.20.1.1
                    > to 20.20.1.5 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.6/32       *[FRR/200] 00:05:37, from 20.20.1.1
                    > to 20.20.1.6 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.7/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.7 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.8/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.8 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.9/32       *[FRR/200] 00:05:38, from 20.20.1.1
                    > to 20.20.1.9 via ge-4/1/0.0
                      to 10.10.15.1 via ge-0/2/4.0, Push 16, Push 299792(top)
20.20.1.10/32      *[FRR/200] 00:05:38, from 20.20.1.1
...
```

show route protocol l2circuit detail

```
user@host> show route protocol l2circuit detail

mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
100000 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via ge-2/0/0.0, selected
                Label operation: Pop       Offset: 4
                State: <Active Int>
                Local AS:    99
                Age: 9:52
                Task: Common L2 VC
                Announcement bits (1): 0-KRT
                AS path: I

ge-2/0/0.0 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via so-1/1/2.0 weight 1, selected
                Label-switched-path my-lsp
                Label operation: Push 100000, Push 100000(top)[0] Offset: -4
                Protocol next hop: 10.245.255.63
                Push 100000 Offset: -4
```

```
                 Indirect next hop: 86af0c0 298
                State: <Active Int>
                Local AS:    99
                Age: 9:52
                Task: Common L2 VC
                Announcement bits (2): 0-KRT 1-Common L2 VC
                AS path: I

l2circuit.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

10.245.255.63:CtrlWord:4:3:Local/96 (1 entry, 1 announced)
        *L2CKT  Preference: 7
                Next hop: via so-1/1/2.0 weight 1, selected
                Label-switched-path my-lsp
                Label operation: Push 100000[0]
                Protocol next hop: 10.245.255.63 Indirect next hop: 86af000 296
                State: <Active Int>
                Local AS:    99
                Age: 10:21
                Task: l2 circuit
                Announcement bits (1): 0-LDP
                AS path: I
                VC Label 100000, MTU 1500, VLAN ID 512
```

show route protocol l2vpn extensive

```
user@host> show route protocol l2vpn extensive

inet.0: 14 destinations, 15 routes (13 active, 0 holddown, 1 hidden)

inet.3: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 7 destinations, 7 routes (7 active, 0 holddown, 0 hidden)
800001 (1 entry, 1 announced)
TSI:
KRT in-kernel 800001 /36 -> {so-0/0/0.0}
        *L2VPN  Preference: 7
                Next hop: via so-0/0/0.0 weight 49087 balance 97%, selected
                Label operation: Pop       Offset: 4
                State: <Active Int>
                Local AS:    69
                Age: 7:48
                Task: Common L2 VC
                Announcement bits (1): 0-KRT
                AS path: I

so-0/0/0.0 (1 entry, 1 announced)
TSI:
KRT in-kernel so-0/0/0.0.0      /16 -> {indirect(288)}
        *L2VPN  Preference: 7
                Next hop: via so-0/0/1.0, selected
                Label operation: Push 800000 Offset: -4
                Protocol next hop: 10.255.14.220
                Push 800000 Offset: -4
                 Indirect next hop: 85142a0 288
                State: <Active Int>
                Local AS:    69
                Age: 7:48
                Task: Common L2 VC
```



```
Announcement bits (2): 0-KRT 1-Common L2 VC
AS path: I
Communities: target:69:1 Layer2-info: encaps:PPP,
control flags:2, mtu: 0
```

show route protocol ldp

```
user@host> show route protocol ldp
inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)

inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.16.1/32    *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0, Push 100000
192.168.17.1/32    *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0

private1__.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)

mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

100064             *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0, Pop
100064(S=0)        *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0, Pop
100080             *[LDP/9] 1d 23:03:35, metric 1
                    > via t1-4/0/0.0, Swap 100000
```

show route protocol ldp extensive

```
user@host> show route protocol ldp extensive
192.168.16.1/32 (1 entry, 1 announced)
        State: <FlashAll>
        *LDP    Preference: 9
                Next-hop reference count: 3
                Next hop: via t1-4/0/0.0, selected
                Label operation: Push 100000
                State: <Active Int>
                Local AS: 64500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (2): 0-Resolve tree 1 2-Resolve tree 2
                AS path: I

192.168.17.1/32 (1 entry, 1 announced)
        State: <FlashAll>
        *LDP    Preference: 9
                Next-hop reference count: 3
                Next hop: via t1-4/0/0.0, selected
                State: <Active Int>
                Local AS: 64500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (2): 0-Resolve tree 1 2-Resolve tree 2
                AS path: I

private1__.inet.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

```
mpls.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)

100064 (1 entry, 1 announced)
TSI:
KRT in-kernel 100064 /36 -> {t1-4/0/0.0}
        *LDP    Preference: 9
                Next-hop reference count: 2
                Next hop: via t1-4/0/0.0, selected
                State: <Active Int>
                Local AS: 64500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
                Prefixes bound to route: 192.168.17.1/32

100064(S=0) (1 entry, 1 announced)
TSI:
KRT in-kernel 100064 /40 -> {t1-4/0/0.0}
        *LDP    Preference: 9
                Next-hop reference count: 2
                Next hop: via t1-4/0/0.0, selected
                Label operation: Pop
                State: <Active Int>
                Local AS: 64500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I

100080 (1 entry, 1 announced)
TSI:
KRT in-kernel 100080 /36 -> {t1-4/0/0.0}
        *LDP    Preference: 9
                Next-hop reference count: 2
                Next hop: via t1-4/0/0.0, selected
                Label operation: Swap 100000
                State: <Active Int>
                Local AS: 64500
                Age: 1d 23:03:58        Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
                Prefixes bound to route: 192.168.16.1/32
```

show route protocol ospf (Layer 3 VPN)

```
user@host> show route protocol ospf
inet.0: 40 destinations, 40 routes (39 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

10.39.1.4/30       *[OSPF/10] 00:05:18, metric 4
                    > via t3-3/2/0.0
10.39.1.8/30        [OSPF/10] 00:05:18, metric 2
                    > via t3-3/2/0.0
10.255.14.171/32   *[OSPF/10] 00:05:18, metric 4
                    > via t3-3/2/0.0
10.255.14.179/32   *[OSPF/10] 00:05:18, metric 2
                    > via t3-3/2/0.0
172.16.233.5/32      *[OSPF/10] 20:25:55, metric 1
```



```
VPN-AB.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.39.1.16/30       [OSPF/10] 00:05:43, metric 1
                    > via so-0/2/2.0
10.255.14.173/32   *[OSPF/10] 00:05:43, metric 1
                    > via so-0/2/2.0
172.16.233.5/32      *[OSPF/10] 20:26:20, metric 1
```

show route protocol ospf detail

```
user@host> show route protocol ospf detail
VPN-AB.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.39.1.16/30 (2 entries, 0 announced)
         OSPF   Preference:  10
                Nexthop: via so-0/2/2.0, selected
                State: <Int>
                Inactive reason: Route Preference
                Age: 6:25       Metric: 1
                Area: 0.0.0.0
                Task: VPN-AB-OSPF
                AS path: I
                Communities: Route-Type:0.0.0.0:1:0

...
```

show route protocol rip

```
user@host> show route protocol rip
inet.0: 26 destinations, 27 routes (25 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

VPN-AB.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
10.255.14.177/32   *[RIP/100] 20:24:34, metric 2
                    > to 10.39.1.22 via t3-0/2/2.0
172.16.233.9/32      *[RIP/100] 00:03:59, metric 1
```

show route protocol rip detail

```
user@host> show route protocol rip detail
inet.0: 26 destinations, 27 routes (25 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

VPN-AB.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
10.255.14.177/32 (1 entry, 1 announced)
        *RIP    Preference: 100
                Nexthop: 10.39.1.22 via t3-0/2/2.0, selected
                State: <Active Int>
                Age: 20:25:02   Metric: 2
                Task: VPN-AB-RIPv2
                Announcement bits (2): 0-KRT 2-BGP.0.0.0.0+179
                AS path: I
                Route learned from 10.39.1.22 expires in 96 seconds
```

show route protocol ripng table inet6

```
user@host> show route protocol ripng table inet6
inet6.0: 4215 destinations, 4215 routes (4214 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

1111::1/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::2/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::3/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::4/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::5/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
1111::6/128        *[RIPng/100] 02:13:33, metric 2
                    > to fe80::2a0:a5ff:fe3d:56 via t3-0/2/0.0
```

show route protocol static detail

```
user@host> show route protocol static detail
inet.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
10.5.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 324
                Address: 0x9274010
                Next-hop reference count: 27
                Next hop: 192.168.187.126 via fxp0.0, selected
                Session Id: 0x0
                State: <Active NoReadvrt Int Ext>
                Age: 7w3d 21:24:25
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.10.0.0/16 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 324
                Address: 0x9274010
                Next-hop reference count: 27
                Next hop: 192.168.187.126 via fxp0.0, selected
                Session Id: 0x0
                State: <Active NoReadvrt Int Ext>
                Age: 7w3d 21:24:25
                Validation State: unverified
                Task: RT
                Announcement bits (1): 0-KRT
                AS path: I

10.13.10.0/23 (1 entry, 1 announced)
        *Static Preference: 5
                Next hop type: Router, Next hop index: 324
                Address: 0x9274010
                Next-hop reference count: 27
                Next hop: 192.168.187.126 via fxp0.0, selected
                Session Id: 0x0
                State: <Active NoReadvrt Int Ext>
                Age: 7w3d 21:24:25
```



```
Validation State: unverified
Task: RT
Announcement bits (1): 0-KRT
AS path: I
```

show route receive-protocol

List of Syntax

Syntax on page 1526
Syntax (EX Series Switches) on page 1526

Syntax

show route receive-protocol *protocol neighbor-address*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)

Syntax (EX Series Switches)

show route receive-protocol *protocol neighbor-address*
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display the routing information as it was received through a particular neighbor using a particular dynamic routing protocol.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

protocol neighbor-address—Protocol transmitting the route (**bgp**, **dvmrp**, **msdp**, **pim**, **rip**, or **ripng**) and address of the neighboring router from which the route entry was received.

Additional Information

The output displays the selected routes and the attributes with which they were received, but does not show the effects of import policy on the routing attributes.

Required Privilege Level

view

List of Sample Output

show route receive-protocol bgp on page 1529
show route receive-protocol bgp extensive on page 1529
show route receive-protocol bgp table extensive on page 1529
show route receive-protocol bgp logical-system extensive on page 1530
show route receive-protocol bgp detail (Layer 2 VPN) on page 1531
show route receive-protocol bgp extensive (Layer 2 VPN) on page 1531
show route receive-protocol bgp (Layer 3 VPN) on page 1532
show route receive-protocol bgp detail (Layer 3 VPN) on page 1532
show route receive-protocol bgp detail (Long-Lived Graceful Restart) on page 1533
show route receive-protocol bgp detail (Labeled Unicast) on page 1533
show route receive-protocol bgp extensive (Layer 3 VPN) on page 1534

Output Fields

Table 92 on page 1527 describes the output fields for the **show route receive-protocol** command. Output fields are listed in the approximate order in which they appear.

Table 92: show route receive-protocol Output Fields

Field Name	Field Description	Level of Output
routing-table-name	Name of the routing table—for example, inet.0.	All levels
number destinations	Number of destinations for which there are routes in the routing table.	All levels
number routes	Number of routes in the routing table and total number of routes in the following states: • **active** • **holddown** (routes that are in pending state before being declared inactive) • **hidden** (routes that are not used because of a routing policy)	All levels
Prefix	Destination prefix.	none **brief**
MED	Multiple exit discriminator value included in the route.	none **brief**
destination-prefix (entry, announced)	Destination prefix. The **entry** value is the number of routes for this destination, and the **announced** value is the number of routes being announced for this destination.	**detail extensive**
Accepted LongLivedStale	The LongLivedStale flag indicates that the route was marked LLGR-stale by this router, as part of the operation of LLGR receiver mode. Either this flag or the LongLivedStaleImport flag may be displayed for a route. Neither of these flags are displayed at the same time as the Stale (ordinary GR stale) flag.	**detail extensive**
Accepted LongLivedStaleImport	The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy. Either this flag or the LongLivedStale flag may be displayed for a route. Neither of these flags are displayed at the same time as the Stale (ordinary GR stale) flag. Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and import into the inet.0 routing table	**detail extensive**
ImportAccepted LongLivedStaleImport	Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and imported into the inet.0 routing table The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy.	**detail extensive**
Route Distinguisher	64-bit prefix added to IP subnets to make them unique.	**detail extensive**
Label-Base, range	First label in a block of labels and label block size. A remote PE routing device uses this first label when sending traffic toward the advertising PE routing device.	**detail extensive**
VPN Label	Virtual private network (VPN) label. Packets are sent between CE and PE routing devices by advertising VPN labels. VPN labels transit over either an RSVP or an LDP label-switched path (LSP) tunnel.	**detail extensive**

Table 92: show route receive-protocol Output Fields *(continued)*

Field Name	Field Description	Level of Output
Next hop	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route.	All levels
Localpref or Lclpref	Local preference value included in the route.	All levels
AS path	Autonomous system (AS) path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the number that precedes the AS path. This number represents the number of ASs present in the AS path, when calculated as defined in RFC 4271. This value is used the AS-path merge process, as defined in RFC 4893. • **[ ]**—If more than one AS number is configured on the router, or if AS path prepending is configured, brackets enclose the local AS number associated with the AS path. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.	All levels
Route Labels	Stack of labels carried in the BGP route update.	**detail extensive**
Cluster list	(For route reflected output only) Cluster ID sent by the route reflector.	**detail extensive**
Originator ID	(For route reflected output only) Address of routing device that originally sent the route to the route reflector.	**detail extensive**
Communities	Community path attribute for the route. See the Output Field table in the show route detail command for all possible values for this field.	**detail extensive**
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.	**detail extensive**
Attrset AS	Number, local preference, and path of the AS that originated the route. These values are stored in the **Attrset** attribute at the originating routing device.	**detail extensive**
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).	**detail extensive**
control flags	Control flags: **none** or **Site Down**.	**detail extensive**

Table 92: show route receive-protocol Output Fields *(continued)*

Field Name	Field Description	Level of Output
mtu	Maximum transmission unit (MTU) of the Layer 2 circuit.	**detail extensive**

Sample Output

show route receive-protocol bgp

```
user@host> show route receive-protocol bgp 10.255.245.215

inet.0: 28 destinations, 33 routes (27 active, 0 holddown, 1 hidden)
Prefix                   Next hop              MED     Lclpref    AS path
10.22.1.0/24             10.255.245.215       0       100         I
10.22.2.0/24             10.255.245.215       0       100         I
```

show route receive-protocol bgp extensive

```
user@host> show route receive-protocol bgp 10.255.245.63 extensive
inet.0: 244 destinations, 244 routes (243 active, 0 holddown, 1 hidden)
Prefix             Next hop                MED    Lclpref AS path
172.16.1.0/24 (1 entry, 1 announced)
     Next hop: 10.0.50.3
     Localpref: 100
     AS path: I <Orginator>
     Cluster list:  10.2.3.1
     Originator ID: 10.255.245.45
172.16.163.0/16 (1 entry, 1 announced)
     Next hop: 111.222.5.254
     Localpref: 100
     AS path: I <Originator>
     Cluster list:  10.2.3.1
     Originator ID: 10.255.245.68
172.16.164.0/16 (1 entry, 1 announced)
     Next hop: 111.222.5.254
     Localpref: 100
     AS path: I <Originator>
     Cluster list:  10.2.3.1
     Originator ID: 10.255.245.45
172.16.195.0/24 (1 entry, 1 announced)
     Next hop: 111.222.5.254
     Localpref: 100
     AS path: I <Originator>
     Cluster list:  10.2.3.1
     Originator ID: 10.255.245.68
inet.2: 63 destinations, 63 routes (63 active, 0 holddown, 0 hidden)
Prefix             Next hop                MED    Lclpref AS path
inet.3: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
Prefix             Next hop                MED    Lclpref AS path
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix             Next hop                MED    Lclpref AS path
mpls.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
```

show route receive-protocol bgp table extensive

```
user@host> show route receive-protocol bgp 207.17.136.192 table inet.0 66.117.68.0/24 extensive
```



```
inet.0: 227315 destinations, 227316 routes (227302 active, 0 holddown, 13 hidden)
* 66.117.63.0/24 (1 entry, 1 announced)
     Nexthop: 207.17.136.29
     Localpref: 100
     AS path: AS2 PA[6]: 14203 2914 3356 29748 33437 AS_TRANS
     AS path: AS4 PA[2]: 33437 393219
     AS path: Merged[6]: 14203 2914 3356 29748 33437 393219 I
     Communities: 2914:420
```

show route receive-protocol bgp logical-system extensive

```
user@host> show route receive-protocol bgp 10.0.0.9 logical-system PE4 extensive
inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)
* 10.0.0.0/30 (1 entry, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

* 10.0.0.4/30 (1 entry, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

  10.0.0.8/30 (2 entries, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

* 10.9.9.1/32 (1 entry, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

* 10.100.1.1/32 (1 entry, 1 announced)
     Accepted
     Route Label: 3
     Nexthop: 10.0.0.9
     AS path: 13979 I

* 172.16.44.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 300096
     Nexthop: 10.0.0.9
     AS path: 13979 I
     AIGP: 203

* 172.16.55.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 300112
     Nexthop: 10.0.0.9
     AS path: 13979 7018 I
     AIGP: 25

* 172.16.66.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 300144
     Nexthop: 10.0.0.9
```



```
     AS path: 13979 7018 I

* 172.16.99.0/24 (1 entry, 1 announced)
     Accepted
     Route Label: 300160
     Nexthop: 10.0.0.9
     AS path: 13979 7018 I
```

show route receive-protocol bgp detail (Layer 2 VPN)

```
user@host> show route receive-protocol bgp 10.255.14.171 detail
inet.0: 68 destinations, 68 routes (67 active, 0 holddown, 1 hidden)
Prefix             Nexthop                MED    Lclpref AS path
inet.3: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Prefix             Nexthop                MED    Lclpref AS path
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix             Nexthop                MED    Lclpref AS path
mpls.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
Prefix             Nexthop                MED    Lclpref AS path
frame-vpn.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0
hidden)
Prefix             Nexthop                MED    Lclpref AS path
10.255.245.35:1:5:1/96 (1 entry, 1 announced)
     Route Distinguisher: 10.255.245.35:1
     Label-base : 800000, range : 4, status-vector : 0x0
     Nexthop: 10.255.245.35
     Localpref: 100
     AS path: I
     Communities: target:65299:100 Layer2-info: encaps:FRAME RELAY,
     control flags: 0, mtu: 0
bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix             Nexthop                MED    Lclpref AS path
10.255.245.35:1:5:1/96 (1 entry, 0 announced)
     Route Distinguisher: 10.255.245.35:1
     Label-base : 800000, range : 4, status-vector : 0x0
     Nexthop: 10.255.245.35
     Localpref: 100
     AS path: I
     Communities: target:65299:100 Layer2-info: encaps:FRAME RELAY,
     control flags:0, mtu: 0
```

show route receive-protocol bgp extensive (Layer 2 VPN)

```
user@host> show route receive-protocol bgp 10.255.14.171 extensive
inet.0: 68 destinations, 68 routes (67 active, 0 holddown, 1 hidden)
Prefix             Nexthop                MED    Lclpref AS path
inet.3: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
Prefix             Nexthop                MED    Lclpref AS path
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix             Nexthop                MED    Lclpref AS path
mpls.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
Prefix             Nexthop                MED    Lclpref AS path
frame-vpn.l2vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
Prefix             Nexthop                MED    Lclpref AS path
10.255.245.35:1:5:1/96 (1 entry, 1 announced)
     Route Distinguisher: 10.255.245.35:1
     Label-base : 800000, range : 4, status-vector : 0x0
     Nexthop: 10.255.245.35
     Localpref: 100
     AS path: I
```

```
     Communities: target:65299:100 Layer2-info: encaps:FRAME RELAY,
     control flags:0, mtu: 0
bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Prefix            Nexthop               MED    Lclpref AS path
10.255.245.35:1:5:1/96 (1 entry, 0 announced)
     Route Distinguisher: 10.255.245.35:1
     Label-base : 800000, range : 4, status-vector : 0x0
     Nexthop: 10.255.245.35
     Localpref: 100
     AS path: I
     Communities: target:65299:100 Layer2-info: encaps:FRAME RELAY,
     control flags:0, mtu: 0
```

show route receive-protocol bgp (Layer 3 VPN)

```
user@host> show route receive-protocol bgp 10.255.14.171
      inet.0: 33 destinations, 33 routes (32 active, 0 holddown, 1 hidden)
      Prefix            Nexthop               MED    Lclpref AS path
      inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
      Prefix            Nexthop               MED    Lclpref AS path
      VPN-A.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
      Prefix            Nexthop               MED    Lclpref AS path
      10.255.14.175/32   10.255.14.171                     100 2 I
      10.255.14.179/32   10.255.14.171           2         100 I
      VPN-B.inet.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
      Prefix            Nexthop               MED    Lclpref AS path
      10.255.14.175/32   10.255.14.171                     100 2 I
      10.255.14.177/32   10.255.14.171                     100 I
      iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
      Prefix            Nexthop               MED    Lclpref AS path
      mpls.0: 9 destinations, 9 routes (9 active, 0 holddown, 0 hidden)
      Prefix            Nexthop               MED    Lclpref AS path
      bgp.l3vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
      Prefix            Nexthop               MED    Lclpref AS path
      10.255.14.171:300:10.255.14.177/32
                         10.255.14.171                     100 I
      10.255.14.171:100:10.255.14.179/32
                         10.255.14.171           2         100 I
      10.255.14.171:200:10.255.14.175/32
                         10.255.14.171                     100 2 I
```

show route receive-protocol bgp detail (Layer 3 VPN)

```
user@host> show route receive-protocol bgp 10.255.14.174 detail
inet.0: 16 destinations, 17 routes (15 active, 0 holddown, 1 hidden)
inet.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
vpna.inet.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
* 10.49.0.0/30 (1 entry, 1 announced)
     Route Distinguisher: 10.255.14.176:2
     VPN Label: 101264
     Nexthop: 10.255.14.174
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AttrSet AS: 100
         Localpref: 100
         AS path: I
* 10.255.14.172/32 (1 entry, 1 announced)
     Route Distinguisher: 10.255.14.176:2
     VPN Label: 101280
```



```
     Nexthop: 10.255.14.174
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AttrSet AS: 100
         Localpref: 100
         AS path: I
iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
mpls.0: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
* 10.255.14.174:2:10.49.0.0/30 (1 entry, 0 announced)
     Route Distinguisher: 10.255.14.174:2
     VPN Label: 101264
     Nexthop: 10.255.14.174
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AttrSet AS: 100
         Localpref: 100
         AS path: I
* 10.255.14.174:2:10.255.14.172/32 (1 entry, 0 announced)
     Route Distinguisher: 10.255.14.174:2
     VPN Label: 101280
     Nexthop: 10.255.14.174
     Localpref: 100
     AS path: I
     Communities: target:200:100
     AttrSet AS: 100
         Localpref: 100
         AS path: I
inet6.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
```

show route receive-protocol bgp detail (Long-Lived Graceful Restart)

```
user@host> show route receive-protocol bgp 10.4.12.11 detail

  bgp.l2vpn.0: 38 destinations, 39 routes (37 active, 0 holddown, 1 hidden)
  * 172.16.1.4:100:172.16.1.4/96 AD (1 entry, 1 announced)
    Accepted LongLivedStale LongLivedStaleImport
    Nexthop: 10.4.12.11
    Localpref: 100
    AS path: I
```

show route receive-protocol bgp detail (Labeled Unicast)

```
user@host> show route receive-protocol bgp 1.1.1.1 detail
inet.0: 45 destinations, 46 routes (45 active, 0 holddown, 0 hidden)
* 1.1.1.8/32 (2 entries, 2 announced)
Accepted
Route Labels: 1000123(top) 1000124 1000125 1000126
Nexthop: 1.1.1.4
Localpref: 100
AS path: I
Entropy label capable, next hop field matches route next hop

inet.3: 15 destinations, 21 routes (6 active, 0 holddown, 14 hidden)

iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)

mpls.0: 11 destinations, 11 routes (11 active, 0 holddown, 0 hidden)
```

```
inet6.0: 26 destinations, 28 routes (26 active, 0 holddown, 0 hidden)

* 100::1/128 (2 entries, 2 announced)
Accepted
Route Labels: 1000123(top) 1000124 1000125 1000126
Nexthop: ::ffff:1.1.1.4
Localpref: 100
AS path: I

inet6.3: 22 destinations, 23 routes (22 active, 0 holddown, 0 hidden)
```

show route receive-protocol bgp extensive (Layer 3 VPN)

```
user@host> show route receive-protocol bgp 10.255.245.63 extensive
inet.0: 244 destinations, 244 routes (243 active, 0 holddown, 1 hidden)
     Prefix             Nexthop               MED    Lclpref AS path
     172.16.1.0/24 (1 entry, 1 announced)
          Nexthop: 10.0.50.3
          Localpref: 100
          AS path: I <Originator>
          Cluster list:  10.2.3.1
          Originator ID: 10.255.245.45
     172.16.163.0/16 (1 entry, 1 announced)
          Nexthop: 111.222.5.254
          Localpref: 100
          AS path: I <Originator>
          Cluster list:  10.2.3.1
          Originator ID: 10.255.245.68
     172.16.164.0/16 (1 entry, 1 announced)
          Nexthop: 111.222.5.254
          Localpref: 100
          AS path: I <Originator>
          Cluster list:  10.2.3.1
          Originator ID: 10.255.245.45
     172.16.195.0/24 (1 entry, 1 announced)
          Nexthop: 111.222.5.254
          Localpref: 100
          AS path: I <Originator>
          Cluster list:  10.2.3.1
          Originator ID: 10.255.245.68
     inet.2: 63 destinations, 63 routes (63 active, 0 holddown, 0 hidden)
     Prefix             Nexthop               MED    Lclpref AS path
     inet.3: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
     Prefix             Nexthop               MED    Lclpref AS path
     iso.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
     Prefix             Nexthop               MED    Lclpref AS path
     mpls.0: 48 destinations, 48 routes (48 active, 0 holddown, 0 hidden)
```


show route table

List of Syntax

Syntax on page 1535
Syntax (EX Series Switches and QFX Series Switches) on page 1535

Syntax

show route table *routing-table-name*
<brief | detail | extensive | terse>
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches and QFX Series Switches)

show route table *routing-table-name*
<brief | detail | extensive | terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 14.1X53-D15 for QFX Series switches.
Show route table evpn statement introduced in Junos OS Release 15.1X53-D30 for QFX Series switches.

Description

Display the route entries in a particular routing table.

Options

brief | detail | extensive | terse—(Optional) Display the specified level of output.

logical-system (all | ***logical-system-name*****)**—(Optional) Perform this operation on all logical systems or on a particular logical system.

routing-table-name—Display route entries for all routing tables whose names begin with this string (for example, inet.0 and inet6.0 are both displayed when you run the **show route table inet** command).

Required Privilege Level

view

Related Documentation

- *show route summary*

List of Sample Output

show route table bgp.l2.vpn on page 1546
show route table bgp.l3vpn.0 on page 1546
show route table bgp.l3vpn.0 detail on page 1547
show route table bgp.rtarget.0 (When Proxy BGP Route Target Filtering Is Configured) on page 1548
show route table bgp.evpn.0 on page 1548
show route table evpna.evpn.0 on page 1549
show route table inet.0 on page 1549
show route table inet.3 on page 1550
show route table inet.3 protocol ospf on page 1550
show route table inet6.0 on page 1550
show route table inet6.3 on page 1550

Output Fields Table 80 on page 1372 describes the output fields for the **show route table** command. Output fields are listed in the approximate order in which they appear.

Table 93: show route table Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).

Restart complete	All protocols have restarted for this routing table.
	
	Restart state:
	
	• **Pending:***protocol-name*—List of protocols that have not yet completed graceful restart for this routing table.
	• **Complete**—All protocols have restarted for this routing table.
	
	For example, if the output shows-
	
	• `LDP.inet.0             : 5 routes (4 active, 1 holddown, 0 hidden)`
	`    Restart Pending: OSPF LDP VPN`
	
	This indicates that **OSPF**, **LDP**, and **VPN** protocols did not restart for the **LDP.inet.0** routing table.
	• `vpls_1.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)`
	`Restart Complete`
	
	This indicates that all protocols have restarted for the **vpls_1.l2vpn.0** routing table.

number destinations	Number of destinations for which there are routes in the routing table.

Table 93: show route table Output Fields *(continued)*

<table>
<tr><th>Field Name</th><th>Field Description</th></tr>
<tr><td>number routes</td><td>Number of routes in the routing table and total number of routes in the following states:

active (routes that are active)
holddown (routes that are in the pending state before being declared inactive)
hidden (routes that are not used because of a routing policy)
</td></tr>
<tr><td>route-destination (entry, announced)</td><td>Route destination (for example:10.0.0.1/24). The entry value is the number of routes for this destination, and the announced value is the number of routes being announced for this destination. Sometimes the route destination is presented in another format, such as:

MPLS-label (for example, 80001).
interface-name (for example, ge-1/0/2).
neighbor-address:control-word-status:encapsulation type:vc-id:source (Layer 2 circuit only; for example, 10.1.1.195:NoCtrlWord:1:1:Local/96).

neighbor-address—Address of the neighbor.
control-word-status—Whether the use of the control word has been negotiated for this virtual circuit: NoCtrlWord or CtrlWord.
encapsulation type—Type of encapsulation, represented by a number: (1) Frame Relay DLCI, (2) ATM AAL5 VCC transport, (3) ATM transparent cell transport, (4) Ethernet, (5) VLAN Ethernet, (6) HDLC, (7) PPP, (8) ATM VCC cell transport, (10) ATM VPC cell transport.
vc-id—Virtual circuit identifier.
source—Source of the advertisement: Local or Remote.

inclusive multicast Ethernet tag route—Type of route destination represented by (for example, 3:100.100.100.10:100::0::10::100.100.100.10/384):

route distinguisher—(8 octets) Route distinguisher (RD) must be the RD of the EVPN instance (EVI) that is advertising the NLRI.
Ethernet tag ID—(4 octets) Identifier of the Ethernet tag. Can set to 0 or to a valid Ethernet tag value.
IP address length—(1 octet) Length of IP address in bits.
originating router's IP address—(4 or 16 octets) Must set to the provider edge (PE) device's IP address. This address should be common for all EVIs on the PE device, and may be the PE device's loopback address.

</td></tr>
<tr><td>label stacking</td><td>(Next-to-the-last-hop routing device for MPLS only) Depth of the MPLS label stack, where the label-popping operation is needed to remove one or more labels from the top of the stack. A pair of routes is displayed, because the pop operation is performed only when the stack depth is two or more labels.

S=0 route indicates that a packet with an incoming label stack depth of 2 or more exits this routing device with one fewer label (the label-popping operation is performed).
If there is no S= information, the route is a normal MPLS route, which has a stack depth of 1 (the label-popping operation is not performed).
</td></tr>
</table>

Table 93: show route table Output Fields *(continued)*

Field Name	Field Description
[*protocol*, *preference*]	Protocol from which the route was learned and the preference value for the route. • **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • **-** —A hyphen indicates the last active route. • *****—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route. In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the **LocalPref** value for Route 1 is 100, the **Preference2** value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value and a lower **Preference2** value.
Level	(IS-IS only). In IS-IS, a single AS can be divided into smaller groups called areas. Routing between areas is organized hierarchically, allowing a domain to be administratively divided into smaller areas. This organization is accomplished by configuring Level 1 and Level 2 intermediate systems. Level 1 systems route within an area. When the destination is outside an area, they route toward a Level 2 system. Level 2 intermediate systems route between areas and toward other ASs.
Route Distinguisher	IP subnet augmented with a 64-bit prefix.
PMSI	Provider multicast service interface (MVPN routing table).
Next-hop type	Type of next hop. For a description of possible values for this field, see Table 84 on page 1426.
Next-hop reference count	Number of references made to the next hop.
Flood nexthop branches exceed maximum message	Indicates that the number of flood next-hop branches exceeded the system limit of 32 branches, and only a subset of the flood next-hop branches were installed in the kernel.
Source	IP address of the route source.
Next hop	Network layer address of the directly reachable neighboring system.
via	Interface used to reach the next hop. If there is more than one interface available to the next hop, the name of the interface that is actually used is followed by the word **Selected**. This field can also contain the following information: • Weight—Value used to distinguish primary, secondary, and fast reroute backup routes. Weight information is available when MPLS label-switched path (LSP) link protection, node-link protection, or fast reroute is enabled, or when the standby state is enabled for secondary paths. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible. • Balance—Balance coefficient indicating how traffic of unequal cost is distributed among next hops when a routing device is performing unequal-cost load balancing. This information is available when you enable BGP multipath load balancing.
Label-switched-path *lsp-path-name*	Name of the LSP used to reach the next hop.

Table 93: show route table Output Fields *(continued)*

Field Name	Field Description
Label operation	MPLS label and operation occurring at this routing device. The operation can be **pop** (where a label is removed from the top of the stack), **push** (where another label is added to the label stack), or **swap** (where a label is replaced by another label).
Interface	(Local only) Local interface name.
Protocol next hop	Network layer address of the remote routing device that advertised the prefix. This address is used to derive a forwarding next hop.
Indirect next hop	Index designation used to specify the mapping between protocol next hops, tags, kernel export policy, and the forwarding next hops.
State	State of the route (a route can be in more than one state). See Table 85 on page 1428.
Local AS	AS number of the local routing devices.
Age	How long the route has been known.
AIGP	Accumulated interior gateway protocol (AIGP) BGP attribute.
Metric*n*	Cost value of the indicated route. For routes within an AS, the cost is determined by IGP and the individual protocol metrics. For external routes, destinations, or routing domains, the cost is determined by a preference value.
MED-plus-IGP	Metric value for BGP path selection to which the IGP cost to the next-hop destination has been added.
TTL-Action	For MPLS LSPs, state of the TTL propagation attribute. Can be enabled or disabled for all RSVP-signaled and LDP-signaled LSPs or for specific VRF routing instances.
Task	Name of the protocol that has added the route.
Announcement bits	The number of BGP peers or protocols to which Junos OS has announced this route, followed by the list of the recipients of the announcement. Junos OS can also announce the route to the kernel routing table (KRT) for installing the route into the Packet Forwarding Engine, to a resolve tree, a Layer 2 VC, or even a VPN. For example, ***n*-Resolve inet** indicates that the specified route is used for route resolution for next hops found in the routing table. • *n*—An index used by Juniper Networks customer support only.

Table 93: show route table Output Fields *(continued)*

Field Name	Field Description
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **Recorded**—The AS path is recorded by the sample process (sampled). • **?**—Incomplete; typically, the AS path was aggregated. When AS path numbers are included in the route, the format is as follows: • **[ ]**—Brackets enclose the number that precedes the AS path. This number represents the number of ASs present in the AS path, when calculated as defined in RFC 4271. This value is used in the AS-path merge process, as defined in RFC 4893. • **[ ]**—If more than one AS number is configured on the routing device, or if AS path prepending is configured, brackets enclose the local AS number associated with the AS path. • **{ }**—Braces enclose AS sets, which are groups of AS numbers in which the order does not matter. A set commonly results from route aggregation. The numbers in each AS set are displayed in ascending order. • **( )**—Parentheses enclose a confederation. • **([ ])**—Parentheses and brackets enclose a confederation set. NOTE: In Junos OS Release 10.3 and later, the AS path field displays an unrecognized attribute and associated hexadecimal value if BGP receives attribute 128 (attribute set) and you have not configured an independent domain in any routing instance.
validation-state	(BGP-learned routes) Validation status of the route: • **Invalid**—Indicates that the prefix is found, but either the corresponding AS received from the EBGP peer is not the AS that appears in the database, or the prefix length in the BGP update message is longer than the maximum length permitted in the database. • **Unknown**—Indicates that the prefix is not among the prefixes or prefix ranges in the database. • **Unverified**—Indicates that the origin of the prefix is not verified against the database. This is because the database got populated and the validation is not called for in the BGP import policy, although origin validation is enabled, or the origin validation is not enabled for the BGP peers. • **Valid**—Indicates that the prefix and autonomous system pair are found in the database.
FECs bound to route	Indicates point-to-multipoint root address, multicast source address, and multicast group address when multipoint LDP (M-LDP) inband signaling is configured.
Primary Upstream	When multipoint LDP with multicast-only fast reroute (MoFRR) is configured, indicates the primary upstream path. MoFRR transmits a multicast join message from a receiver toward a source on a primary path, while also transmitting a secondary multicast join message from the receiver toward the source on a backup path.
RPF Nexthops	When multipoint LDP with MoFRR is configured, indicates the reverse-path forwarding (RPF) next-hop information. Data packets are received from both the primary path and the secondary paths. The redundant packets are discarded at topology merge points due to the RPF checks.
Label	Multiple MPLS labels are used to control MoFRR stream selection. Each label represents a separate route, but each references the same interface list check. Only the primary label is forwarded while all others are dropped. Multiple interfaces can receive packets using the same label.

Table 93: show route table Output Fields *(continued)*

Field Name	Field Description
weight	Value used to distinguish MoFRR primary and backup routes. A lower weight value is preferred. Among routes with the same weight value, load balancing is possible.
VC Label	MPLS label assigned to the Layer 2 circuit virtual connection.
MTU	Maximum transmission unit (MTU) of the Layer 2 circuit.
VLAN ID	VLAN identifier of the Layer 2 circuit.
Prefixes bound to route	Forwarding equivalent class (FEC) bound to this route. Applicable only to routes installed by LDP.
Communities	Community path attribute for the route. See Table 86 on page 1430 for all possible values for this field.
Layer2-info: encaps	Layer 2 encapsulation (for example, VPLS).
control flags	Control flags: **none** or **Site Down**.
mtu	Maximum transmission unit (MTU) information.
Label-Base, range	First label in a block of labels and label block size. A remote PE routing device uses this first label when sending traffic toward the advertising PE routing device.
status vector	Layer 2 VPN and VPLS network layer reachability information (NLRI).
Accepted Multipath	Current active path when BGP multipath is configured.
Accepted LongLivedStale	The LongLivedStale flag indicates that the route was marked LLGR-stale by this router, as part of the operation of LLGR receiver mode. Either this flag or the LongLivedStaleImport flag might be displayed for a route. Neither of these flags is displayed at the same time as the Stale (ordinary GR stale) flag.
Accepted LongLivedStaleImport	The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy. Either this flag or the LongLivedStale flag might be displayed for a route. Neither of these flags is displayed at the same time as the Stale (ordinary GR stale) flag. Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and import into the inet.0 routing table
ImportAccepted LongLivedStaleImport	Accept all received BGP long-lived graceful restart (LLGR) and LLGR stale routes learned from configured neighbors and imported into the inet.0 routing table The LongLivedStaleImport flag indicates that the route was marked LLGR-stale when it was received from a peer, or by import policy.
Accepted MultipathContrib	Path currently contributing to BGP multipath.
Localpref	Local preference value included in the route.
Router ID	BGP router ID as advertised by the neighbor in the open message.

Table 93: show route table Output Fields *(continued)*

Field Name	Field Description
Primary Routing Table	In a routing table group, the name of the primary routing table in which the route resides.
Secondary Tables	In a routing table group, the name of one or more secondary tables in which the route resides.

Table 84 on page 1426 describes all possible values for the Next-hop Types output field.

Table 94: Next-hop Types Output Field Values

Next-Hop Type	Description
Broadcast (bcast)	Broadcast next hop.
Deny	Deny next hop.
Discard	Discard next hop.
Flood	Flood next hop. Consists of components called branches, up to a maximum of 32 branches. Each flood next-hop branch sends a copy of the traffic to the forwarding interface. Used by point-to-multipoint RSVP, point-to-multipoint LDP, point-to-multipoint CCC, and multicast.
Hold	Next hop is waiting to be resolved into a unicast or multicast type.
Indexed (idxd)	Indexed next hop.
Indirect (indr)	Used with applications that have a protocol next hop address that is remote. You are likely to see this next-hop type for internal BGP (IBGP) routes when the BGP next hop is a BGP neighbor that is not directly connected.
Interface	Used for a network address assigned to an interface. Unlike the router next hop, the interface next hop does not reference any specific node on the network.
Local (locl)	Local address on an interface. This next-hop type causes packets with this destination address to be received locally.
Multicast (mcst)	Wire multicast next hop (limited to the LAN).
Multicast discard (mdsc)	Multicast discard.
Multicast group (mgrp)	Multicast group member.
Receive (recv)	Receive.
Reject (rjct)	Discard. An ICMP unreachable message was sent.

Table 94: Next-hop Types Output Field Values *(continued)*

Next-Hop Type	Description
Resolve (rslv)	Resolving next hop.
Routed multicast (mcrt)	Regular multicast next hop.
Router	A specific node or set of nodes to which the routing device forwards packets that match the route prefix. To qualify as a next-hop type router, the route must meet the following criteria: • Must not be a direct or local subnet for the routing device. • Must have a next hop that is directly connected to the routing device.
Table	Routing table next hop.
Unicast (ucst)	Unicast.
Unilist (ulst)	List of unicast next hops. A packet sent to this next hop goes to any next hop in the list.

Table 85 on page 1428 describes all possible values for the State output field. A route can be in more than one state (for example, **<Active NoReadvrt Int Ext>**).

Table 95: State Output Field Values

Value	Description
Accounting	Route needs accounting.
Active	Route is active.
Always Compare MED	Path with a lower multiple exit discriminator (MED) is available.
AS path	Shorter AS path is available.
Cisco Non-deterministic MED selection	Cisco nondeterministic MED is enabled, and a path with a lower MED is available.
Clone	Route is a clone.
Cluster list length	Length of cluster list sent by the route reflector.
Delete	Route has been deleted.
Ex	Exterior route.

Table 95: State Output Field Values *(continued)*

Value	Description
Ext	BGP route received from an external BGP neighbor.
FlashAll	Forces all protocols to be notified of a change to any route, active or inactive, for a prefix. When not set, protocols are informed of a prefix only when the active route changes.
Hidden	Route not used because of routing policy.
IfCheck	Route needs forwarding RPF check.
IGP metric	Path through next hop with lower IGP metric is available.
Inactive reason	Flags for this route, which was not selected as best for a particular destination.
Initial	Route being added.
Int	Interior route.
Int Ext	BGP route received from an internal BGP peer or a BGP confederation peer.
Interior > Exterior > Exterior via Interior	Direct, static, IGP, or EBGP path is available.
Local Preference	Path with a higher local preference value is available.
Martian	Route is a martian (ignored because it is obviously invalid).
MartianOK	Route exempt from martian filtering.
Next hop address	Path with lower metric next hop is available.
No difference	Path from neighbor with lower IP address is available.
NoReadvrt	Route not to be advertised.
NotBest	Route not chosen because it does not have the lowest MED.
Not Best in its group	Incoming BGP AS is not the best of a group (only one AS can be the best).
NotInstall	Route not to be installed in the forwarding table.
Number of gateways	Path with a greater number of next hops is available.
Origin	Path with a lower origin code is available.

Table 95: State Output Field Values *(continued)*

Value	Description
Pending	Route pending because of a hold-down configured on another route.
Release	Route scheduled for release.
RIB preference	Route from a higher-numbered routing table is available.
Route Distinguisher	64-bit prefix added to IP subnets to make them unique.
Route Metric or MED comparison	Route with a lower metric or MED is available.
Route Preference	Route with lower preference value is available.
Router ID	Path through a neighbor with lower ID is available.
Secondary	Route not a primary route.
Unusable path	Path is not usable because of one of the following conditions: • The route is damped. • The route is rejected by an import policy. • The route is unresolved.
Update source	Last tiebreaker is the lowest IP address value.

Table 86 on page 1430 describes the possible values for the Communities output field.

Table 96: Communities Output Field Values

Value	Description
area-number	4 bytes, encoding a 32-bit area number. For AS-external routes, the value is **0**. A nonzero value identifies the route as internal to the OSPF domain, and as within the identified area. Area numbers are relative to a particular OSPF domain.
bandwidth: ***local AS number:link-bandwidth-number***	Link-bandwidth community value used for unequal-cost load balancing. When BGP has several candidate paths available for multipath purposes, it does not perform unequal-cost load balancing according to the link-bandwidth community unless all candidate paths have this attribute.
domain-id	Unique configurable number that identifies the OSPF domain.
domain-id-vendor	Unique configurable number that further identifies the OSPF domain.
link-bandwidth-number	Link-bandwidth number: from **0** through **4,294,967,295** (bytes per second).
local AS number	Local AS number: from **1** through **65,535**.

Table 96: Communities Output Field Values *(continued)*

Value	Description
options	1 byte. Currently this is only used if the route type is **5** or **7**. Setting the least significant bit in the field indicates that the route carries a type 2 metric.
origin	(Used with VPNs) Identifies where the route came from.
ospf-route-type	1 byte, encoded as **1** or **2** for intra-area routes (depending on whether the route came from a type 1 or a type 2 LSA); **3** for summary routes; **5** for external routes (area number must be**0**); **7** for NSSA routes; or **129** for sham link endpoint addresses.
route-type-vendor	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute **0x8000**. The format is *area-number:ospf-route-type:options*.
rte-type	Displays the area number, OSPF route type, and option of the route. This is configured using the BGP extended community attribute **0x0306**. The format is *area-number:ospf-route-type:options*.
target	Defines which VPN the route participates in; **target** has the format ***32-bit IP address:16-bit number***. For example, 10.19.0.0:100.
unknown IANA	Incoming IANA codes with a value between **0x1** and **0x7fff**. This code of the BGP extended community attribute is accepted, but it is not recognized.
unknown OSPF vendor community	Incoming IANA codes with a value above **0x8000**. This code of the BGP extended community attribute is accepted, but it is not recognized.

Sample Output

show route table bgp.l2.vpn

```
user@host> show route table bgp.l2.vpn
bgp.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.24.1:1:4:1/96
                   *[BGP/170] 01:08:58, localpref 100, from 192.168.24.1
                      AS path: I
                    > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am
```

show route table bgp.l3vpn.0

```
user@host> show route table bgp.l3vpn.0
      bgp.l3vpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
      + = Active Route, - = Last Active, * = Both

      10.255.71.15:100:10.255.71.17/32
                         *[BGP/170] 00:03:59, MED 1, localpref 100, from
      10.255.71.15
                            AS path: I
                          > via so-2/1/0.0, Push 100020, Push 100011(top)
      10.255.71.15:200:10.255.71.18/32
```



```
                               *[BGP/170] 00:03:59, MED 1, localpref 100, from
10.255.71.15
                                  AS path: I
                                > via so-2/1/0.0, Push 100021, Push 100011(top)
```

show route table bgp.l3vpn.0 detail

```
user@host> show route table bgp.l3vpn.0 detail
bgp.l3vpn.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)

10.255.245.12:1:172.16.4.0/8 (1 entry, 1 announced)
  *BGP    Preference: 170/-101
          Route Distinguisher: 10.255.245.12:1
          Source: 10.255.245.12
          Next hop: 192.168.208.66 via fe-0/0/0.0, selected
          Label operation: Push 182449
          Protocol next hop: 10.255.245.12
          Push 182449
          Indirect next hop: 863a630 297
          State: <Active Int Ext>
          Local AS:    35 Peer AS:    35
          Age: 12:19      Metric2: 1
          Task: BGP_35.10.255.245.12+179
          Announcement bits (1): 0-BGP.0.0.0.0+179
          AS path: 30 10458 14203 2914 3356 I (Atomic) Aggregator: 3356 4.68.0.11

          Communities: 2914:420 target:11111:1 origin:56:78
          VPN Label: 182449
          Localpref: 100
           Router ID: 10.255.245.12

10.255.245.12:1:4.17.225.0/24 (1 entry, 1 announced)
  *BGP    Preference: 170/-101
          Route Distinguisher: 10.255.245.12:1
          Source: 10.255.245.12
          Next hop: 192.168.208.66 via fe-0/0/0.0, selected
          Label operation: Push 182465
          Protocol next hop: 10.255.245.12
          Push 182465
          Indirect next hop: 863a8f0 305
          State: <Active Int Ext>
          Local AS:    35 Peer AS:    35
          Age: 12:19      Metric2: 1
          Task: BGP_35.10.255.245.12+179
          Announcement bits (1): 0-BGP.0.0.0.0+179
AS path: 30 10458 14203 2914 11853 11853 11853 6496 6496 6496 6496 6496 6496 I
                 Communities: 2914:410 target:12:34 target:11111:1 origin:12:34
                 VPN Label: 182465
                 Localpref: 100
                 Router ID: 10.255.245.12

10.255.245.12:1:4.17.226.0/23 (1 entry, 1 announced)
  *BGP    Preference: 170/-101
          Route Distinguisher: 10.255.245.12:1
          Source: 10.255.245.12
          Next hop: 192.168.208.66 via fe-0/0/0.0, selected
          Label operation: Push 182465
          Protocol next hop: 10.255.245.12
          Push 182465
          Indirect next hop: 86bd210 330
          State: <Active Int Ext>
```

```
                Local AS:    35 Peer AS:    35
                Age: 12:19      Metric2: 1
                Task: BGP_35.10.255.245.12+179
                Announcement bits (1): 0-BGP.0.0.0.0+179
                AS path: 30 10458 14203 2914 11853 11853 11853 6496 6496 6496 6496 6496

                6496 I
                Communities: 2914:410 target:12:34 target:11111:1 origin:12:34
                VPN Label: 182465
                Localpref: 100
                Router ID: 10.255.245.12

10.255.245.12:1:4.17.251.0/24 (1 entry, 1 announced)
  *BGP   Preference: 170/-101
                Route Distinguisher: 10.255.245.12:1
                Source: 10.255.245.12
                Next hop: 192.168.208.66 via fe-0/0/0.0, selected
                Label operation: Push 182465
                Protocol next hop: 10.255.245.12
                Push 182465
                Indirect next hop: 86bd210 330
                State: <Active Int Ext>
                Local AS:    35 Peer AS:    35
                Age: 12:19      Metric2: 1
                Task: BGP_35.10.255.245.12+179
                Announcement bits (1): 0-BGP.0.0.0.0+179
                AS path: 30 10458 14203 2914 11853 11853 11853 6496 6496 6496 6496 6496

                6496 I
                Communities: 2914:410 target:12:34 target:11111:1 origin:12:34
                VPN Label: 182465
                Localpref: 100
```

show route table bgp.rtarget.0 (When Proxy BGP Route Target Filtering Is Configured)

```
user@host> show route table bgp.rtarget.o
bgp.rtarget.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

100:100:100/96
                   *[RTarget/5] 00:03:14
                      Type Proxy
                        for 10.255.165.103
                        for 10.255.166.124
                      Local
```

show route table bgp.evpn.0

```
user@host> show route table bgp.evpn.0
bgp.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

2:100.100.100.2:100::0::00:26:88:5f:67:b0/304
                   *[BGP/170] 11:00:05, localpref 100, from 100.100.100.2
                      AS path: I, validation-state: unverified
                    > to 100.64.12.2 via xe-2/2/0.0, label-switched-path R0toR1
2:100.100.100.2:100::0::00:51:51:51:51:51/304
                   *[BGP/170] 11:00:05, localpref 100, from 100.100.100.2
                      AS path: I, validation-state: unverified
                    > to 100.64.12.2 via xe-2/2/0.0, label-switched-path R0toR1
```

```
2:100.100.100.3:100::0::00:52:52:52:52:52/304
                   *[BGP/170] 10:59:58, localpref 100, from 100.100.100.3
                      AS path: I, validation-state: unverified
                    > to 100.64.13.3 via ge-2/0/8.0, label-switched-path R0toR2
2:100.100.100.3:100::0::a8:d0:e5:5b:01:c8/304
                   *[BGP/170] 10:59:58, localpref 100, from 100.100.100.3
                      AS path: I, validation-state: unverified
                    > to 100.64.13.3 via ge-2/0/8.0, label-switched-path R0toR2
3:100.100.100.2:100::1000::100.100.100.2/304
                   *[BGP/170] 11:00:16, localpref 100, from 100.100.100.2
                      AS path: I, validation-state: unverified
                    > to 100.64.12.2 via xe-2/2/0.0, label-switched-path R0toR1
3:100.100.100.2:100::2000::100.100.100.2/304
                   *[BGP/170] 11:00:16, localpref 100, from 100.100.100.2
                      AS path: I, validation-state: unverified
                    > to 100.64.12.2 via xe-2/2/0.0, label-switched-path R0toR1
```

show route table evpna.evpn.0

```
user@host> show route table evpna.evpn.0
evpna.evpn.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

3:100.100.100.10:100::0::10::100.100.100.10/384
                   *[EVPN/170] 01:37:09
                      Indirect
3:100.100.100.2:100::2000::100.100.100.2/304
                   *[EVPN/170] 01:37:12
                      Indirect
```

show route table inet.0

```
user@host> show route table inet.0
inet.0: 12 destinations, 12 routes (11 active, 0 holddown, 1 hidden)
+ = Active Route, - = Last Active, * = Both

0.0.0.0/0          *[Static/5] 00:51:57
                    > to 172.16.5.254 via fxp0.0
10.0.0.1/32         *[Direct/0] 00:51:58
                    > via at-5/3/0.0
10.0.0.2/32         *[Local/0] 00:51:58
                     Local
10.12.12.21/32     *[Local/0] 00:51:57
                     Reject
10.13.13.13/32     *[Direct/0] 00:51:58
                    > via t3-5/2/1.0
10.13.13.14/32     *[Local/0] 00:51:58
                     Local
10.13.13.21/32     *[Local/0] 00:51:58
                     Local
10.13.13.22/32     *[Direct/0] 00:33:59
                    > via t3-5/2/0.0
127.0.0.1/32        [Direct/0] 00:51:58
                    > via lo0.0
10.222.5.0/24     *[Direct/0] 00:51:58
                    > via fxp0.0
10.222.5.81/32    *[Local/0] 00:51:58
                     Local
```

show route table inet.3

```
user@host> show route table inet.3
inet.3: 5 destinations, 5 routes (5 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.0.0.5/32        *[LDP/9] 00:25:43, metric 10, tag 200
                      to 10.2.94.2 via lt-1/2/0.49
                    > to 10.2.3.2 via lt-1/2/0.23
```

show route table inet.3 protocol ospf

```
user@host> show route table inet.3 protocol ospf
inet.3: 9 destinations, 18 routes (9 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1.1.1.20/32         [L-OSPF/10] 1d 00:00:56, metric 2
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800020
                     to 10.0.6.60 via lt-1/2/0.12, Push 800020, Push 800030(top)
1.1.1.30/32         [L-OSPF/10] 1d 00:01:01, metric 3
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800030
                      to 10.0.6.60 via lt-1/2/0.12, Push 800030
1.1.1.40/32         [L-OSPF/10] 1d 00:01:01, metric 4
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800040
                      to 10.0.6.60 via lt-1/2/0.12, Push 800040
1.1.1.50/32         [L-OSPF/10] 1d 00:01:01, metric 5
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800050
                      to 10.0.6.60 via lt-1/2/0.12, Push 800050
1.1.1.60/32         [L-OSPF/10] 1d 00:01:01, metric 6
                    > to 10.0.10.70 via lt-1/2/0.14, Push 800060
                      to 10.0.6.60 via lt-1/2/0.12, Pop
```

show route table inet6.0

```
user@host> show route table inet6.0
inet6.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Route, * = Both

fec0:0:0:3::/64 *[Direct/0] 00:01:34
>via fe-0/1/0.0

fec0:0:0:3::/128 *[Local/0] 00:01:34
>Local

fec0:0:0:4::/64 *[Static/5] 00:01:34
>to fec0:0:0:3::ffff via fe-0/1/0.0
```

show route table inet6.3

```
user@router> show route table inet6.3
inet6.3: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

::10.255.245.195/128
                   *[LDP/9] 00:00:22, metric 1
                    > via so-1/0/0.0
::10.255.245.196/128
                   *[LDP/9] 00:00:08, metric 1
                    > via so-1/0/0.0, Push 100008
```

show route table inetflow detail

```
user@host> show route table inetflow detail
inetflow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
10.12.44.1,*/48 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next-hop reference count: 2
                State: <Active Ext>
                Local AS: 64502 Peer AS: 64500
                Age: 4
                Task: BGP_64500.10.12.99.5+3792
                Announcement bits (1): 0-Flow
                AS path: 64500 I
                Communities: traffic-rate:0:0
                Validation state: Accept, Originator: 10.12.99.5
                Via: 10.12.44.0/24, Active
                Localpref: 100
                Router ID: 10.255.71.161

10.12.56.1,*/48 (1 entry, 1 announced)
        *Flow   Preference: 5
                Next-hop reference count: 2
                State: <Active>
                Local AS: 64502
                Age: 6:30
                Task: RT Flow
                Announcement bits (2): 0-Flow 1-BGP.0.0.0.0+179
                AS path: I
                Communities: 1:1
```

show route table lsdist.0 extensive

```
user@host> show route table lsdist.0 extensive
lsdist.0: 10 destinations, 10 routes (10 active, 0 holddown, 0 hidden)
NODE { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 ISIS-L1:0 }/1152
 (1 entry, 1 announced)
TSI:
Page 0 idx 0, (group ibgp type Internal) Type 1 val 0xa62f378 (adv_entry)
   Advertised metrics:
     Nexthop: Self
     Localpref: 100
     AS path: [4170512532] I
     Communities:
Path NODE { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 ISIS-L1:0 }
 Vector len 4.  Val: 0
        *IS-IS  Preference: 15
                Level: 1
                Next hop type: Fictitious, Next hop index: 0
                Address: 0x95dfc64
                Next-hop reference count: 9
                State: <Active NotInstall>
                Local AS: 4170512532
                Age: 6:05
                Validation State: unverified
                Task: IS-IS
                Announcement bits (1): 0-BGP_RT_Background
                AS path: I
                IPv4 Router-ids:
                   128.220.11.197
                Area membership:
```



```
                   47 00 05 80 ff f8 00 00 00 01 08 00 01
               SPRING-Capabilities:                 - SRGB block [Start: 800000,
 Range: 256, Flags: 0xc0]
                 SPRING-Algorithms:
                    - Algo: 0
 LINK { Local { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 }.{
IPv4:8.65.1.105 } Remote { AS:4170512532 BGP-LS ID:4170512532 ISO:4284.3300.5067)
TSI:
Page 0 idx 0, (group ibgp type Internal) Type 1 val 0xa62f3cc (adv_entry)
    Advertised metrics:
      Nexthop: Self
      Localpref: 100
      AS path: [4170512532] I
      Communities:
Path LINK { Local { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 }.{
 IPv4:8.65.1.105 } Remote { AS:4170512532 BGP-LS ID:4170512532 ISO:4284.33000
      *IS-IS  Preference: 15
                Level: 1
                Next hop type: Fictitious, Next hop index: 0
                Address: 0x95dfc64
                Next-hop reference count: 9
                State: <Active NotInstall>
                Local AS: 4170512532
                Age: 6:05
                Validation State: unverified
                Task: IS-IS
                Announcement bits (1): 0-BGP_RT_Background
                AS path: I
                Color: 32768
                Maximum bandwidth: 1000Mbps
                Reservable bandwidth: 1000Mbps
                Unreserved bandwidth by priority:
                   0   1000Mbps
                   1   1000Mbps
                   2   1000Mbps
                   3   1000Mbps
                   4   1000Mbps
                   5   1000Mbps
                   6   1000Mbps
                   7   1000Mbps
                 Metric: 10
                 TE Metric: 10
                 LAN IPV4 Adj-SID -  Label: 299776, Flags: 0x30,
Weight: 0, Nbr: 10.220.1.83

PREFIX { Node { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 } {
IPv4:128.220.11.197/32 } ISIS-L1:0 }/1152 (1 entry, 1 announced) TSI: Page 0 idx
 0, (group ibgp type Internal) Type 1 val 0xa62f43c (adv_entry)
    Advertised metrics:
      Nexthop: Self
      Localpref: 100
      AS path: [4170512532] I
      Communities:
Path PREFIX { Node { AS:4170512532 BGP-LS ID:4170512532 ISO:3245.3412.3456.00 }
{ IPv4:128.220.11.197/32 } ISIS-L1:0 } Vector len 4.  Val: 0
         *IS-IS  Preference: 15
                 Level: 1
                 Next hop type: Fictitious, Next hop index: 0
                 Address: 0x95dfc64
                 Next-hop reference count: 9
                 State:<Active NotInstall>
```



```
                Local AS: 4170512532
                Age: 6:05
                Validation State: unverified
                Task: IS-IS
                Announcement bits (1): 0-BGP_RT_Background
                AS path: I
                              Prefix SID: 67, Flags: 0x40, Algo: 0
```

show route table l2circuit.0

```
user@host> show route table l2circuit.0
l2circuit.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.195:NoCtrlWord:1:1:Local/96
                   *[L2CKT/7] 00:50:47
                    > via so-0/1/2.0, Push 100049
                      via so-0/1/3.0, Push 100049
10.1.1.195:NoCtrlWord:1:1:Remote/96
                   *[LDP/9] 00:50:14
                      Discard
10.1.1.195:CtrlWord:1:2:Local/96
                   *[L2CKT/7] 00:50:47
                    > via so-0/1/2.0, Push 100049
                      via so-0/1/3.0, Push 100049
10.1.1.195:CtrlWord:1:2:Remote/96
                   *[LDP/9] 00:50:14
                      Discard
```

show route table mpls

```
user@host> show route table mpls
mpls.0: 4 destinations, 4 routes (4 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 00:13:55, metric 1
                      Receive
1                  *[MPLS/0] 00:13:55, metric 1
                      Receive
2                  *[MPLS/0] 00:13:55, metric 1
                      Receive
1024               *[VPN/0] 00:04:18
                      to table red.inet.0, Pop
```

show route table mpls extensive

```
user@host> show route table mpls extensive
100000 (1 entry, 1 announced)
TSI:
KRT in-kernel 100000 /36 -> {so-1/0/0.0}
        *LDP    Preference:   9
                Next hop:  via so-1/0/0.0, selected
                Pop
                State: <Active Int>
                Age: 29:50      Metric: 1
                Task: LDP
                Announcement bits (1): 0-KRT
                AS path: I
                Prefixes bound to route: 10.0.0.194/32
```


show route table mpls.0

```
user@host> show route table mpls.0
mpls.0: 18 destinations, 19 routes (18 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 11:39:56, metric 1
                      to table inet.0
0(S=0)             *[MPLS/0] 11:39:56, metric 1
                      to table mpls.0
1                  *[MPLS/0] 11:39:56, metric 1
                      Receive
2                  *[MPLS/0] 11:39:56, metric 1
                      to table inet6.0
2(S=0)             *[MPLS/0] 11:39:56, metric 1
                      to table mpls.0
13                 *[MPLS/0] 11:39:56, metric 1
                      Receive
303168             *[EVPN/7] 11:00:49, routing-instance pbbn10, route-type
Ingress-MAC, ISID 0
                      to table pbbn10.evpn-mac.0
303184             *[EVPN/7] 11:00:53, routing-instance pbbn10, route-type
Ingress-IM, ISID 1000
                      to table pbbn10.evpn-mac.0
                    [EVPN/7] 11:00:53, routing-instance pbbn10, route-type
Ingress-IM, ISID 2000
                      to table pbbn10.evpn-mac.0
303264             *[EVPN/7] 11:00:53, remote-pe 100.100.100.2, routing-instance
 pbbn10, route-type Egress-IM, ISID 1000
                    > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303280             *[EVPN/7] 11:00:53, remote-pe 100.100.100.2, routing-instance
 pbbn10, route-type Egress-IM, ISID 2000
                    > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303328             *[EVPN/7] 11:00:49, remote-pe 100.100.100.2, routing-instance
 pbbn10, route-type Egress-MAC, ISID 0
                    > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303344             *[EVPN/7] 11:00:49, remote-pe 100.100.100.2, routing-instance
 pbbn10, route-type Egress-MAC, ISID 0
                    > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303360             *[EVPN/7] 11:00:47, routing-instance pbbn10, route-type
Egress-MAC, ISID 0, BMAC 00:26:88:5f:67:b0
                    > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303376             *[EVPN/7] 11:00:47, routing-instance pbbn10, route-type
Egress-MAC, ISID 0, BMAC 00:51:51:51:51:51
                    > to 100.1.12.2 via xe-2/2/0.0, label-switched-path R0toR1
303392             *[EVPN/7] 11:00:35, remote-pe 100.100.100.3, routing-instance
 pbbn10, route-type Egress-MAC, ISID 0
                    > to 100.1.13.3 via ge-2/0/8.0, label-switched-path R0toR2
303408             *[EVPN/7] 11:00:35, remote-pe 100.100.100.3, routing-instance
 pbbn10, route-type Egress-MAC, ISID 0
                    > to 100.1.13.3 via ge-2/0/8.0, label-switched-path R0toR2
303424             *[EVPN/7] 11:00:33, routing-instance pbbn10, route-type
Egress-MAC, ISID 0, BMAC a8:d0:e5:5b:01:c8
                    > to 100.1.13.3 via ge-2/0/8.0, label-switched-path R0toR2
303440             *[EVPN/7] 11:00:33, routing-instance pbbn10, route-type
Egress-MAC, ISID 0, BMAC 00:52:52:52:52:52
                    > to 100.1.13.3 via ge-2/0/8.0, label-switched-path R0toR2
```


show route table mpls.0 detail (PTX Series)

```
user@host> show route table mpls.0 detail
ge-0/0/2.600 (1 entry, 1 announced)
        *L2VPN  Preference: 7
                Next hop type: Indirect
                Address: 0x9438f34
                Next-hop reference count: 2
                Next hop type: Router, Next hop index: 567
                Next hop: 10.0.0.1 via ge-0/0/1.0, selected
                Label operation: Push 299808
                Label TTL action: prop-ttl
                Load balance label: Label 299808:None;
                Session Id: 0x1
                Protocol next hop: 10.255.255.1
                Label operation: Push 299872 Offset: 252
                Label TTL action: no-prop-ttl
                Load balance label: Label 299872:Flow label PUSH;
                Composite next hop: 0x9438ed8 570 INH Session ID: 0x2
                Indirect next hop: 0x9448208 262142 INH Session ID: 0x2
                State: <Active Int>
                Age: 21         Metric2: 1
                Validation State: unverified
                Task: Common L2 VC
                Announcement bits (2): 0-KRT 2-Common L2 VC
                AS path: I
```

show route table mpls.0 ccc ge-0/0/1.1004 detail

```
user@host>show route table mpls.0 ccc ge-0/0/1.1004 detail
mpls.0: 121 destinations, 121 routes (121 active, 0 holddown, 0 hidden)
ge-0/0/1.1004 (1 entry, 1 announced)
        *EVPN   Preference: 7
                Next hop type: List, Next hop index: 1048577
                Address: 0xdc14770
                Next-hop reference count: 3
                Next hop: ELNH Address 0xd011e30
                    Next hop type: Indirect, Next hop index: 0
                    Address: 0xd011e30
                    Next-hop reference count: 3
                    Protocol next hop: 100.100.100.1
                    Label operation: Push 301952
                    Composite next hop: 0xd011dc0 754 INH Session ID: 0x146
                    Indirect next hop: 0xb69a890 1048615 INH Session ID: 0x146
                        Next hop type: Router, Next hop index: 735
                        Address: 0xd00e530
                        Next-hop reference count: 23
                        Next hop: 100.46.1.2 via ge-0/0/5.0
                        Label-switched-path pe4_to_pe1
                        Label operation: Push 300320
                        Label TTL action: prop-ttl
                        Load balance label: Label 300320: None;
                        Label element ptr: 0xd00e580
                        Label parent element ptr: 0x0
                        Label element references: 18
                        Label element child references: 16
                        Label element lsp id: 5
                Next hop: ELNH Address 0xd012070
                    Next hop type: Indirect, Next hop index: 0
                    Address: 0xd012070
```

```
                Next-hop reference count: 3
                Protocol next hop: 100.100.100.2
                Label operation: Push 301888
                Composite next hop: 0xd012000 755 INH Session ID: 0x143
                Indirect next hop: 0xb69a9a0 1048641 INH Session ID: 0x143
                    Next hop type: Router, Next hop index: 716
                    Address: 0xd00e710
                    Next-hop reference count: 23
                    Next hop: 100.46.1.2 via ge-0/0/5.0
                    Label-switched-path pe4_to_pe2
                    Label operation: Push 300304
                    Label TTL action: prop-ttl
                    Load balance label: Label 300304: None;
                    Label element ptr: 0xd00e760
                    Label parent element ptr: 0x0
                    Label element references: 15
                    Label element child references: 13
                    Label element lsp id: 6
            Next hop: ELNH Address 0xd0121f0, selected
                Next hop type: Indirect, Next hop index: 0
                Address: 0xd0121f0
                Next-hop reference count: 3
                Protocol next hop: 100.100.100.3
                Label operation: Push 301984
                Composite next hop: 0xd012180 756 INH Session ID: 0x145
                Indirect next hop: 0xb69aab0 1048642 INH Session ID: 0x145
                    Next hop type: Router, Next hop index: 801
                    Address: 0xd010ed0
                    Next-hop reference count: 32
                    Next hop: 100.46.1.2 via ge-0/0/5.0
                    Label-switched-path pe4_to_pe3
                    Label operation: Push 300336
                    Label TTL action: prop-ttl
                    Load balance label: Label 300336: None;
                    Label element ptr: 0xd0108c0
                    Label parent element ptr: 0x0
                    Label element references: 22
                    Label element child references: 20
                    Label element lsp id: 7
            State: < Active Int >
            Age: 2:06:50
            Validation State: unverified
            Task: evpn global task
            Announcement bits (1): 1-KRT
            AS path: I
```

show route table mpls.0 protocol evpn

```
user@host>show route table mpls.0 protocol evpn
mpls.0: 121 destinations, 121 routes (121 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

299872             *[EVPN/7] 02:30:58, routing-instance mhevpn, route-type
Ingress-IM, vlan-id 10
                      to table mhevpn.evpn-mac.0
300016             *[EVPN/7] 02:30:38, routing-instance VS-1, route-type
Ingress-IM, vlan-id 110
                      to table VS-1.evpn-mac.0
300032             *[EVPN/7] 02:30:38, routing-instance VS-1, route-type
Ingress-IM, vlan-id 120
                      to table VS-1.evpn-mac.0
```



```
300048             *[EVPN/7] 02:30:38, routing-instance VS-1, route-type
Ingress-IM, vlan-id 130
                      to table VS-1.evpn-mac.0
300064             *[EVPN/7] 02:30:38, routing-instance VS-2, route-type
Ingress-IM, vlan-id 210
                      to table VS-2.evpn-mac.0
300080             *[EVPN/7] 02:30:38, routing-instance VS-2, route-type
Ingress-IM, vlan-id 220
                      to table VS-2.evpn-mac.0
300096             *[EVPN/7] 02:30:38, routing-instance VS-2, route-type
Ingress-IM, vlan-id 230
                      to table VS-2.evpn-mac.0
300112             *[EVPN/7] 02:27:06, routing-instance mhevpn, route-type
Egress-MAC, ESI 00:44:44:44:44:44:44:44:44:44
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300128             *[EVPN/7] 02:29:22, routing-instance mhevpn, route-type
Ingress-Aliasing
                      to table mhevpn.evpn-mac.0
300144             *[EVPN/7] 02:27:06, routing-instance VS-1, route-type
Egress-MAC, ESI 00:44:44:44:44:44:44:44:44:44
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300160             *[EVPN/7] 02:29:22, routing-instance VS-1, route-type
Ingress-Aliasing
                      to table VS-1.evpn-mac.0
300176             *[EVPN/7] 02:27:07, routing-instance VS-2, route-type
Egress-MAC, ESI 00:44:44:44:44:44:44:44:44:44
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300192             *[EVPN/7] 02:29:22, routing-instance VS-2, route-type
Ingress-Aliasing
                      to table VS-2.evpn-mac.0
300208             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-IM, vlan-id 120
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300224             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 mhevpn, route-type Egress-IM, vlan-id 10
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300240             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-IM, vlan-id 110
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300256             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-IM, vlan-id 130
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300272             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-IM, vlan-id 210
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300288             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-IM, vlan-id 220
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300304             *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-IM, vlan-id 230
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300320             *[EVPN/7] 02:27:06, routing-instance VS-1, route-type
Egress-MAC, ESI 00:11:11:11:11:11:11:11:11:11
                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300336             *[EVPN/7] 02:27:06, routing-instance VS-1, route-type
Egress-MAC, ESI 00:33:33:33:33:33:33:33:33:33
                    to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
```



```
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300368              *[EVPN/7] 02:27:07, routing-instance VS-2, route-type
Egress-MAC, ESI 00:33:33:33:33:33:33:33:33:33
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300384              *[EVPN/7] 02:27:07, routing-instance VS-2, route-type
Egress-MAC, ESI 00:11:11:11:11:11:11:11:11:11
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300416              *[EVPN/7] 02:27:06, routing-instance mhevpn, route-type
Egress-MAC, ESI 00:33:33:33:33:33:33:33:33:33
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300432              *[EVPN/7] 02:27:06, routing-instance mhevpn, route-type
Egress-MAC, ESI 00:11:11:11:11:11:11:11:11:11
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
300480              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300496              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300560              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-1, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300592              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 VS-2, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
300608              *[EVPN/7] 02:29:23
                    > via ge-0/0/1.1001, Pop
300624              *[EVPN/7] 02:29:23
                    > via ge-0/0/1.2001, Pop
301232              *[EVPN/7] 02:29:17
                    > via ge-0/0/1.1002, Pop
301296              *[EVPN/7] 02:29:10
                    > via ge-0/0/1.1003, Pop
301312              *[EVPN/7] 02:27:06
                    > via ae10.2003, Pop
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301360              *[EVPN/7] 02:29:01
                    > via ge-0/0/1.1004, Pop
301408              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 vpws1004, route-type Egress, vlan-id 2004
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
301456              *[EVPN/7] 02:27:06
                    > via ae10.1010, Pop
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301552              *[EVPN/7] 02:27:07, routing-instance VS-1, route-type
Egress-MAC, ESI 00:22:22:22:22:22:22:22:22:22
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301568              *[EVPN/7] 02:27:07, routing-instance VS-2, route-type
```



```
Egress-MAC, ESI 00:22:22:22:22:22:22:22:22:22
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301648              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 vpws1010, route-type Egress, vlan-id 2010
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
301664              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 mhevpn, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
301680              *[EVPN/7] 02:27:07, remote-pe 100.100.100.2, routing-instance
 mhevpn, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
301696              *[EVPN/7] 02:27:07, routing-instance mhevpn, route-type
Egress-MAC, ESI 00:22:22:22:22:22:22:22:22:22
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301712              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301728              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301744              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-IM, vlan-id 230
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301760              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 vpws1010, route-type Egress, vlan-id 2010
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301776              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 mhevpn, route-type Egress-MAC
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301792              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-IM, vlan-id 130
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301808              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 vpws1004, route-type Egress, vlan-id 2004
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301824              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 mhevpn, route-type Egress-IM, vlan-id 10
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301840              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 vpws1002, route-type Egress, vlan-id 2002
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301856              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 vpws1003, route-type Egress, vlan-id 2003
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301872              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 vpws1003, route-type Egress Protection, vlan-id 2003
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301888              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 vpws1010, route-type Egress Protection, vlan-id 1010
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301904              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-IM, vlan-id 220
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301920              *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-IM, vlan-id 210
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
301936              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-IM, vlan-id 230
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301952              *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-SH, vlan-id 230
```



```
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301968               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-IM, vlan-id 220
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
301984               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-SH, vlan-id 220
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302000               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-IM, vlan-id 210
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302016               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-SH, vlan-id 210
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302032               *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302048               *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-2, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302064               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302080               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-2, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302096               *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302112               *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302128               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302144               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-MAC
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302160               *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-IM, vlan-id 120
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302176               *[EVPN/7] 02:27:07, remote-pe 100.100.100.1, routing-instance
 VS-1, route-type Egress-IM, vlan-id 110
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302192               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-IM, vlan-id 130
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302208               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-SH, vlan-id 130
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302224               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-IM, vlan-id 120
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302240               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-SH, vlan-id 120
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302256               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-IM, vlan-id 110
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302272               *[EVPN/7] 02:27:07, remote-pe 100.100.100.3, routing-instance
 VS-1, route-type Egress-SH, vlan-id 110
                    > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
```



```
302288             *[EVPN/7] 02:27:06, remote-pe 100.100.100.1, routing-instance
 mhevpn, route-type Egress-MAC
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302304             *[EVPN/7] 02:27:06, remote-pe 100.100.100.1, routing-instance
 mhevpn, route-type Egress-MAC
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302320             *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 mhevpn, route-type Egress-MAC
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302336             *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 mhevpn, route-type Egress-MAC
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302352             *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 vpws1004, route-type Egress, vlan-id 2004
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302368             *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 mhevpn, route-type Egress-IM, vlan-id 10
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302384             *[EVPN/7] 02:27:06, remote-pe 100.100.100.3, routing-instance
 mhevpn, route-type Egress-SH, vlan-id 10
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302400             *[EVPN/7] 02:26:21
                    > via ge-0/0/1.3001, Pop
302432             *[EVPN/7] 02:26:21, remote-pe 100.100.100.3, routing-instance
 vpws3001, route-type Egress, vlan-id 40000
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302448             *[EVPN/7] 02:26:21, remote-pe 100.100.100.1, routing-instance
 vpws3001, route-type Egress, vlan-id 40000
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302464             *[EVPN/7] 02:26:20, remote-pe 100.100.100.2, routing-instance
 vpws3001, route-type Egress, vlan-id 40000
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
302480             *[EVPN/7] 02:26:14
                    > via ge-0/0/1.3016, Pop
302512             *[EVPN/7] 02:26:14, remote-pe 100.100.100.1, routing-instance
 vpws3016, route-type Egress, vlan-id 40016
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302528             *[EVPN/7] 02:26:14, remote-pe 100.100.100.2, routing-instance
 vpws3016, route-type Egress, vlan-id 40016
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
302560             *[EVPN/7] 02:26:06
                    > via ae10.3011, Pop
                      to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302592             *[EVPN/7] 02:26:07, remote-pe 100.100.100.1, routing-instance
 vpws3011, route-type Egress, vlan-id 401100
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
302608             *[EVPN/7] 02:26:07, remote-pe 100.100.100.2, routing-instance
 vpws3011, route-type Egress, vlan-id 401100
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
302624             *[EVPN/7] 02:26:07, remote-pe 100.100.100.3, routing-instance
 vpws3011, route-type Egress Protection, vlan-id 301100
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302656             *[EVPN/7] 02:25:59
                    > via ae10.3006, Pop
                      to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302688             *[EVPN/7] 02:26:00, remote-pe 100.100.100.2, routing-instance
 vpws3006, route-type Egress, vlan-id 400600
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2
302704             *[EVPN/7] 02:26:00, remote-pe 100.100.100.1, routing-instance
 vpws3006, route-type Egress, vlan-id 400600
                  > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
```

```
302720             *[EVPN/7] 02:25:59, remote-pe 100.100.100.3, routing-instance
 vpws3006, route-type Egress, vlan-id 400600
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
302736             *[EVPN/7] 02:25:59, remote-pe 100.100.100.3, routing-instance
 vpws3006, route-type Egress Protection, vlan-id 300600
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
ge-0/0/1.1001      *[EVPN/7] 02:29:23
                    > via ge-0/0/1.2001
ge-0/0/1.2001      *[EVPN/7] 02:29:23
                    > via ge-0/0/1.1001
ge-0/0/1.1002      *[EVPN/7] 02:27:06
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
ae10.2003          *[EVPN/7] 02:29:10
                    > via ge-0/0/1.1003
ge-0/0/1.1003      *[EVPN/7] 02:27:06
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3

                    > via ae10.2003
ge-0/0/1.1004      *[EVPN/7] 02:27:06
                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
ae10.1010          *[EVPN/7] 02:27:06
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
ge-0/0/1.3001      *[EVPN/7] 02:26:20
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
ge-0/0/1.3016      *[EVPN/7] 02:26:13
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
ae10.3011          *[EVPN/7] 02:26:06
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1
ae10.3006          *[EVPN/7] 02:25:59
                   > to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe1

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe2

                     to 100.46.1.2 via ge-0/0/5.0, label-switched-path pe4_to_pe3
```

show route table mpls.0 protocol ospf

```
user@host> show route table mpls.0 protocol ospf
mpls.0: 29 destinations, 29 routes (29 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

299952             *[L-OSPF/10] 23:59:42, metric 0
                    > to 10.0.10.70 via lt-1/2/0.14, Pop
                     to 10.0.6.60 via lt-1/2/0.12, Swap 800070, Push 800030(top)
299952(S=0)        *[L-OSPF/10] 23:59:42, metric 0
                    > to 10.0.10.70 via lt-1/2/0.14, Pop
                     to 10.0.6.60 via lt-1/2/0.12, Swap 800070, Push 800030(top)
299968             *[L-OSPF/10] 23:59:48, metric 0
                    > to 10.0.6.60 via lt-1/2/0.12, Pop
```


show route table mpls.0 extensive (PTX Series)

```
user@host> show route table mpls.0 extensive
ge-0/0/2.600 (1 entry, 1 announced)
TSI:
KRT in-kernel ge-0/0/2.600.0      /32 -> {composite(570)}
        *L2VPN  Preference: 7
                Next hop type: Indirect
                Address: 0x9438f34
                Next-hop reference count: 2
                Next hop type: Router, Next hop index: 567
                Next hop: 10.0.0.1 via ge-0/0/1.0, selected
                Label operation: Push 299808
                Label TTL action: prop-ttl
                Load balance label: Label 299808:None;
                Session Id: 0x1
                Protocol next hop: 10.255.255.1
                Label operation: Push 299872 Offset: 252
                Label TTL action: no-prop-ttl
                Load balance label: Label 299872:Flow label PUSH;
                Composite next hop: 0x9438ed8 570 INH Session ID: 0x2
                Indirect next hop: 0x9448208 262142 INH Session ID: 0x2
                State: <Active Int>
                Age: 47         Metric2: 1
                Validation State: unverified
                Task: Common L2 VC
                Announcement bits (2): 0-KRT 2-Common L2 VC
                AS path: I
                Composite next hops: 1
                        Protocol next hop: 10.255.255.1 Metric: 1
                        Label operation: Push 299872 Offset: 252
                        Label TTL action: no-prop-ttl
                        Load balance label: Label 299872:Flow label PUSH;
                        Composite next hop: 0x9438ed8 570 INH Session ID: 0x2
                        Indirect next hop: 0x9448208 262142 INH Session ID: 0x2
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.0.0.1 via ge-0/0/1.0
                                Session Id: 0x1
                        10.255.255.1/32 Originating RIB: inet.3
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 10.0.0.1 via ge-0/0/1.0
```

show route table mpls.0 (RSVP Route—Transit LSP)

```
user@host> show route table mpls.0

mpls.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

0                  *[MPLS/0] 00:37:31, metric 1
                      Receive
1                  *[MPLS/0] 00:37:31, metric 1
                      Receive
2                  *[MPLS/0] 00:37:31, metric 1
                      Receive
13                 *[MPLS/0] 00:37:31, metric 1
                      Receive
300352             *[RSVP/7/1] 00:08:00, metric 1
```

```
                   > to 10.64.0.106 via ge-1/0/1.0, label-switched-path lsp1_p2p
300352(S=0)        *[RSVP/7/1] 00:08:00, metric 1
                   > to 10.64.0.106 via ge-1/0/1.0, label-switched-path lsp1_p2p
300384             *[RSVP/7/2] 00:05:20, metric 1
                    > to 10.64.1.106 via ge-1/0/0.0, Pop
300384(S=0)        *[RSVP/7/2] 00:05:20, metric 1
                    > to 10.64.1.106 via ge-1/0/0.0, Pop
```

show route table vpls_1 detail

```
user@host> show route table vpls_1 detail
vpls_1.l2vpn.0: 1 destinations, 1 routes (1 active, 0 holddown, 0 hidden)
Restart Complete

172.16.1.11:1000:1:1/96 (1 entry, 1 announced)
*L2VPN Preference: 170/-1
Receive table: vpls_1.l2vpn.0
Next-hop reference count: 2
State: <Active Int Ext>
Age: 4:29:47 Metric2: 1
Task: vpls_1-l2vpn
Announcement bits (1): 1-BGP.0.0.0.0+179
AS path: I
Communities: Layer2-info: encaps:VPLS, control flags:Site-Down
Label-base: 800000, range: 8, status-vector: 0xFF
```

show route table vpn-a

```
user@host> show route table vpn-a
vpn-a.l2vpn.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)

+ = Active Route, - = Last Active, * = Both
192.168.16.1:1:1:1/96
                   *[VPN/7] 05:48:27
                     Discard
192.168.24.1:1:2:1/96
                   *[BGP/170] 00:02:53, localpref 100, from 192.168.24.1
                      AS path: I
                    > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am
192.168.24.1:1:3:1/96
                   *[BGP/170] 00:02:53, localpref 100, from 192.168.24.1
                      AS path: I
                    > to 10.0.16.2 via fe-0/0/1.0, label-switched-path am
```

show route table vpn-a.mdt.0

```
user@host> show route table vpn-a.mdt.0
vpn-a.mdt.0: 3 destinations, 3 routes (3 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:1:0:10.255.14.216:232.1.1.1/144
                   *[MVPN/70] 01:23:05, metric2 1
                      Indirect
1:1:1:10.255.14.218:232.1.1.1/144
                   *[BGP/170] 00:57:49, localpref 100, from 10.255.14.218
                      AS path: I
                    > via so-0/0/0.0, label-switched-path r0e-to-r1
1:1:2:10.255.14.217:232.1.1.1/144
                   *[BGP/170] 00:57:49, localpref 100, from 10.255.14.217
```



```
                      AS path: I
> via so-0/0/1.0, label-switched-path r0-to-r2
```

show route table VPN-A detail

```
user@host> show route table VPN-A detail
VPN-AB.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
10.255.179.9/32 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 10.255.179.13:200
                Next hop type: Indirect
                Next-hop reference count: 5
                Source: 10.255.179.13
                Next hop type: Router, Next hop index: 732
                Next hop: 10.39.1.14 via fe-0/3/0.0, selected
                Label operation: Push 299824, Push 299824(top)
                Protocol next hop: 10.255.179.13
                Push 299824
                Indirect next hop: 8f275a0 1048574
                State: (Secondary Active Int Ext)
                Local AS: 1 Peer AS: 1
                Age: 3:41:06 Metric: 1 Metric2: 1
                Task: BGP_1.10.255.179.13+64309
                Announcement bits (2): 0-KRT 1-BGP RT Background
                AS path: I
                Communities: target:1:200 rte-type:0.0.0.0:1:0
                Import Accepted
                VPN Label: 299824 TTL Action: vrf-ttl-propagate
                Localpref: 100
                Router ID: 10.255.179.13
                Primary Routing Table bgp.l3vpn.0
```

show route table VPN-AB.inet.0

```
user@host> show route table VPN-AB.inet.0
      VPN-AB.inet.0: 8 destinations, 8 routes (8 active, 0 holddown, 0 hidden)
      + = Active Route, - = Last Active, * = Both

      10.39.1.0/30       *[OSPF/10] 00:07:24, metric 1
                          > via so-7/3/1.0
      10.39.1.4/30       *[Direct/0] 00:08:42
                          > via so-5/1/0.0
      10.39.1.6/32       *[Local/0] 00:08:46
                           Local
      10.255.71.16/32    *[Static/5] 00:07:24
                          > via so-2/0/0.0
      10.255.71.17/32    *[BGP/170] 00:07:24, MED 1, localpref 100, from
      10.255.71.15
                            AS path: I
                          > via so-2/1/0.0, Push 100020, Push 100011(top)
      10.255.71.18/32    *[BGP/170] 00:07:24, MED 1, localpref 100, from
      10.255.71.15
                            AS path: I
                          > via so-2/1/0.0, Push 100021, Push 100011(top)
      10.255.245.245/32  *[BGP/170] 00:08:35, localpref 100
                            AS path: 2 I
                          > to 10.39.1.5 via so-5/1/0.0
      10.255.245.246/32  *[OSPF/10] 00:07:24, metric 1
                          > via so-7/3/1.0
```


show route table VPN_blue.mvpn-inet6.0

```
user@host> show route table VPN_blue.mvpn-inet6.0
vpn_blue.mvpn-inet6.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

1:10.255.2.202:65536:10.255.2.202/432
                   *[BGP/170] 00:02:37, localpref 100, from 10.255.2.202
                      AS path: I
                    > via so-0/1/3.0
1:10.255.2.203:65536:10.255.2.203/432
                   *[BGP/170] 00:02:37, localpref 100, from 10.255.2.203
                      AS path: I
                    > via so-0/1/0.0
1:10.255.2.204:65536:10.255.2.204/432
                   *[MVPN/70] 00:57:23, metric2 1
                      Indirect
5:10.255.2.202:65536:128:::192.168.90.2:128:ffff::1/432
                   *[BGP/170] 00:02:37, localpref 100, from 10.255.2.202
                      AS path: I
                    > via so-0/1/3.0
6:10.255.2.203:65536:64500:128:::10.12.53.12:128:ffff::1/432
                   *[PIM/105] 00:02:37
                      Multicast (IPv6)
7:10.255.2.202:65536:64500:128:::192.168.90.2:128:ffff::1/432
                   *[MVPN/70] 00:02:37, metric2 1
                      Indirect
```

show route table vrf1.mvpn.0 extensive

```
user@host> show route table vrf1.mvpn.0 extensive
1:10.255.50.77:1:10.255.50.77/240 (1 entry, 1 announced)
        *MVPN   Preference: 70
                PMSI: Flags 0x0: Label 0: RSVP-TE:
Session_13[10.255.50.77:0:25624:10.255.50.77]
                Next hop type: Indirect
                Address: 0xbb2c944
                Next-hop reference count: 360
                Protocol next hop: 10.255.50.77
                Indirect next hop: 0x0 - INH Session ID: 0x0
                State: <Active Int Ext>
                Age: 53:03      Metric2: 1
                Validation State: unverified
                Task: mvpn global task
                Announcement bits (3): 0-PIM.vrf1 1-mvpn global task 2-rt-export

                AS path: I
```

show route table inetflow detail

```
user@host> show route table inetflow detail
inetflow.0: 2 destinations, 2 routes (2 active, 0 holddown, 0 hidden)
10.12.44.1,*/48 (1 entry, 1 announced)
        *BGP    Preference: 170/-101
                Next-hop reference count: 2
                State: <Active Ext>
                Local AS: 64502 Peer AS: 64500
                Age: 4
                Task: BGP_64500.10.12.99.5+3792
                Announcement bits (1): 0-Flow
```



```
                AS path: 64500 I
                Communities: traffic-rate:0:0
                Validation state: Accept, Originator: 10.12.99.5
                Via: 10.12.44.0/24, Active
                Localpref: 100
                Router ID: 10.255.71.161

10.12.56.1,*/48 (1 entry, 1 announced)
        *Flow   Preference: 5
                Next-hop reference count: 2
                State: <Active>
                Local AS: 64502
                Age: 6:30
                Task: RT Flow
                Announcement bits (2): 0-Flow 1-BGP.0.0.0.0+179
                AS path: I
                Communities: 1:1
```

```
user@host> show route table green.l2vpn.0 (VPLS Multihoming with FEC 129)
green.l2vpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

10.1.1.2:100:10.1.1.2/96 AD
                   *[VPLS/170] 1d 03:11:03, metric2 1
                         Indirect
10.1.1.4:100:10.1.1.4/96 AD
                   *[BGP/170] 1d 03:11:02, localpref 100, from 10.1.1.4
                      AS path: I, validation-state: unverified
                    >    via ge-1/2/1.5
10.1.1.2:100:1:0/96 MH
                   *[VPLS/170] 1d 03:11:03, metric2 1
                         Indirect
10.1.1.4:100:1:0/96 MH
                   *[BGP/170] 1d 03:11:02, localpref 100, from 10.1.1.4
                      AS path: I, validation-state: unverified
                    >    via ge-1/2/1.5
10.1.1.4:NoCtrlWord:5:100:100:10.1.1.2:10.1.1.4/176
                   *[VPLS/7] 1d 03:11:02, metric2 1
                    >    via ge-1/2/1.5
10.1.1.4:NoCtrlWord:5:100:100:10.1.1.4:10.1.1.2/176
                   *[LDP/9] 1d 03:11:02
                         Discard
```

```
user@host> show route table red extensive
red.inet.0: 364481 destinations, 714087 routes (364480 active, 48448 holddown, 1
 hidden)
10.0.0.0/32 (3 entries, 1 announced)
        State: <OnList CalcForwarding>
TSI:
KRT in-kernel 10.0.0.0/32 -> {composite(1048575)} Page 0 idx 1 Type 1 val 0x934342c

    Nexthop: Self
    AS path: [2] I
    Communities: target:2:1
Path 10.0.0.0 from 10.3.0.0 Vector len 4.  Val: 1
        @BGP    Preference: 170/-1
                Route Distinguisher: 2:1
                Next hop type: Indirect
                Address: 0x258059e4
                Next-hop reference count: 2
```

```
                Source: 2.2.0.0
                Next hop type: Router
                Next hop: 10.1.1.1 via ge-1/1/9.0, selected
                Label operation: Push 707633
                Label TTL action: prop-ttl
                Session Id: 0x17d8
                Protocol next hop: 10.2.0.0
                Push 16
                Composite next hop: 0x25805988 - INH Session ID: 0x193c
                Indirect next hop: 0x23eea900 - INH Session ID: 0x193c
                State: <Secondary Active Int Ext ProtectionPath ProtectionCand>
                Local AS:     2 Peer AS:     2
                Age: 23         Metric2: 35
                Validation State: unverified
                Task: BGP_172.16.2.0.0+34549
                AS path: I
                Communities: target:2:1
                Import Accepted
                VPN Label: 16
                Localpref: 0
                Router ID: 10.2.0.0
                Primary Routing Table bgp.l3vpn.0
                Composite next hops: 1
                        Protocol next hop: 10.2.0.0 Metric: 35
                        Push 16
                        Composite next hop: 0x25805988 - INH Session ID: 0x193c
                        Indirect next hop: 0x23eea900 - INH Session ID: 0x193c
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.1.1.1 via ge-1/1/9.0
                                Session Id: 0x17d8
                        2.2.0.0/32 Originating RIB: inet.3
                          Metric: 35                     Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 10.1.1.1 via ge-1/1/9.0
         BGP    Preference: 170/-1
                Route Distinguisher: 2:1
                Next hop type: Indirect
                Address: 0x9347028
                Next-hop reference count: 3
                Source: 10.3.0.0
                Next hop type: Router, Next hop index: 702
                Next hop: 10.1.4.2 via ge-1/0/0.0, selected
                Label operation: Push 634278
                Label TTL action: prop-ttl
                Session Id: 0x17d9
                Protocol next hop: 10.3.0.0
                Push 16
                Composite next hop: 0x93463a0 1048575 INH Session ID: 0x17da
                Indirect next hop: 0x91e8800 1048574 INH Session ID: 0x17da
                State: <Secondary NotBest Int Ext ProtectionPath ProtectionCand>

                Inactive reason: Not Best in its group - IGP metric
                Local AS:     2 Peer AS:     2
                Age: 3:34       Metric2: 70
                Validation State: unverified
                Task: BGP_172.16.3.0.0+32805
                Announcement bits (2): 0-KRT 1-BGP_RT_Background
                AS path: I
                Communities: target:2:1
                Import Accepted
```



```
                VPN Label: 16
                Localpref: 0
                Router ID: 10.3.0.0
                Primary Routing Table bgp.l3vpn.0
                Composite next hops: 1
                        Protocol next hop: 10.3.0.0 Metric: 70
                        Push 16
                        Composite next hop: 0x93463a0 1048575 INH Session ID:
0x17da
                        Indirect next hop: 0x91e8800 1048574 INH Session ID:
0x17da
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.1.4.2 via ge-1/0/0.0
                                Session Id: 0x17d9
                        10.3.0.0/32 Originating RIB: inet.3
                          Metric: 70                      Node path count: 1
                          Forwarding nexthops: 1
                                Nexthop: 10.1.4.2 via ge-1/0/0.0
        #Multipath Preference: 255
                Next hop type: Indirect
                Address: 0x24afca30
                Next-hop reference count: 1
                Next hop type: Router
                Next hop: 10.1.1.1 via ge-1/1/9.0, selected
                Label operation: Push 707633
                Label TTL action: prop-ttl
                Session Id: 0x17d8
                Next hop type: Router, Next hop index: 702
                Next hop: 10.1.4.2 via ge-1/0/0.0
                Label operation: Push 634278
                Label TTL action: prop-ttl
                Session Id: 0x17d9
                Protocol next hop: 10.2.0.0
                Push 16
                Composite next hop: 0x25805988 - INH Session ID: 0x193c
               Indirect next hop: 0x23eea900 - INH Session ID: 0x193c Weight 0x1

                Protocol next hop: 10.3.0.0
                Push 16
                Composite next hop: 0x93463a0 1048575 INH Session ID: 0x17da
              Indirect next hop: 0x91e8800 1048574 INH Session ID: 0x17da Weight
 0x4000
                State: <ForwardingOnly Int Ext>
                Inactive reason: Forwarding use only
                Age: 23         Metric2: 35
                Validation State: unverified
                Task: RT
                AS path: I
                Communities: target:2:1
```

show route table bgp.evpn.0 extensive |no-more (EVPN)

```
show route table bgp.evpn.0 extensive | no-more
bgp.evpn.0: 6 destinations, 6 routes (6 active, 0 holddown, 0 hidden)
2:1000:10::100::00:aa:aa:aa:aa:aa/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
```

```
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
         State: Local AS: 17  Peer AS:17  Age:21:12  Metric2:1  Validation State:
 unverified
                Task: BGP_17.1.2.3.4+50756
                AS path: I
                Communities: target:1111:8388708 encapsulation0:0:0:0:3
                Import Accepted
                Route Label: 100
                ESI: 00:00:00:00:00:00:00:00:00:00
                Localpref: 100
                Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.2.3.4 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.10.10.1 via xe-0/0/1.0
                                Session Id: 0x2
                        1.2.3.4/32 Originating RIB: inet.0
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 2
                                Nexthop: 10.92.78.102 via em0.0

2:1000:10::200::00:bb:bb:bb:bb:bb/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
          State: Local AS:17  Peer AS:17  Age:19:43  Metric2:1  Validation
State:unverified
                Task: BGP_17.1.2.3.4+50756
                AS path: I
                Communities: target:2222:22 encapsulation0:0:0:0:3
                Import Accepted
                Route Label: 200
                ESI: 00:00:00:00:00:00:00:00:00:00
                Localpref: 100
                Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.2.3.4 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.10.10.1 via xe-0/0/1.0
                                Session Id: 0x2
                        10.2.3.4/32 Originating RIB: inet.0
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 2
                                Nexthop: 10.92.78.102 via em0.0

2:1000:10::300::00:cc:cc:cc:cc:cc/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
```

```
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
          State: Local AS:17  Peer AS:17  Age:17:21  Metric2:1  Validation State:
 unverified  Task: BGP 17,1,2,3,4+50756
          AS path: I
                Communities: target:3333:33 encapsulation0:0:0:0:3
                Import Accepted
                Route Label: 300
                ESI: 00:00:00:00:00:00:00:00:00:00
                Localpref: 100
                Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.2.3.4 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.10.10.1 via xe-0/0/1.0
                                Session Id: 0x2
                        10.2.3.4/32 Originating RIB: inet.0
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 2
                                Nexthop: 10.92.78.102 via em0.0

3:1000:10::100::1.2.3.4/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
                PMSI: Flags 0x0: Label 100: Type INGRESS-REPLICATION 1.2.3.4
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
          State: Local AS:17  Peer AS:17 Age:37:01  Metric2:1  Validation State:
 unverified  Task: BGP 17.1.2.3.4+50756
                AS path: I
                Communities: target:1111:8388708 encapsulation0:0:0:0:3
                Import Accepted
                Localpref: 100
                Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.2.3.4 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.10.10.1 via xe-0/0/1.0
                                Session Id: 0x2
                        10.2.3.4/32 Originating RIB: inet.0
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 2
                                Nexthop: 10.92.78.102 via em0.0

3:1000:10::200::1.2.3.4/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
```

```
                PMSI: Flags 0x0: Label 200: Type INGRESS-REPLICATION 1.2.3.4
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
          State: Local AS: 17 Peer AS: 17   Age:35:22  Metric2:1  Validation
State:unverified  Task: BGP 17.1.2.3.4+50756
           AS path:I  Communities: target:2222:22 encapsulation):0:0:0:0:3

Import Accepted
                Localpref: 100
                Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.2.3.4 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.10.10.1 via xe-0/0/1.0
                                Session Id: 0x2
                        10.2.3.4/32 Originating RIB: inet.0
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 2
                                Nexthop: 10.92.78.102 via em0.0

3:1000:10::300::1.2.3.4/304 (1 entry, 0 announced)
        *BGP    Preference: 170/-101
                Route Distinguisher: 1000:10
                PMSI: Flags 0x0: Label 300: Type INGRESS-REPLICATION 1.2.3.4
                Next hop type: Indirect
                Address: 0x9420fd0
                Next-hop reference count: 12
                Source: 10.2.3.4
                Protocol next hop: 10.2.3.4
                Indirect next hop: 0x2 no-forward INH Session ID: 0x0
         State: Local AS: 17 Peer AS:  17  Age 35:22 Metric2:1  Validation State:
 unverified   Task: BGP 17.1.2.3.4+5075
          6  AS path: I   Communities: target:3333:33  encapsulation0:0:0:0:3
Import Accepted   Localpref:100
          Router ID: 10.2.3.4
                Secondary Tables: default-switch.evpn.0
                Indirect next hops: 1
                        Protocol next hop: 10.2.3.4 Metric: 1
                        Indirect next hop: 0x2 no-forward INH Session ID: 0x0
                        Indirect path forwarding next hops: 1
                                Next hop type: Router
                                Next hop: 10.10.10.1 via xe-0/0/1.0
                                Session Id: 0x2
                        10.2.3.4/32 Originating RIB: inet.0
                          Metric: 1                       Node path count: 1
                          Forwarding nexthops: 2
                                Nexthop: 10.92.78.102 via em0.0
```


show route terse

List of Syntax

Syntax on page 1573
Syntax (EX Series Switches) on page 1573

Syntax

show route terse
<logical-system (all | *logical-system-name*)>

Syntax (EX Series Switches)

show route terse

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Display a high-level summary of the routes in the routing table.

NOTE: For BGP routes, the **show route terse** command displays the local preference attribute and MED instead of the metric1 and metric2 values. This is mostly due to historical reasons.

To display the metric1 and metric2 value of a BGP route, use the **show route extensive** command.

Options

none—Display a high-level summary of the routes in the routing table.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

Required Privilege Level

view

List of Sample Output

show route terse on page 1575

Output Fields

Table 97 on page 1573 describes the output fields for the **show route terse** command. Output fields are listed in the approximate order in which they appear.

Table 97: show route terse Output Fields

Field Name	Field Description
routing-table-name	Name of the routing table (for example, inet.0).
number destinations	Number of destinations for which there are routes in the routing table.

Table 97: show route terse Output Fields *(continued)*

Field Name	Field Description
number routes	Number of routes in the routing table and total number of routes in the following states: • **active** (routes that are active) • **holddown** (routes that are in the pending state before being declared inactive) • **hidden** (routes that are not used because of a routing policy)
route key	Key for the state of the route: • **+**—A plus sign indicates the active route, which is the route installed from the routing table into the forwarding table. • **-** —A hyphen indicates the last active route. • *****—An asterisk indicates that the route is both the active and the last active route. An asterisk before a **to** line indicates the best subpath to the route.
A	Active route. An asterisk (*) indicates this is the active route.
V	Validation status of the route: • **?**—Not evaluated. Indicates that the route was not learned through BGP. • **I**—Invalid. Indicates that the prefix is found, but either the corresponding AS received from the EBGP peer is not the AS that appears in the database, or the prefix length in the BGP update message is longer than the maximum length permitted in the database. • **N**—Unknown. Indicates that the prefix is not among the prefixes or prefix ranges in the database. • **V**—Valid. Indicates that the prefix and autonomous system pair are found in the database.
Destination	Destination of the route.
P	Protocol through which the route was learned: • **A**—Aggregate • **B**—BGP • **C**—CCC • **D**—Direct • **G**—GMPLS • **I**—IS-IS • **L**—L2CKT, L2VPN, LDP, Local • **K**—Kernel • **M**—MPLS, MSDP • **O**—OSPF • **P**—PIM • **R**—RIP, RIPng • **S**—Static • **T**—Tunnel
Prf	Preference value of the route. In every routing metric except for the BGP **LocalPref** attribute, a lesser value is preferred. In order to use common comparison routines, Junos OS stores the 1's complement of the **LocalPref** value in the **Preference2** field. For example, if the **LocalPref** value for Route 1 is 100, the **Preference2** value is -101. If the **LocalPref** value for Route 2 is 155, the **Preference2** value is -156. Route 2 is preferred because it has a higher **LocalPref** value and a lower **Preference2** value.

Table 97: show route terse Output Fields *(continued)*

Field Name	Field Description
Metric 1	First metric value in the route. For routes learned from BGP, this is the MED metric.
Metric 2	Second metric value in the route. For routes learned from BGP, this is the IGP metric.
Next hop	Next hop to the destination. An angle bracket (>) indicates that the route is the selected route.
AS path	AS path through which the route was learned. The letters at the end of the AS path indicate the path origin, providing an indication of the state of the route at the point at which the AS path originated: • **I**—IGP. • **E**—EGP. • **?**—Incomplete; typically, the AS path was aggregated.

Sample Output

show route terse

```
user@host> show route terse
inet.0: 10 destinations, 12 routes (10 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

A V Destination        P Prf   Metric 1   Metric 2  Next hop        AS path
* ? 172.16.1.1/32        O  10         1             >10.0.0.2
  ?                    B 170        100                             I
  unverified                                      >10.0.0.2
* ? 172.16.1.1/32        D   0                       >lo0.2
* V 2.2.0.2/32         B 170        110                             200 I
  valid                                           >10.0.0.2
* ? 10.0.0.0/30        D   0                      >lt-1/2/0.1
  ?                    B 170        100                             I
  unverified                                      >10.0.0.2
* ? 10.0.0.1/32        L   0                       Local
* ? 10.0.0.4/30        B 170        100                             I
  unverified                                      >10.0.0.2
* ? 10.0.0.8/30        B 170        100                             I
  unverified                                      >10.0.0.2
* I 172.16.1.1/32      B 170         90                             200 I
  invalid                                         >10.0.0.2
* N 192.168.2.3/32     B 170        100                             200 I
  unknown                                         >10.0.0.2
* ? 172.16.233.5/32      O  10         1              MultiRecv
```

show validation database

Syntax

show validation database
<brief | detail>
<instance *instance-name*>
<logical-system *logical-system-name*>
<mismatch>
<origin-autonomous-system *as-number*>
<record *ip-prefix*>
<session *ip-address*>

Release Information

Command introduced in Junos OS Release 12.2.

Description

Display information about the route validation database when resource public key infrastructure (RPKI) BGP route validation is configured. You can query all route validation records that match a given prefix or origin-autonomous-system. In addition, you can filter the output by a specific RPKI cache session.

Options

none—Display all route validation database entries.

brief | detail—(Optional) Display the specified level of output.

instance *instance-name*—(Optional) Display information about route validation database entries for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system *logical-system-name*—(Optional) Perform this operation on a particular logical system.

mismatch—(Optional) Filter the output by mismatched origin autonomous systems.

origin-autonomous-system *as-number*—(Optional) Filter the output by mismatched origin autonomous systems. The **mismatch** qualifier is useful for finding conflicting origin-autonomous-system information between RPKI caches. Mismatches might occur during cache reconfiguration.

record *ip-prefix*—(Optional) Filter the output by route validation records that match a given prefix.

session *ip-address*—(Optional) Filter the output by a specific RPKI cache session.

Required Privilege Level

view

Related Documentation

- *Use Case and Benefit of Origin Validation*
- *Understanding Origin Validation for BGP*
- *Example: Configuring Origin Validation for BGP*

List of Sample Output show validation database on page 1577

Output Fields Table 98 on page 1577 describes the output fields for the **show validation database** command. Output fields are listed in the approximate order in which they appear.

Table 98: show validation database Output Fields

Field Name	Field Description	Level of Output
Prefix	Route validation (RV) record prefix. RV records are received from the cache server and can also be configured statically at the **[edit routing-options validation static]** hierarchy level .	All levels
Origin-AS	Legitimate originator autonomous system (AS).	All levels
Session	IP address of the RPKI cache server.	All levels
State	State of the route validation records. The state can be **valid**, **invalid** or **unknown**.	All levels
Mismatch	Conflicting origin-autonomous-system information between RPKI caches when nonstop active routing (NSR) is configured.	All levels
IPv4 records	Number of IPv4 route validation records.	All levels
IPv6 records	Number of IPv6 route validation records.	All levels

Sample Output

show validation database

```
user@host> show validation database
 RV database for instance master

  Prefix                 Origin-AS Session         State   Mismatch
  172.16.1.0/24-32               1 10.0.77.1          valid
  172.16.2.0/24-32               2 10.0.77.1          valid
  172.16.3.0/24-32               3 10.0.77.1          valid
  172.16.4.0/24-32               4 10.0.77.1          valid
  172.16.5.0/24-32               5 10.0.77.1          valid
  172.16.6.0/24-32               6 10.0.77.1          valid
  172.16.7.0/24-32               7 10.0.77.1          valid
  172.16.8.0/24-32               8 10.0.77.1          valid
  72.9.224.0/19-24           26234 192.168.1.100   valid    *
  72.9.224.0/19-24            3320 192.168.1.200   invalid *
  10.0.0.0/8-32                  0 internal        valid

  IPv4 records: 14
  IPv6 records: 0
```

show validation group

Syntax

show validation group
<instance *instance-name*>
<logical-system *logical-system-name*>

Release Information

Command introduced in Junos OS Release 12.2.

Description

Display information about route validation redundancy groups.

Options

none—Display information about all route validation groups.

instance *instance-name*—(Optional) Display information about route validation groups for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system *logical-system-name*—(Optional) Perform this operation on a particular logical system.

Required Privilege Level

view

Related Documentation

- *Use Case and Benefit of Origin Validation*
- *Understanding Origin Validation for BGP*
- *Example: Configuring Origin Validation for BGP*

List of Sample Output

show validation group on page 1579

Output Fields

Table 99 on page 1578 describes the output fields for the **show validation group** command. Output fields are listed in the approximate order in which they appear.

Table 99: show validation group Output Fields

Field Name	Field Description
Group	Group name.
Maximum sessions	Number of concurrent sessions for each group. The default is 2. The number is configurable with the **max-sessions** statement.
Session	Resource public key infrastructure (RPKI) cache session IP address.
State	State of the connection between the routing device and the cache server. **Up** means that the connection is established. **Connect** means that the connection is not established.

Table 99: show validation group Output Fields *(continued)*

Field Name	Field Description
Preference	Each cache server has a preference. Higher preferences are preferred. During a session start or restart, the routing device attempts to start a session with the cache server that has the numerically highest preference. The routing device connects to multiple cache servers in preference order. The default preference is 100. The preference is configurable with the **preference** statement.

Sample Output

show validation group

```
user@host> show validation group
master
    Group: test, Maximum sessions: 3
      Session 10.255.255.11, State: Up, Preference: 100
      Session 10.255.255.12, State: Up, Preference: 100
    Group: test2, Maximum sessions: 2
      Session 10.255.255.13, State: Connect, Preference: 100
```

show validation replication database

Syntax

show validation replication database
<brief | detail>
<instance *instance-name*>
<logical-system *logical-system-name*>
<origin-autonomous-system *as-number*>
<record *ip-prefix*>
<session *ip-address*>

Release Information

Command introduced in Junos OS Release 12.2.

Description

Display the state of the nonstop active routing (NSR) records. The output is the same as the output of the show validation database command, except for the **Mismatch** column.

Options

none—Display all route validation database entries.

brief | detail—(Optional) Display the specified level of output.

instance *instance-name*—(Optional) Display information about route validation database entries for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system *logical-system-name*—(Optional) Perform this operation on a particular logical system.

origin-autonomous-system *as-number*—(Optional) Filter the output by mismatched origin autonomous systems. The **mismatch** qualifier is useful for finding conflicting origin-autonomous-system information between resource public key infrastructure (RPKI) caches. Mismatches might occur during cache reconfiguration.

record *ip-prefix*—(Optional) Filter the output by route validation records that match a given prefix.

session *ip-address*—(Optional) Filter the output by a specific RPKI cache session.

Required Privilege Level

view

Related Documentation

- *Use Case and Benefit of Origin Validation*
- *Understanding Origin Validation for BGP*
- *Example: Configuring Origin Validation for BGP*

List of Sample Output

show validation replication database on page 1581

Output Fields Table 100 on page 1581 describes the output fields for the **show validation replication database** command. Output fields are listed in the approximate order in which they appear.

Table 100: show validation replication database Output Fields

Field Name	Field Description	Level of Output
Prefix	Route validation (RV) record prefix. RV records are received from the cache server and can also be configured statically at the **[edit routing-options validation static]** hierarchy level.	All levels
Origin-AS	Legitimate originator autonomous system (AS).	All levels
Session	IP address of the RPKI cache server.	All levels
State	State of the route validation records. The state can be **valid** or **invalid**.	All levels
IPv4 records	Number of IPv4 route validation records.	All levels
IPv6 records	Number of IPv6 route validation records.	All levels

Sample Output

show validation replication database

```
user@host> show validation replication database
RV database for instance master

  Prefix                Origin-AS Session         State
  172.16.1.0/24-32              1 10.0.77.1          valid
  172.16.2.0/24-32              2 10.0.77.1          valid
  172.16.3.0/24-32              3 10.0.77.1          valid
  172.16.4.0/24-32              4 10.0.77.1          valid
  172.16.5.0/24-32              5 10.0.77.1          valid
  172.16.6.0/24-32              6 10.0.77.1          valid
  172.16.7.0/24-32              7 10.0.77.1          valid
  172.16.8.0/24-32              8 10.0.77.1          valid
  72.9.224.0/19-24          26234 192.168.1.100   valid
  72.9.224.0/19-24           3320 192.168.1.200   invalid
  10.0.0.0/8-32                 0 internal        valid

  IPv4 records: 14
  IPv6 records: 0
```


show validation session

Syntax

show validation session
<brief | detail>
<*destination*>
<instance *instance-name*>
<logical-system *logical-system-name*>

Release Information

Command introduced in Junos OS Release 12.2.

Description

Display information about all sessions or a specific session with a resource public key infrastructure (RPKI) cache server.

Options

none—Display information about all sessions.

destination—(Optional) Display information about a specific session.

brief | detail—(Optional) Display the specified level of output.

instance ***instance-name***—(Optional) Display information about sessions for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system ***logical-system-name***—(Optional) Perform this operation on a particular logical system.

Required Privilege Level

view

Related Documentation

- *Use Case and Benefit of Origin Validation*
- *Understanding Origin Validation for BGP*
- *Example: Configuring Origin Validation for BGP*

List of Sample Output

show validation session brief on page 1584
show validation session detail on page 1584

Output Fields

Table 101 on page 1582 describes the output fields for the **show validation session** command. Output fields are listed in the approximate order in which they appear.

Table 101: show validation session Output Fields

Field Name	Field Description	Level of Output
Session	IP address of the RPKI cache server.	All levels

Table 101: show validation session Output Fields *(continued)*

Field Name	Field Description	Level of Output
State	State of the connection between the routing device and the cache server. **Up** means that the connection is established. **Connect** means that the connection is not established.	All levels
Flaps	Number of attempts to establish a session.	None and **brief**
Uptime	Length of time that the session has remained establshed.	None and **brief**
#IPv4/IPv6 records	Number of IPv4 and IPv6 route validation records.	None and **brief**
Session index	Every session has an index number.	**detail**
Group	Name of the group to which the session belongs	**detail**
Preference	Each cache server has a preference. Higher preferences are preferred. During a session start or restart, the routing device attempts to start a session with the cache server that has the numerically highest preference. The routing device connects to multiple cache servers in preference order. The default preference is 100. The preference is configurable with the **preference** statement.	**detail**
Port	TCP port number for the outgoing connection with the cache server. The well-known RPKI port is TCP port 2222. For a given deployment, an RPKI cache server might listen on some other TCP port number. If so, you can configure the alternative port number with the **port** statement.	**detail**
Refresh time	Liveliness check interval for an RPKI cache server. Every**refresh-time** (seconds), a serial query protocol data unit (PDU) with the last known serial number is transmitted. The **hold-time** must be at least 2 x the **refresh-time**.	**detail**
Hold time	Length of time in seconds that the session between the routing device and the cache server is considered operational without any activity. After the hold time expires, the session is dropped. Reception of any PDU from the cache server resets the hold timer. The **hold-time** is 600 seconds, by default, and must be least 2 x the **refresh-time**. If the hold time expires, the session is considered to be down. This, in turn, triggers a session restart event. During a session restart, the routing device attempts to start a session with the cache server that has the numerically highest **preference**.	**detail**

Table 101: show validation session Output Fields *(continued)*

Field Name	Field Description	Level of Output
Record Life time	Amount of time that route validation (RV) records learned from a cache server are valid. RV records expire if the session to the cache server goes down and remains down for the **record-lifetime** (seconds).	**detail**
Serial (Full Update)	Number of full serial updates.	**detail**
Serial (Incremental Update)	Number of incremental serial updates.	**detail**
Session flaps	Number of attempts to establish a session.	**detail**
Session uptime	Length of time that the session has remained establshed.	**detail**
Last PDU received	Time when the most recent PDU was received.	**detail**
IPv4 prefix count	Number of IPv4 sessions.	**detail**
IPv6 prefix count	Number of IPv6 sessions.	**detail**

Sample Output

show validation session brief

```
user@host> show validation session brief
 Session                                 State    Flaps     Uptime #IPv4/IPv6
records
  1.3.0.2                                 up          2    00:01:37 13/0
  10.255.255.11                           up          3    00:00:01 1/0
  10.255.255.12                           connect     2             64/68
```

show validation session detail

```
user@host> show validation session detail
Session 10.0.77.1, State: up
    Group: test, Preference: 100
    Local IPv4 address: 10.0.77.2, Port: 2222
    Refresh time: 300s
    Session flaps: 14, Last Session flap: 5h13m18s ago
    Hold time: 900s
    Record Life time: 3600s
    Serial (Full Update): 0
    Serial (Incremental Update): 0
      Session flaps 2
      Session uptime: 00:48:35
      Last PDU received: 00:03:35
      IPv4 prefix count: 71234
      IPv6 prefix count: 345
```


show validation statistics

Syntax

show validation statistics
<instance *instance-name*>
<logical-system *logical-system-name*>

Release Information

Command introduced in Junos OS Release 12.2.

Description

Display route validation statistics.

Options

none—Display statistics for all routing instances.

instance *instance-name*—(Optional) Display information for the specified routing instance. The instance name can be master for the main instance, or any valid configured instance name or its prefix.

logical-system *logical-system-name*—(Optional) Perform this operation on a particular logical system.

Required Privilege Level

view

Related Documentation

- *Use Case and Benefit of Origin Validation*
- *Understanding Origin Validation for BGP*
- *Example: Configuring Origin Validation for BGP*

List of Sample Output

show validation statistics on page 1586

Output Fields

Table 102 on page 1585 describes the output fields for the **show validation statistics** command. Output fields are listed in the approximate order in which they appear.

Table 102: show validation statistics Output Fields

Field Name	Field Description
Total RV records	Group name.
Total Replication RV records	Number of concurrent sessions for each group. The default is 2. The number is configurable with the **max-sessions** statement.
Prefix entries	Resource public key infrastructure (RPKI) cache session IP address.
Origin-AS entries	State of the connection between the routing device and the cache server. **Up** means that the connection is up. **Connect** means that the connection is not up.

Table 102: show validation statistics Output Fields *(continued)*

Field Name	Field Description
Memory utilization	Each cache server has a preference. Higher preferences are preferred. During a session start or restart, the routing device attempts to start a session with the cache server that has the numerically highest preference. The routing device connects to multiple cache servers in preference order. The default preference is 100. The preference is configurable with the **preference** statement.
Policy origin-validation requests	Number of queries for validation state of a given instance and prefix.
Valid	Number of valid prefixes reported by the validation query.
Invalid	Number of invalid prefixes reported by the validation query.
Unknown	Number of unknown prefixes reported by the validation query. This means that the prefix is not found in the database.
BGP import policy reevaluation notifications	A change, addition, or deletion of a route validation record triggers a BGP import reevaluation for all exact matching and more specific prefixes.
inet.0	Number of IPv4 route validation records that have been added, deleted, or changed.
inet6.0	Number of IPv6 route validation records that have been added, deleted, or changed.

Sample Output

show validation statistics

```
user@host> show validation statistics
Total RV records:        453455
  Total Replication RV records: 453455
    Prefix entries:          35432
    Origin-AS entries:      124400
  Memory utilization: 16.31MB
  Policy origin-validation requests: 234995
    valid:                   23445
    invalid:                 14666
    unknown:                 34567
  BGP import policy reevaluation notifications: 460268
    inet.0:                 435345
    inet6.0:                  3454
```

test policy

Syntax

test policy *policy-name prefix*

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.

Description

Test a policy configuration to determine which prefixes match routes in the routing table.

NOTE: If you are using the **test policy** command on a logical system, you must first set the CLI to the logical system context. For example, if you want to test a routing policy that is configured on logical system R2, first run the **set cli logical-system R2** command.

Options

policy-name—Name of a policy.

prefix—Destination prefix to match.

Additional Information

All prefixes in the default unicast routing table (inet.0) that match prefixes that are the same as or longer than the specific prefix are processed by the **from** clause in the specified policy. All prefixes accepted by the policy are displayed. The **test policy** command evaluates a policy differently from the BGP import process. When testing a policy that contains an **interface** match condition in the **from** clause, the **test policy** command uses the match condition. In contrast, BGP does not use the **interface** match condition when evaluating the policy against routes learned from internal BGP (IBGP) or external BGP (EGBP) multihop peers.

When testing a policy, you can see the length of time (in microseconds) required to evaluate the policy and the number of times it has been executed by running the show policy ***policy-name*** **statistics** command.

Required Privilege Level

view

Related Documentation

- Understanding Routing Policy Tests on page 543
- Example: Testing a Routing Policy with Complex Regular Expressions on page 544
- show policy on page 1364

List of Sample Output

test policy on page 1588

Output Fields For information about output fields, see the output field tables for the show route command, the show route detail command, the show route extensive command, or the show route terse command.

Sample Output

test policy

```
user@host> test policy test-statics 172.16.0.1/8
inet.0: 44 destinations, 44 routes (44 active, 0 holddown, 0 hidden)
Prefixes passing policy:

172.16.3.0/8          *[BGP/170] 16:22:46, localpref 100, from 10.255.255.41
                    AS Path: 50888 I
                     > to 10.11.4.32 via en0.2, label-switched-path l2
172.16.3.1/32         *[IS-IS/18] 2d 00:21:46, metric 0, tag 2
                     > to 10.0.4.7 via fxp0.0
172.16.3.2/32         *[IS-IS/18] 2d 00:21:46, metric 0, tag 2
                     > to 10.0.4.7 via fxp0.0
172.16.3.3/32         *[IS-IS/18] 2d 00:21:46, metric 0, tag 2
                     > to 10.0.4.7 via fxp0.0
172.16.3.4/32         *[IS-IS/18] 2d 00:21:46, metric 0, tag 2
                     > to 10.0.4.7 via fxp0.0
Policy test-statics: 5 prefixes accepted, 0 prefixes rejected
```

Traffic Policer Operational Commands

- clear firewall
- show firewall
- show firewall filter version
- show firewall log
- show firewall prefix-action-stats
- show interfaces policers
- show policer

clear firewall

List of Syntax

Syntax on page 1589
Syntax (EX Series Switches) on page 1589

Syntax

clear firewall (all | counter *counter-name* | filter *filter-name* | log (all | *logical-system-name*) | logical-system *logical-system-name*)

Syntax (EX Series Switches)

clear firewall (all | counter *counter-name* | filter *filter-name* | log (all | *logical-system-name*) | policer counter (all | counter-id *counter-index*))

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
logical-system option introduced in Junos OS Release 9.3.
log option introduced before Junos OS Release 11.4.

Description

Clear statistics about configured firewall filters.

When you clear the counters of a filter, this impacts not only the counters shown by the CLI, but also the ones tracked by SNMP2.

Subscriber management uses firewall filters to capture and report the volume-based service accounting counters that are used for subscriber billing. The **clear firewall** command also clears the service accounting counters that are reported to the RADIUS accounting server. For this reason, you must be cautious in specifying which firewall statistics you want to clear.

NOTE: The clear firewall command cannot be used to clear the Routing Engine filter counters on a backup Routing Engine that is enabled for graceful Routing Engine switchover (GRES).

If you clear statistics for firewall filters that are applied to Trio-based DPCs and that also use the **prefix-action** action on matched packets, wait at least 5 seconds before you enter the **show firewall prefix-action-stats** command. A 5-second pause between issuing the **clear firewall** and **show firewall prefix-action-stats** commands avoids a possible timeout of the **show firewall prefix-action-stats** command.

Options

all—Clear the packet and byte counts for all filters. On EX Series switches, this option also clears the packet counts for all policer counters.

counter *counter-name*—Clear the packet and byte counts for a filter counter that has been configured with the counter firewall filter action.

filter *filter-name*—Clear the packet and byte counts for the specified firewall filter.

log (all | *logical-system-name*)—Clear log entries for IPv4 firewall filters that have **then log** as an action. Use **log all** to clear all log entries or **log *logical-system-name*** to clear log entries for the specified logical system.

logical-system *logical-system-name*—Clear the packet and byte counts for the specified logical system.

policer counter (all | counter-id *counter-index*)—(EX8200 switches only) Clear all policer counters using the **policer counter all** command, or clear a specific policer counter using the **policer counter counter-id *counter-index*** command. The value of *counter-index* can be 0, 1, or 2.

Required Privilege Level

clear

Related Documentation

- show firewall on page 1591

List of Sample Output

clear firewall all on page 1590
clear firewall (counter counter-name) on page 1590
clear firewall (filter filter-name) on page 1590
clear firewall (policer counter all) (EX8200 Switch) on page 1590
clear firewall (policer counter counter-id counter-index) (EX8200 Switch) on page 1590

Sample Output

clear firewall all

```
user@host> clear firewall all
```

clear firewall (counter counter-name)

```
user@host> clear firewall counter port-filter-counter
```

clear firewall (filter filter-name)

```
user@host> clear firewall filter ingress-port-filter
```

clear firewall (policer counter all) (EX8200 Switch)

```
user@switch> clear firewall policer counter all
```

clear firewall (policer counter counter-id counter-index) (EX8200 Switch)

```
user@switch> clear firewall policer counter counter-id 0
```

show firewall

List of Syntax

Syntax on page 1591
Syntax (EX Series Switches) on page 1591

Syntax

show firewall
<application (CFM | eswd | RMPS)>>
<counter *counter-name*>
<detail>
<filter *filter-name*>
<filter regex *regular-expression*>
<logical-system (all | *logical-system-name*)>
<terse>

Syntax (EX Series Switches)

show firewall
<application (CFM | eswd | RMPS)>>
<counter *counter-name*>
<detail>
<filter *filter-name*>
<filter regex *regular-expression*>
<log <(detail | interface *interface-name*)>>
<policer counters <(detail | counter-id *counter-index* <detail>)>>
<terse>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
Option **logical-system** introduced in Junos OS Release 9.3.
Option **terse** introduced in Junos OS Release 9.4.
Option **policer counters** introduced in Junos OS Release 12.2 for EX Series switches.
Option **detail** introduced in Junos OS Release 12.3 for EX Series switches.
Option **detail** introduced in Junos OS Release 14.1 for MX Series routers.
Option **regex *regular-expression*** introduced in Junos OS Release 14.2.

Description

Display enhanced statistics and counters for all configured firewall filters.

Options

none—(Optional) Display statistics and counters for all configured firewall filters and counters. For EX Series switches, this command also displays statistics about all configured policers.

application (CFM | eswd | RMPS)—(Optional) Show firewall elements owned by the selected software component:

- Connectivity Fault Management (CFM)
- Ethernet switching daemon (eswd)—Shows only on devices that support it.
- Resource Management and Packet Steering (RMPS)

counter *counter-name*—(Optional) Name of a filter counter.

detail—(EX Series switches and MX Series routers only) (Optional) Display firewall filter statistics and enhanced policer statistics and counters.

filter *filter-name*—(Optional) Name of a configured filter.

filter regex *regular-expression*—(Optional) Regular expression that matches the names of a subset of filters.

logical-system (all | *logical-system-name*)—(Optional) Perform this operation on all logical systems or on a particular logical system.

log—(Optional) Display log entries for firewall filters.

log <(detail | interface *interface-name*)>—(EX Series switches only) (Optional) Display detailed log entries of firewall activity or log information about a specific interface.

policer counters <(detail | counter-id *counter-index* <detail>)>—(EX8200 switches only) (Optional) Display enhanced policer counter statistics in brief or in detail.

terse—(Optional) Display firewall filter names only.

Required Privilege Level

view

Related Documentation

- clear firewall on page 1589
- show firewall log on page 1600
- *Verifying That Firewall Filters Are Operational*
- *Verifying That Policers Are Operational*
- show policer on page 1607
- *Enhanced Policer Statistics Overview*
- *enhanced-policer*

List of Sample Output

Output Fields

Table 103 on page 1593 lists the output fields for the **show firewall** command. Output fields are listed in the approximate order in which they appear.

Table 103: show firewall Output Fields

Field Name	Field Description
Filter	Name of a filter that has been configured with the **filter** statement at the **[edit firewall]** hierarchy level. Except on EX Series switches: • When an interface-specific filter is displayed, the name of the filter is followed by the full interface name and by either **-i** for an input filter or **-o** for an output filter. • When dynamic filters are displayed, the name of the filter is followed by the full interface name and by either **-in** for an input filter or **-out** for an output filter. When a logical system–specific filter is displayed, the name of the filter is prefixed with two underscore (__) characters and the name of the logical system (for example, **__ls1/filter1**). • When a service filter is displayed that uses a service set, the separator between the service-set name and the service-filter name is a semicolon (:). NOTE: For **bridge family filter**, the **ip-protocol** match criteria is supported only for IPv4 and not for IPv6. This is applicable for line cards that support the Junos Trio chipset, such as the MX 3D MPC line cards.
Counters	Display filter counter information: • **Name**—Name of a filter counter that has been configured with the **counter** firewall filter action. • **Bytes**—Number of bytes that match the filter term under which the **counter** action is specified. • **Packets**—Number of packets that matched the filter term under which the **counter** action is specified. NOTE: On M and T Series routers, firewall filters cannot count **ip-options** packets on a per option type and per interface basis. A limited work around is to use the **show pfe statistics ip options** command to see **ip-options** statistics on a per Packet Forwarding Engine (PFE) basis. See *show pfe statistics ip* for sample output.
Policers	Display policer information: • **Name**—Name of policer. • **Bytes**—(For two-color policers on MX Series routers and EX Series switches, and for hierarchical policers on interfaces hosted on MICs and MPCs in MX Series routers) Number of bytes that match the filter term under which the policer action is specified. This is only the number out-of-specification (out-of-spec) byte counts, not all the bytes in all packets policed by the policer. For other combinations of policer type, device, and line card type, this field is blank. • **Packets**—Number of packets that matched the filter term under which the policer action is specified. This is only the number of out-of-specification (out-of-spec) packet counts, not all packets policed by the policer.
Policer Counter Index	(EX8200 switch only) Global management counter ID. The counter ID value (*counter-index*) can be 0, 1, or 2.
Green	(EX8200 switch only) Number of packets within the limits. The number of packets is smaller than the committed information rate (CIR).
Yellow	(EX8200 switch only) Number of packets partially within the limits. The number of packets is greater than the CIR, but the burst size is within the excess burst size (EBS) limit.

Table 103: show firewall Output Fields *(continued)*

Field Name	Field Description
Discard	(EX8200 switch only) Number of discarded packets.
Bytes	(EX8200 switch only) Number of green, yellow, red, or discarded packets in bytes.
Packets	(EX8200 switch only) Number of green, yellow, red, or discarded packets.
Filter name	(EX8200 switch only) Name of the filter with a term associated to a policer.
Term name	(EX8200 switch only) Name of the term associated with a policer.
Policer name	(EX8200 switch only) Name of the policer that is associated with a global management counter.
P1-t1	• OOS packet statistics for packets that are marked out-of-specification (out-of-spec) by the policer. Changes to all packets that have out-of-spec actions, such as discard, color marking, or forwarding-class, are included in this counter. • Offered packet statistics for traffic subjected to policing. • Transmitted packet statistics for traffic that is not discarded by the policer. When the policer action is discard, the statistics are the same as the in-spec statistics; when the policer action is non-discard (loss-priority or forwarding-class), the statistics are included in this counter.

Sample Output

show firewall filter (MX Series Router and EX Series Switch)

```
user@host> show firewall filter test
Filter: test
Counters:
Name                                Bytes                Packets
Counter-1                               0                      0
Counter-2                               0                      0
Policers:
Name                                Bytes                Packets
Policer-1                            2770                     70
```

show firewall filter (non MX Series Router and EX Series Switch)

```
user@host> show firewall filter test
Filter: test
Counters:
Name                                Bytes                Packets
Counter-1                               0                      0
Counter-2                               0                      0
Policers:
Name                                Bytes                Packets
Policer-1                                                     70
```

show firewall filter (Dynamic Input Filter)

```
user@host> show firewall filter dfwd-ge-5/0/0.1-in
Filter: dfwd-ge-5/0/0.1-in
Counters:
Name                                                Bytes                Packets
c1-ge-5/0/0.1-in                                        0                      0
```

show firewall (Logical Systems)

```
user@host> show firewall

Filter: __lr1/test
Counters:
Name                                                Bytes                Packets
icmp                                                  420                      5
Filter: __default_bpdu_filter__
Filter: __lr1/inet_filter1
Counters:
Name                                                Bytes                Packets
inet_tcp_count                                          0                      0
inet_udp_count                                          0                      0
Filter: __lr1/inet_filter2
Counters:
Name                                                Bytes                Packets
inet_icmp_count                                         0                      0
inet_pim_count                                          0                      0
Filter: __lr2/inet_filter1
Counters:
Name                                                Bytes                Packets
inet_tcp_count                                          0                      0
```



```
inet_udp_count                                        0                    0
```

show firewall (counter counter-name)

```
user@host> show firewall counter icmp-counter
Filter: ingress-port-voip-class-filter
Counters:
Name                                              Bytes              Packets
icmp-counter                                          0                    0
```

show firewall log

```
user@host> show firewall log
Log :

Time      Filter    Action Interface     Protocol        Src Addr
          Dest Addr
08:00:53  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
        192.168.3.4
08:00:52  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
        192.168.3.4
08:00:51  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
        192.168.3.4
08:00:50  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
        192.168.3.4
08:00:49  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
        192.168.3.4
08:00:48  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
        192.168.3.4
08:00:47  pfe       R      ge-1/0/1.0    ICMP            192.168.3.5
        192.168.3.4
```

show firewall policer counters (EX8200 Switch)

```
user@switch> show firewall policer counters
Policer Counter Index 0:
                          Bytes            Packets
Green:                       73              15914
Yellow:                       9               1962
Discard:                    119              25942

Policer Counter Index 1:
                           Bytes            Packets
Green:                         0                  0
Yellow:                        0                  0
Discard:                       0                  0

Policer Counter Index 2:
                           Bytes            Packets
Green:                         0                  0
Yellow:                        0                  0
Discard:                       0                  0
```

show firewall policer counters (detail) (EX8200 Switch)

```
user@switch> show firewall policer counters detail
Policer Counter Index 0:
                          Bytes            Packets
```



```
Green:                        73              15914
Yellow:                        9               1962
Discard:                     119              25942

Filter name                   Term name            Policer name
myfilter                      polcr-term-1         myfilter-polcr-1
inet-filter-ae                ae-snmp              policer-1
inet-filter-ae                ae-ssh               policer-2

Policer Counter Index 1:
                             Bytes            Packets
Green:                         0                  0
Yellow:                        0                  0
Discard:                       0                  0

Filter name                   Term name            Policer name

Policer Counter Index 2:
                             Bytes            Packets
Green:                         0                  0
Yellow:                        0                  0
Discard:                       0                  0

Filter name                   Term name            Policer name
```

show firewall policer counters (counter-id counter-index) (EX8200 Switch)

```
user@switch> show firewall policer counters counter-id 0
Policer Counter Index 0:
                             Bytes            Packets
Green:                        73              15914
Yellow:                        9               1962
Discard:                     119              25942
```

show firewall policer counters (counter-id counter-index detail) (EX8200 Switch)

```
user@switch> show firewall policer counters counter-id 0 detail
Policer Counter Index 0:
                             Bytes            Packets
Green:                        73              15914
Yellow:                        9               1962
Discard:                     119              25942

Filter name                   Term name            Policer name
myfilter                      polcr-term-1         myfilter-polcr-1
inet-filter-ae                ae-snmp              policer-1
inet-filter-ae                ae-ssh               policer-2
```

show firewall detail

```
user@host> show firewall detail
Filter: __default_bpdu_filter__

Filter: foo
Counters:
Name                                                 Bytes              Packets
c1                                                17652140               160474
Policers:
Name                                                 Bytes              Packets
P1-t1
```

```
OOS                                                 0                18286
Offered                                             0 18446744073709376546
Transmitted                                         0 18446744073709358260
```

show firewall filter version

Syntax show firewall filter version <*filter-name*>

Release Information Command introduced in Junos OS Release 10.2R2.

Description Display the version number of the installed firewall filter in the Routing Engine.

Options none—(Optional) Display the version number of all installed firewall filters.

filter-name—(Optional) Name of a configured filter. If you specify the name of a filter, only the version number of that filter is displayed.

Additional Information The initial version number is 1. This number increments by one when you modify the firewall filter settings or an associated prefix action. The maximum version number is 4,294,967,295. When the version number reaches 4,294,967,295, this number is reset to 1.

Required Privilege Level view

List of Sample Output show firewall filter version on page 1599

Output Fields Table 104 on page 1599 lists the output fields for the **show firewall filter version** command. Output fields are listed in the approximate order in which they appear.

Table 104: show firewall filter version Output Fields

Field Name	Field Description
Filter	Name of a filter that has been configured with the **filter** statement at the **[edit firewall]** hierarchy level.
Version	Display the version number of the firewall filter.

Sample Output

show firewall filter version

```
user@host> show firewall filter version
Filter version information :
Filter                                             Version
test                                                    10
```

show firewall log

List of Syntax

Syntax on page 1600
Syntax (EX Series Switches) on page 1600

Syntax

show firewall log
<detail>
<extensive>
<interface *interface-name*>
<logical-system (*logical-system-name* | all)>

Syntax (EX Series Switches)

show firewall log
<detail>
<interface *interface-name*>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced in Junos OS Release 9.0 for EX Series switches.
extensive option introduced in Junos OS Release 16.1.
logical-system option introduced in Junos OS Release 9.3.

Description

Display log information about firewall filters.

Options

none—Display log information about firewall filters.

detail—(Optional) Display detailed information.

extensive—(Optional) Display hex dump of packet captured by log action.

interface *interface-name*—(Optional) Display log information about a specific interface.

logical-system (*logical-system-name* | all)—(Optional) Perform this operation on all logical systems or on a particular system.

Required Privilege Level

view

List of Sample Output

show firewall log on page 1601
show firewall log detail on page 1601
show firewall log extensive on page 1602

Output Fields

Table 105 on page 1600 lists the output fields for the **show firewall log** command. Output fields are listed in the approximate order in which they appear.

Table 105: show firewall log Output Fields

Field Name	Field Description
Time of Log	Time that the event occurred.

Table 105: show firewall log Output Fields *(continued)*

Field Name	Field Description
Filter	• Displays the name of a configured firewall filter or service filter only if the packet hit the filter's **log** action in a kernel filter (in the control plane). For any traffic that reaches the Routing Engine, the packets hit the **log** action in the kernel. • For all other logged packets (packet hit the filter's **log** action in the Packet Forwarding Engine), this field displays **pfe** instead of a configured filter name.
Filter Action	Filter action: • **A**—Accept • **D**—Discard • **R**—Reject
Name of Interface	• Displays a physical interface name if the packet arrived at a port on a line card. • Displays **local** if the packet was generated by the device's internal Ethernet interface, **em1** or **fxp1**, which connects the Routing Engine with the router's packet-forwarding components.
Name of protocol	Packet's protocol name: **egp**, **gre**, **icmp**, **ipip**, **ospf**, **pim**, **rsvp**, **tcp**, or **udp**.
Packet length	Length of the packet.
Source address	Packet's source address.
Destination address	Packet's destination address and port.

Sample Output

show firewall log

```
user@host>show firewall log
Time       Filter     Action Interface     Protocol  Src Addr      Dest Addr

13:10:12   pfe        D      rlsq0.902     ICMP      192.0.2.2   192.0.2.1

13:10:11   pfe        D      rlsq0.902     ICMP      192.0.2.2   192.0.2.1
```

show firewall log detail

```
user@host> show firewall log detail
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0Name of protocol: TCP, Packet Length: 50824, Source address:
203.0.113.108:829,
Destination address: 192.168.70.66:513
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
Name of protocol: TCP, Packet Length: 1020, Source address: 203.0.113.108:829,
Destination address: 192.168.70.66:513
```

```
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
Name of protocol: TCP, Packet Length: 49245, Source address: 203.0.113.108:829,
Destination address: 192.168.70.66:513
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
Name of protocol: TCP, Packet Length: 49245, Source address: 203.0.113.108:829,
Destination address: 192.168.70.66:513
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
Name of protocol: TCP, Packet Length: 49245, Source address: 203.0.113.108:829,
Destination address: 192.168.70.66:513
Time of Log: 2004-10-13 10:37:17 PDT, Filter: f, Filter action: accept, Name of
interface: fxp0.0
Name of protocol: TCP, Packet Length: 49245, Source address: 203.0.113.108:829,
Destination address: 192.168.70.66:513
....
```

show firewall log extensive

```
user@host> show firewall log extensive
Time of Log: 2016-01-17 22:16:21 PST, Filter: pfe, Filter action: accept, Name
of interface: xe-0/0/1.0
Name of protocol: UDP, Packet Length: 98, Source address: 203.0.113.1, Destination
 address: 203.0.113.1
 :  00-0F: 00 01 03 ee ee ff 00 01 - 09 22 55 ee 81 00 02 58
 :  10-1F: 08 00 45 00 00 62 00 00 - 00 00 40 11 77 8a 01 00
 :  20-2F: 00 01 02 00 00 01 1c 00 - 1c 00 00 4e 19 83 00 01
 :  30-3F: 02 03 04 05 06 07 08 09 - 0a 0b 0c 0d 0e 0f 10 11
 :  40-4F: 12 13 14 15 16 17 18 19 - 1a 1b 1c 1d 1e 1f 20 21
 :  50-5F: 22 23 24 25 26 27 28 29 - 2a 2b 00 00 00 00 00 00
 :  60-6F: 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
 :  70-7F: 00 00 00 00 00 00 00 00 - 00 00 00 00 00 00 00 00
```


show firewall prefix-action-stats

List of Syntax

Syntax (filter-specific mode) on page 1603
Syntax (term-specific mode) on page 1603

Syntax (filter-specific mode)

show firewall prefix-action-stats filter *filter-name* prefix-action *prefix-action-name*
<from *number* to *number*>
<logical-system (*logical-system-name* | all)>

Syntax (term-specific mode)

show firewall prefix-action-stats filter *filter-name* prefix-action *prefix-action-name-term-name*
<from *number* to *number*>
<logical-system (*logical-system-name* | all)>

Release Information

Command introduced before Junos OS Release 7.4.
logical-system option introduced in Junos OS Release 9.3.

Description

Display prefix action statistics about configured firewall filters.

If you clear statistics for firewall filters that are applied to MPCs and that also use the **prefix-action** action on matched packets, wait at least 5 seconds before you enter the **show firewall prefix-action-stats** command. A 5-second pause between issuing the **clear firewall** and **show firewall prefix-action-stats** commands avoids a possible timeout of the **show firewall prefix-action-stats** command.

By default, policers operate in *term-specific* mode.

See “Filter-Specific Policer Overview” on page 1060 for information about how to configure policers in *filter-specific* mode.

Options

filter *filter-name*—Name of a filter.

prefix-action *prefix-action-name*—Name of a prefix action.

from *number* to *number*—(Optional) Starting and ending counter or policer.

logical-system (*logical-system-name* | all)—(Optional) Perform this operation on all logical systems or on a particular system.

Required Privilege Level

view

Related Documentation

- clear firewall on page 1589

List of Sample Output

show firewall prefix-action-stats on page 1604

Output Fields

Table 106 on page 1604 lists the output fields for the **show firewall prefix-action-stats** command. Output fields are listed in the approximate order in which they appear.

Table 106: show firewall prefix-action-stats Output Fields

Field Name	Field Description
Filter	Filter name. Filters configured for logical systems include the name of the filter prefixed with the two underscore characters (__) and the name of the logical system (for example, **__ls1/filter1**).

Sample Output

The following sample output assumes that the policer *act1* is in term mode and that there is a term named *term1* configured in the firewall filter *test.*

show firewall prefix-action-stats

```
user@host> show firewall prefix-action-stats filter test prefix-action act1-term1 from 0 to 9
Filter: test
Counters:
Name                      Bytes              Packets
act1-0                       0                    0
act1-1                       0                    0
act1-2                       0                    0
act1-3                       0                    0
act1-4                       0                    0
act1-5                       0                    0
act1-6                       0                    0
act1-7                       0                    0
act1-8                       0                    0
act1-9                       0                    0
Policers:
Name                      Bytes              Packets
act1-0                       0                    0
act1-1                       0                    0
act1-2                       0                    0
act1-3                       0                    0
act1-4                       0                    0
act1-5                       0                    0
act1-6                       0                    0
act1-7                       0                    0
act1-8                       0                    0
act1-9                       0                    0
```


show interfaces policers

Syntax

show interfaces policers
<*interface-name*>

Release Information

Command introduced before Junos OS Release 7.4.
Command introduced on PTX Series Packet Transport Routers for Junos OS Release 12.1.

Description

Display all policers that are installed on each interface in a system.

Options

none—Display policer information about all interfaces.

interface-name—(Optional) Display filter information about a particular interface.

Additional Information

For information about how to configure policers, see the *Junos Policy Framework Configuration Guide*. For related operational mode commands, see the *Junos Routing Protocols and Policies Command Reference*.

Required Privilege Level

view

List of Sample Output

show interfaces policers on page 1606
show interfaces policers interface-name on page 1606
show interfaces policers (PTX Series Packet Transport Routers) on page 1606

Output Fields

Table 107 on page 1605 lists the output fields for the **show interfaces policers** command. Output fields are listed in the approximate order in which they appear.

Table 107: show interfaces policers Output Fields

Field Name	Field Description
Interface	Name of the interface.
Admin	Interface state: **up** or **down**.
Link	Link state: **up** or **down**.
Proto	Protocol configured on the interface.
Input Policer	Policer to be evaluated when packets are received on the interface. It has the format ***interface-name*-in-policer**.
Output Policer	Policer to be evaluated when packets are transmitted on the interface. It has the format ***interface-name*-out-policer.**

Sample Output

show interfaces policers

```
user@host> show interfaces policers
Interface       Admin Link Proto Input Policer         Output Policer
ge-0/0/0        up    up
ge-0/0/0.0      up    up   inet
                           iso
gr-0/3/0        up    up
ip-0/3/0        up    up
mt-0/3/0        up    up
pd-0/3/0        up    up
pe-0/3/0        up    up
...
so-2/0/0        up    up
so-2/0/0.0      up    up   inet  so-2/0/0.0-in-policer so-2/0/0.0-out-policer
                           iso
so-2/1/0        up    down
...
```

show interfaces policers interface-name

```
user@host> show interfaces policers so-2/1/0
Interface       Admin Link Proto Input Policer         Output Policer
so-2/1/0        up    down
so-2/1/0.0      up    down inet  so-2/1/0.0-in-policer so-2/1/0.0-out-policer
                           iso
                           inet6
```

show interfaces policers (PTX Series Packet Transport Routers)

```
user@host> show interfaces policers em0
Interface       Admin Link Proto Input Policer         Output Policer
    em0             up    up
    em0.0           up    up
                   inet
```


show policer

Syntax

show policer
<detail>
<*policer-name*>

Release Information

Command introduced before Junos OS Release 7.4.
Option **detail** introduced in Junos OS Release 12.3.

Description

Display the number of policed packets for a given policer or an aggregate policer. An aggregate policer is an aggregate of different policers on the same logical interface.

Options

none—Display the number of policed packets for all configured policers.

detail—(Optional) Display enhanced statistics and counters for policers.

policer-name—(Optional) Display the number of policed packets for the specified policer.

Required Privilege Level

view

List of Sample Output

show policer (MX Series) on page 1608
show policer (non MX Series Router) on page 1608
show policer (Aggregate Policer, non MX Series Router) on page 1608
show policer detail on page 1609

Output Fields

Table 108 on page 1607 lists the output fields for the **show policer** command. Output fields are listed in the approximate order in which they appear.

Table 108: show policer Output Fields

Field Name	Field Description
Name	Name of the policer.
Bytes	• (For two-color policers on MX Series routers, and for hierarchical policers on MS-DPC, MIC, and MPC interfaces on MX Series routers)—Total number of bytes policed by the specified policer. For other combinations of policer type, device, and line card type, this field is blank. • (T Series and M10i)—Not applicable. The Bytes information is not displayed.
Packets	Total number of packets policed by the specified policer.

Table 108: show policer Output Fields *(continued)*

Field Name	Field Description
Policer detail	• OOS packet statistics for packets that are marked out-of-specification by the policer. Changes to all packets that have out-of-specification actions, such as discard, color marking, or forwarding-class, are included in this counter. • Offered packet statistics for traffic subjected to policing. • Transmitted packet statistics for traffic that is not discarded by the policer. When the policer action is discard, the statistics are the same as the within-specification statistics; when the policer action is non-discard (loss-priority or forwarding-class), the statistics are included in this counter.

Sample Output

show policer (MX Series)

```
user@host> show policer
Policers:
Name                                            Bytes              Packets
__default_arp_policer__                        314520                 5242
pol-2M-ge-1/2/0.1-inet-i                     10372300               103723
pol-2M-ge-1/2/0.1-inet6-i                     7727800                77278
pol-2M-ge-1/2/0.1-mpls-i                      7070336                67984
pol-2M-ge-1/2/0.1001-vpls-i                  65153700               651537
pol-2M-ge-1/2/0.2001-vpls-i                  65180900               651809
pol-2M-ge-1/2/0.3001-ccc-i                   62202144               647939
```

show policer (non MX Series Router)

```
user@host> show policer
Policers:

Name                                            Bytes              Packets
__default_arp_policer__                            NA                 5242
pol-2M-ge-1/2/0.1-inet-i                           NA               103723
pol-2M-ge-1/2/0.1-inet6-i                          NA                77278
pol-2M-ge-1/2/0.1-mpls-i                           NA                67984
pol-2M-ge-1/2/0.1001-vpls-i                        NA               651537
pol-2M-ge-1/2/0.2001-vpls-i                        NA               651809
pol-2M-ge-1/2/0.3001-ccc-i                         NA               647939
```

show policer (Aggregate Policer, non MX Series Router)

```
user@host> show policer
Policers:
Name                                            Bytes              Packets
__default_arp_policer__                            NA                    0
P1-ae0.0-log_int-o                                 NA                    0
P2-ge-7/0/2.0-inet-o                               NA                    0
P2-ge-7/0/2.0-inet6-o                              NA                    0
__policer_tmpl__-term                              NA                    0
__policer_tmpl__-fc0                               NA                    0
__policer_tmpl__-fc0                               NA                    0
__policer_tmpl__-fc1                               NA                    0
__policer_tmpl__-fc0                               NA                    0
```



```
__policer_tmpl__-fc1                                    NA              0
__policer_tmpl__-fc2                                    NA              0
__policer_tmpl__-fc0                                    NA              0
__policer_tmpl__-fc1                                    NA              0
__policer_tmpl__-fc2                                    NA              0
__policer_tmpl__-fc3                                    NA              0
```

show policer detail

```
user@host> show policer detail
Policers:
Name                                                 Bytes        Packets
__default_arp_policer__
    OOS                                                  0              0
    Offered                                              0            496
    Transmitted                                          0            496
P1-xe-1/0/0.0-inet-i
    OOS                                                  0          11329
    Offered                                              0         111188
    Transmitted                                          0          99859
```

