Classifying Packets Based on Various Packet Header Fields on EX9200 Switches

Published: 2013-04-19

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Table of Contents

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:

- EX 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 the CLI User Guide.

Documentation Conventions

Table 1 on page ix 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.

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

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the**configure** command: user@host> **configure**
`Fixed-width text like this`	Represents output that appears on the terminal screen.	`user@host>` **show chassis alarms** `No alarms currently active`

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

Convention	Description	Examples
Italic text like this	• Introduces or emphasizes important new terms. • Identifies book names. • Identifies RFC and Internet draft titles.	• A policy *term* is a named structure that defines match conditions and actions. • *Junos OS System Basics Configuration Guide* • RFC 1997, *BGP Communities Attribute*
Italic text like this	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# **set system domain-name** *domain-name*
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	• 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**.
< > (angle brackets)	Enclose optional keywords or variables.	**stub <default-metric** ***metric*****>;**
| (pipe symbol)	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.	**broadcast | multicast** **(*string1* | *string2* | *string3*)**
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	**rsvp { # Required for dynamic MPLS only**
[ ] (square brackets)	Enclose a variable for which you can substitute one or more values.	**community name members [** ***community-ids*** **]**

Indention and braces ({ })	Identify a level in the configuration hierarchy.	[edit]
		routing-options {
		  static {
		  route default {
		  nexthop *address*;
		  retain;
		  }
		  }
		}

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
GUI Conventions		
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 send your comments to techpubs-comments@juniper.net, or fill out the documentation feedback form at https://www.juniper.net/cgi-bin/docbugreport/ . If you are using e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release 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 JNASC 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: https://www.juniper.net/alerts/

- 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

Overview

Multifield Classifier Overview

A multifield classifier is a method of classifying traffic flows. Devices that sit at the edge of a network usually classify packets according to codings that are located in multiple packet header fields. Multifield classification is normally performed at the network edge because of the general lack of DiffServ code point (DSCP) or IP precedence support in end-user applications.

In an edge router, a multifield classifier provides the filtering functionality that scans through a variety of packet fields to determine the forwarding class for a packet. Typically, a classifier performs matching operations on the selected fields against a configured value.

Unlike a behavior aggregate (BA), which classifies packets based on class-of-service (CoS) bits in the packet header, a multifield classifier can examine multiple fields in the packet header—for example, the source and destination address of the packet, and the source and destination port numbers of the packet. A multifield classifier typically matches one or more of the six packet header fields: destination address, source address, IP protocol, source port, destination port, and DSCP. Multifield classifiers are used when a simple BA classifier is insufficient to classify a packet.

In the Juniper Networks® Junos® operating system (Junos OS), you configure a multifield classifier with a firewall filter and its associated match conditions. This enables you to use any filter match criteria to locate packets that require classification. From a CoS perspective, multifield classifiers (or firewall filter rules) provide the following services:

- Classify packets to a forwarding class and loss priority. The forwarding class determines the output queue. The loss priority is used by schedulers in conjunction with the random early discard (RED) algorithm to control packet discard during periods of congestion.

- Police traffic to a specific bandwidth and burst size. Packets exceeding the policer limits can be discarded, or can be assigned to a different forwarding class, to a different loss priority, or to both.

NOTE: You *police* traffic on input to conform to established CoS parameters, setting loss handling and forwarding class assignments as needed. You *shape* traffic on output to make sure that router resources, especially bandwidth, are distributed fairly. However, input policing and output shaping are two different CoS processes, each with their own configuration statements.

Overview of Simple Filters

Simple filters are recommended for metropolitan Ethernet applications. They are supported on Gigabit Ethernet intelligent queuing 2 (IQ2) and Enhanced Queuing Dense Port Concentrator (DPC) interfaces only.

Unlike normal filters, simple filters are for IPv4 traffic only and have the following restrictions:

- The **next term** action is not supported.
- Qualifiers, such as the **except** and **protocol-except** statements, are not supported.
- Noncontiguous masks are not supported.
- Multiple source addresses and destination addresses in a single term are not supported. If you configure multiple addresses, only the last one is used.
- Ranges are only valid as source or destination ports. For example, **source-port 400-500** or **destination-port 600-700**.
- Output filters are not supported. You can apply a simple filter to ingress traffic only.
- Simple filters are not supported for interfaces in an aggregated-Ethernet bundle.
- Explicitly configurable terminating actions, such as **accept**, **reject**, and **discard**, are not supported. Simple filters always accept packets.

NOTE: On the MX Series routers with the Enhanced Queuing DPC, the forwarding class is not supported as a **from** match condition.

Two-Color Policers and Shaping Rate Changes

When you configure a change in shaping rate, it is important to consider the effect on the bandwidth limit. Whenever the shaping rate changes, the bandwidth limit is adjusted based on whether a logical interface (unit) or bandwidth percentage policer is configured.

When a logical interface bandwidth policer is configured, the order of priority for the shaping rate (if configured at that level) is:

- The shaping rate applied to the logical interface (unit).
- The shaping rate applied to the physical interface (port).
- The physical interface speed.

When a bandwidth percentage policer is configured, the order of priority for the shaping rate (if configured at that level) is:

- The shaping rate applied to the physical interface (port).
- The physical interface speed.

These guidelines must be kept in mind when calculating the logical link speed and link speed from the configured shaping rate, which determines the rate-limited bandwidth after the policer is applied.

Related Documentation

- Example: Two-Color Policers and Shaping Rate Changes on page 21

Understanding IEEE 802.1p Inheritance push and swap from a Transparent Tag

MX Series routers with Modular Port Concentrators (MPCs) and Modular Interface Cards (MICs) and M320 routers and EX Series switches support configurable IEEE 802.1p inheritance of push and swap bits from the transparent tag of each incoming packet which allows you to classify incoming packets based on the IEEE 802.1p bits from the transparent tag.

During a tagging operation, Junos OS by default inherits the IEEE 802.1p bits from incoming tags in swap and push operations from the known tags configured on the interface.

It can be useful to override the default behavior by configuring Junos OS to inherit the IEEE 802.1p bits from a transparent tag, and to classify incoming packets based on the IEEE 802.1p bits of the incoming transparent tag. The configuration statements **swap-by-poppush** and **transparent** enable Junos OS to do this.

By default, during a swap operation, the IEEE 802.1p bits of the VLAN tag remain unchanged. When the **swap-by-poppush** operation is enabled on a logical interface, the swap operation is treated as a **pop** operation followed by **push** operation. The **pop** operation removes the existing tag and the associated IEEE 802.1p bits and the push operation copies the inner VLAN IEEE 802.1p bits to the IEEE bits of the VLAN or VLANs being pushed. As a result, the IEEE 802.1p bits are inherited from the incoming transparent tag.

To classify incoming packets based on the IEEE 802.1p bits from the transparent tag, include the **transparent** statement at the **[edit class-of-service interfaces** *interface-name* **unit** *logical-unit-number* **classifiers ieee-802.1 vlan-tag]** hierarchy level.

To configure Junos OS to inherit the IEEE 802.1p bits from the transparent tag, include the **swap-by-poppush** statement at the **[edit interfaces *interface-name* unit *logical-unit-number*]** hierarchy level.

NOTE: IEEE 802.1p Inheritance push and swap is only supported on untagged and single-tagged logical interfaces, and is not supported on dual-tagged logical interfaces.

Related Documentation

- swap-by-poppush
- transparent on page 27
- Understanding swap-by-poppush
- Configuring IEEE 802.1p Inheritance push and swap from the Transparent Tag on page 10
- Understanding Transparent Tag Operations and IEEE 802.1p Inheritance

PART 2

Configuration

CHAPTER 2

Configuration Tasks

Configuring Multifield Classifiers

If you configure both a behavior aggregate (BA) classifier and a multifield classifier, BA classification is performed first; then multifield classification is performed. If they conflict, any BA classification result is overridden by the multifield classifier.

NOTE: For a specified interface, you can configure both a multifield classifier and a BA classifier without conflicts. Because the classifiers are always applied in sequential order, the BA classifier followed by the multifield classifier, any BA classification result is overridden by a multifield classifier if they conflict.

To activate a multifield classifier, you must configure it on a logical interface. There is no restriction on the number of multifield classifiers you can configure.

NOTE: For MX Series routers and EX Series switches, if you configure a firewall filter with a DSCP action or traffic-class action on a DPC, the commit does not fail, but a warning displays and an entry is made in the syslog.

For an L2TP LNS on MX Series routers, you can attach firewall for static LNS sessions by configuring these at logical interfaces directly on the inline services device (si-fpc/pic/port). RADIUS-configured firewall attachments are not supported.

To configure multifield classifiers, include the following statements at the **[edit firewall]** hierarchy level:

```
[edit firewall]
family family-name {
```

```
filter filter-name {
  term term-name {
    from {
      match-conditions;
    }
    then {
      dscp 0;
      forwarding-class class-name;
      loss-priority (high | low);
    }
  }
}
simple-filter filter-name {
  term term-name {
    from {
      match-conditions;
    }
    then {
      forwarding-class class-name;
      loss-priority (high | low | medium);
    }
  }
}
}
```

Configuring Logical Bandwidth Policers

When you configure a policer as a percentage (using the **bandwidth-percent** statement), the bandwidth is calculated as a percentage of either the physical interface media rate or the logical interface shaping rate. To specify that the bandwidth be calculated based on the logical interface shaping rate and not the physical interface media rate, include the **logical-bandwidth-policer** statement. If a shaping rate is not configured for the logical interface, the physical interface media rate is used, even if you include the **logical-bandwidth-policer** You can configure the shaping rate on the logical interface using class-of-service statements.

```
[edit firewall policer policer-name]
logical-bandwidth-policer;
```

Configuring IEEE 802.1p Inheritance push and swap from the Transparent Tag

To classify incoming packets based on the IEEE 802.1p bits from the transparent tag, include the **transparent** statement at the **[edit class-of-service interfaces** *interface-name* **unit** *logical-unit-number* **classifiers ieee-802.1 vlan-tag]** hierarchy level.

Tagged Interface Example

The following example configuration specifies the classification based on the transparent VLAN tag.

```
edit
class-of-service {
  interfaces {
    ge-3/0/1 {
      unit 0 {
        classifiers {
```

```
            ieee-802.1 default vlan-tag transparent;
          }
        }
      }
    }
  }
```

To configure Junos OS to inherit the IEEE 802.1p bits from the transparent tag, include the **swap-by-poppush** statement at the **[edit interfaces** ***interface-name*** **unit** ***logical-unit-number*****]** hierarchy level.

The following is a configuration to swap and push VLAN tags and allow inheritance of the IEEE 802.1p value from the transparent VLAN tag in incoming packets.

```
edit
  ge-3/0/0 {
  vlan-tagging;
  encapsulation vlan-ccc;
  unit 0 {
    encapsulation vlan-ccc;
    vlan-id 100;
    swap-by-poppush;
    input-vlan-map {
      swap-push;
      tag-protocol-id 0x9100;
      inner-tag-protocol-id 0x9100;
      vlan-id 500;
      inner-vlan-id 400;
    }
    output-vlan-map {
      pop-swap;
      inner-vlan-id 100;
      inner-tag-protocol-id 0x88a8;
    }
  }
}
```

The **swap-by-poppush** statement causes a swap operation to be done as a pop followed by a push operation. So for the outer tag, the incoming S-Tag is popped and a new tag is pushed. As a result, the S-Tag inherits the IEEE 802.1p bits from the transparent tag. The inner tag is then pushed, which results in the inner tag inheriting the IEEE 802.1p bits from the transparent tag.

Untagged Interface Example

The following is a configuration to push two VLAN tags and allow inheritance of the IEEE 802.1p value from the transparent VLAN tag in the incoming packet.

```
[edit]
ge-3/0/1 {
  encapsulation ccc;
  unit 0 {
    input-vlan-map {
      push-push;
      tag-protocol-id 0x9100;
      inner-tag-protocol-id 0x9100;
      vlan-id 500;
      inner-vlan-id 400;
```

```
        }
        output-vlan-map{
          pop-pop;
        }
      }
    }
```

No additional configuration is required to inherit the IEEE 802.1p value, as the **push** operation inherits the IEEE 802.1p values by default.

The following configuration specifies the classification based on the transparent VLAN tag.

```
[edit]
class-of-service {
  interfaces {
    ge-3/0/1 {
      unit 0 {
        classifiers {
          ieee-802.1 default vlan-tag transparent;
        }
      }
    }
  }
}
```

Related Documentation

- transparent on page 27
- swap-by-poppush
- Understanding IEEE 802.1p Inheritance push and swap from a Transparent Tag on page 5
- Understanding swap-by-poppush
- Understanding Transparent Tag Operations and IEEE 802.1p Inheritance

CHAPTER 3

Examples

Example: Classifying Packets Based on Their Destination Address

Configure a multifield classifier that ensures that all IPv4 packets destined for the **10.10.10.0/24** network are placed into the **platinum** forwarding class. This assignment occurs regardless of the received CoS bit values in the packet. Apply this filter to the inbound interface **ge-1/2/2.0**.

To verify that your configuration is attached to the correct interface, issue the **show interfaces filters** command.

```
[edit]
firewall {
    family inet {
        filter set-FC-to-platinum {
            term match-a-single-route {
                from {
                    destination-address {
                        10.10.10.0/24;
                    }
                }
                then {
                    forwarding-class platinum;
                    accept;
                }
            }
            term accept-all {
                then accept;
            }
        }
    }
```

```
}
interfaces {
  ge-1/2/2 {
    unit 0 {
      family inet {
        filter {
          input set-FC-to-platinum;
        }
      }
    }
  }
}
```

Example: Configuring and Verifying a Complex Multifield Filter

In this example, SIP signaling (VoIP) messages use TCP/UDP, port 5060, and RTP media channels use UDP with port assignments from 16,384 through 32,767. See the following sections:

- Configuring a Complex Multifield Filter on page 14
- Verifying a Complex Multifield Filter on page 16

Configuring a Complex Multifield Filter

To configure the multifield filter, perform the following actions:

- Classify SIP signaling messages (VoIP network control traffic) as NC with a firewall filter.
- Classify VoIP traffic as EF with the same firewall filter.
- Police all remaining traffic with IP precedence **0** and make it BE.
- Police BE traffic to 1 Mbps with excess data marked with PLP high.
- Apply the firewall filter with policer to the interface.

The firewall filter called **classify** matches on the transport protocol and ports identified in the incoming packets and classifies packets into the forwarding classes specified by your criteria.

The first term, **sip**, classifies SIP signaling messages as network control messages. The **port** statement matches any source port or destination port (or both) that is coded to 5060.

Classifying SIP Signaling Messages

```
firewall {
  family inet {
    filter classify {
      interface-specific;
      term sip {
        from {
          protocol [ udp tcp ];
          port 5060;
```

```
            }
            then {
              forwarding-class network-control;
              accept;
            }
          }
        }
      }
    }
```

The second term, **rtp**, classifies VoIP media channels that use UDP-based transport.

Classifying VoIP Channels That Use UDP

```
term rtp {
  from {
    protocol udp;
    port 16384-32767;
  }
  then {
    forwarding-class expedited-forwarding;
    accept;
  }
}
```

The policer's burst tolerance is set to the recommended value for a low-speed interface, which is ten times the interface MTU. For a high-speed interface, the recommended burst size is the transmit rate of the interface times 3 to 5 milliseconds.

Configuring the Policer

```
policer be-policer {
  if-exceeding {
    bandwidth-limit 1m;
    burst-size-limit 15k;
  }
  then loss-priority high;
}
```

The third term, **be**, ensures that all remaining traffic is policed according to a bandwidth restriction.

Policing All Remaining Traffic

```
term be {
  then policer be-policer;
}
```

The **be** term does not include a **forwarding-class** action modifier. Furthermore, there is no explicit treatment of network control (NC) traffic provided in the **classify** filter. You can configure explicit classification of NC traffic and all remaining IP traffic, but you do not need to, because the default IP precedence classifier correctly classifies the remaining traffic.

Apply the **classify** classifier to the **fe-0/0/2** interface:

Applying the Classifier

```
interfaces {
  fe-0/0/2 {
    unit 0 {
      family inet {
        filter {
          input classify;
        }
        address 10.12.0.13/30;
      }
    }
  }
}
```

Verifying a Complex Multifield Filter

Before the configuration is committed, display the default classifiers in effect on the interface using the **show class-of-service interface *interface-name*** command. The display confirms that the **ipprec-compatibility** classifier is in effect by default.

Verifying Default Classification

```
user@host> show class-of-service fe-0/0/2
Physical interface: fe-0/0/2, Index: 135
Queues supported: 8, Queues in use: 4
  Scheduler map: <default>, Index: 2032638653

  Logical interface: fe-0/0/2.0, Index: 68
    Shaping rate: 32000
    Object         Name                  Type                      Index
    Scheduler-map  <default>                                          27
    Rewrite        exp-default           exp                         21
    Classifier     exp-default           exp                          5
    Classifier     ipprec-compatibility  ip                           8
```

To view the default classifier mappings, use the **show class-of-service classifier name *name*** command. The highlighted output confirms that traffic with IP precedence setting of 0 is correctly classified as BE, and NC traffic, with precedence values of 6 or 7, is properly classified as NC.

Displaying Default Classifier Mappings

```
user@host> show class-of-service classifier name ipprec-compatibility
Classifier: ipprec-compatibility, Code point type: inet-precedence, Index: 12
  Code point          Forwarding class                    Loss priority
  000                 best-effort                         low
  001                 best-effort                         high
  010                 best-effort                         low
  011                 best-effort                         high
  100                 best-effort                         low
  101                 best-effort                         high
  110                 network-control                     low
  111                 network-control                     high
```

After your configuration is committed, verify that your multifield classifier is working correctly. You can monitor the queue counters for the routing device's ***egress*** interface used when forwarding traffic received from the peer. Displaying the queue counters for the ingress interface (**fe-0/0/2**) does not allow you to check your ingress classification,

because queuing generally occurs only at egress in the Junos OS. (Ingress queuing is supported on Gigabit Ethernet IQ2 PICs and Enhanced IQ2 PICs only.)

To verify the operation of the multifield filter:

1. To determine which egress interface is used for the traffic, use the **traceroute** command.
2. After you identify the egress interface, clear its associated queue counters by issuing the **clear interfaces statistics *interface-name*** command.
3. Confirm the default forwarding class-to-queue number assignment. This allows you to predict which queues are used by the VoIP, NC, and other traffic. To do this, issue the **show class-of-service forwarding-class** command.
4. Display the queue counts on the interface by issuing the **show interfaces queue** command.

Example: Writing Different DSCP and EXP Values in MPLS-Tagged IP Packets

On Juniper Networks M320 Multiservice Edge Routers and T Series Core Routers and on EX Series switches, you can selectively set the DSCP field of MPLS-tagged IPv4 and IPv6 packets to **000000**. In the same packets, you can set the MPLS EXP field according to a configured rewrite table, which is based on the forwarding classes that you set in incoming packets using a BA or multifield classifier.

Queue selection is based on the forwarding classes you assign in scheduler maps. This means that you can direct traffic to a single output queue, regardless of whether the DSCP field is unchanged or rewritten to **000000**. To do this, you must configure a multifield classifier that matches selected packets and modifies them with the **dscp 0** action.

Selective marking of DSCP fields to **0**, without affecting output queue assignment, can be useful. For example, suppose you need to use the MPLS EXP value to configure CoS applications for core provider routing devices. At the penultimate egress provider edge (PE) router where the MPLS labels are removed, the CoS bits need to be provided by another value, such as DSCP code points. This case illustrates why it is useful to mark both the DSCP and MPLS EXP fields in the packet. Furthermore, it is useful to be able to mark the two fields differently, because the CoS rules of the core provider routing device might differ from the CoS rules of the egress penultimate router. At egress, as always, you can use a rewrite table to rewrite the MPLS EXP values corresponding to the forwarding classes that you need to set.

NOTE: When both customer-facing and core-facing interfaces exist, you can derive the EXP value in the following precedence order, while adding the MPLS label:

1. EXP value provided by the CoS rewrite action.
2. EXP value derived from the top label of the stack (MPLS label stacking).
3. IPv4 or IPv6 precedence (Layer 3 VPN, Layer 2 VPN, and VPLS scenarios).

For IPv4 traffic, the **dscp 0** action modifier at the **[edit firewall family inet filter *filter-name* term *term-name* then]** hierarchy level is valid. However, for IPv6 traffic, you configure this feature by including the **traffic-class 0** action modifier at the **[edit firewall family inet6 filter *filter-name* term *term-name* then]** hierarchy level.

In the following IPv4 example, term **1** of the multifield classifier matches packets with DSCP **001100** code points coming from a certain VRF, rewrites the bits to DSCP **000000**, and sets the forwarding class to **best-effort**. In term **2**, the classifier matches packets with DSCP **010110** code points and sets the forwarding class to **best-effort**. Because term **2** does not include the **dscp 0** action modifier, the DSCP **010110** bits remain unchanged. Because the classifier sets the forwarding class for both code points to **best-effort**, both traffic types are directed to the same output queue.

NOTE: If you configure a bit string in a DSCP match condition in a firewall filter, then you must include the letter "b" in front of the string, or the match rule creation fails on commit.

```
[edit]
firewall {
  family inet {
    filter vrf-rewrite {
      term 1 {
        from {
          dscp b001100;
        }
        then {
          dscp 0;
          forwarding-class best-effort;
        }
      }
      term 2 {
        from {
          dscp b010110;
        }
        then {
          forwarding-class best-effort;
        }
      }
    }
  }
}
```

Applying the Multifield Classifier

Apply the filter to an input interface corresponding to the VRF:

```
[edit]
interfaces {
  ge-0/1/0 {
    unit 0 {
      family inet {
        filter input vrf-rewrite;
      }
    }
  }
```

```
}
```

NOTE: The **dscp 0** action is supported in both input and output filters. You can use this action for non-MPLS packets as well as for IPv4 and IPv6 packets entering an MPLS network. All IPv4 and IPv6 firewall filter match conditions are supported with the **dscp 0** action.

The following limitations apply:

- You can use a multifield classifier to rewrite DSCP fields to value 0 only. Other values are not supported.
- If a packet matches a filter that has the **dscp 0** action, then the outgoing DSCP value of the packet is 0, even if the packet matches a rewrite rule, and the rewrite rule is configured to mark the packet to a non-zero value. The **dscp 0** action overrides any other rewrite rule actions configured on the routing device.
- Although you can use the **dscp 0** action on an input filter, the output filter and other classifiers do not see the packet as being marked **dscp 0**. Instead, they classify the packet based on its original incoming DSCP value. The DSCP value of the packet is set to 0 after all other classification actions have completed on the packet.

Example: Configuring a Simple Filter

This simple filter sets the loss priority to low for TCP traffic with source address **1.1.1.1**, sets the loss priority to high for HTTP (web) traffic with source addresses in the **4.0.0.0/8** range, and sets the loss priority to low for all traffic with destination address **6.6.6.6**. 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**.

```
[edit]
firewall {
   family inet {
      simple-filter filter1 {
         term 1 {
            from {
               source-address {
                  1.1.1.1/32;
               }
               protocol {
                  tcp;
               }
            }
            then loss-priority low;
         }
         term 2 {
            from {
               source-address {
                  4.0.0.0/8;
```

```
            }
            source-port {
              http;
            }
          }
          then loss-priority high;
        }
        term 3 {
          from {
            destination-address {
              6.6.6.6/32;
            }
          }
          then {
            loss-priority low;
            forwarding-class best-effort;
          }
        }
      }
    }
  }
  interfaces {
    ge-0/0/1 {
      unit 0 {
        family inet {
          simple-filter {
            input filter1;
          }
          address 10.1.2.3/30;
        }
      }
    }
  }
```

Example: Configuring a Logical Bandwidth Policer

This example applies a logical bandwidth policer rate to two logical interfaces on interface **ge-0/2/7**. The policed rate on **unit 0** is 2 Mbps (50 percent of 4 Mbps) and the policed rate on **unit 1** is 1 Mbps (50 percent of 2 Mbps).

```
[edit firewall]
policer Logical_Policer {
  logical-bandwidth-policer; # This applies the policer to logical interfaces
  if-exceeding {
    bandwidth-percent 50; # This applies 50 percent to the shaping-rate
    burst-size-limit 125k;
  }
  then discard;
}

[edit class-of-service]
interfaces {
  ge-0/2/7 {
    unit 0 {
      shaping-rate 4m # This establishes the rate to be policed on unit 0
```

```
    }
    unit 1 {
      shaping-rate 2m # This establishes the rate to be policed on unit 1
    }
  }
}
[edit interfaces ge-0/2/7]
per-unit-scheduler;
vlan-tagging;
unit 0 {
  vlan-id 100;
  family inet {
    policer {
      input Logical_Policer;
      output Logical_Policer;
    }
    address 172.1.1.1/30;
  }
}
unit 1 {
  vlan-id 200;
  family inet {
    policer {
      input Logical_Policer;
      output Logical_Policer;
    }
    address 172.2.1.1/30;
  }
}
```

Example: Two-Color Policers and Shaping Rate Changes

In this example, the shaping rate has been configured for the logical interface, but a bandwidth percentage policer is also configured. Therefore policing is based on the physical interface speed of 1 Gbps.

If both a shaping rate and a bandwidth percentage policer is configured on the same logical interface, the policing is based on the physical interface speed. Here is the example configuration:

```
[edit interfaces]
ge-0/1/0 {
  per-unit-scheduler;
  vlan-tagging;
  unit 0 {
    vlan-id 1;
    family inet {
      policer {
        output policer_test;
      }
      address 10.0.7.1/24;
    }
  }
}
```

```
[edit firewall]
policer policer_test {
   if-exceeding {
      bandwidth-percent 75;
      burst-size-limit 256k;
   }
   then discard;
}

[edit]
class-of-service {
   interfaces {
      ge-0/1/0 {
         unit 0 {
            shaping-rate 15m;
         }
      }
   }
}
```

CHAPTER 4

Configuration Statements

[edit class-of-service] Hierarchy Level

```
class-of-service {
  classifiers {
    type classifier-name {
      forwarding-class class-name {
        loss-priority (high | low | medium-high | medium-low) code-points [ aliases bits ];
      }
      import (classifier-name | default);
    }
  }
  code-point-aliases {
    (dscp | dscp-ipv6 | exp | ieee-802.1 | ieee-802.1ad | inet-precedence) {
      alias-name bits;
    }
  }
  drop-profiles {
    profile-name {
      fill-level percentage drop-probability percentage;
      interpolate {
        drop-probability value;
        fill-level value;
      }
    }
  }
  fabric {
    scheduler-map {
      priority (high | low) scheduler scheduler-name;
    }
  }
  forwarding-class-map {
    map-name {
      class class-name queue-num queue-number <restricted-queue queue-number>;
    }
  }
  forwarding-classes {
```

```
    class class-name policing-priority (normal | premium) queue-num queue-number
      priority (high | low);
    queue queue-number class-name policing-priority (normal | premium) priority (high |
      low);
  }
  forwarding-policy {
    class class-name {
      classification-override {
        forwarding-class class-name;
      }
    }
    next-hop-map map-name {
      forwarding-class class-name {
        discard;
        lsp-next-hop [ lsp-regular-expressions ];
        next-hop [ next-hop-names ];
        non-lsp-next-hop;
      }
    }
  }
  fragmentation-maps {
    map-name {
      forwarding-class class-name {
        drop-timeout milliseconds;
        fragment-threshold bytes;
        multilink-class number;
        no-fragmentation;
      }
    }
  }
  host-outbound-traffic {
    dscp-code-point value;
    forwarding-class class-name;
    ieee-802.1 {
      default value;
      rewrite-rules;
    }
    tcp {
      raise-internet-control-priority;
    }
  }
  interfaces {
    ... the interfaces subhierarchy appears after the main [edit class-of-service] hierarchy
      ...
  }
  }
  restricted-queues {
    forwarding-class class-name queue-number;
  }
  rewrite-rules {
    (dscp | dscp-ipv6 | exp |frame-relay-de | ieee-802.1 | ieee-802.1ad | inet-precedence)
      rewrite-rule {
      forwarding-class class-name {
        loss-priority level code-point (alias | bits);
      }
      import (rewrite-rule | default);
```

```
    }
  }
  routing-instances routing-instance-name {
    classifiers {
      dscp (classifier-name | default);
      dscp-ipv6 (classifier-name | default);
      exp (classifier-name | default);
      ieee-208.1 (classifier-name | default | encapsulated | vlan-tag (inner | outer));
    }
  }
  scheduler-maps {
    map-name {
      forwarding-class class-name scheduler scheduler-name;
    }
  }
  schedulers {
    scheduler-name {
      adjust-minimum value;
      adjust-percent value;
      buffer-size (exact | percent percentage | remainder);
      drop-profile-map loss-priority (any | high | low | medium-high | medium-low)
        protocol any;
      excess-priority (high | low | medium-high | medium-low);
      excess-rate (percent percentage | proportion proportion);
      priority (high | low | medium-high | medium-low | strict-high);
      shaping-rate (bps | percent percentage | burst-size size);
      transmit-rate (bps | percent percentage | remainder) <exact | rate-limit>;
    }
  }
  traceoptions {
    file <files number> <match regular-expression> <size maximum-file-size>
      <world-readable | no-world-readable>;
    flag flag;
    no-remote-trace;
  }
  traffic-control-profiles {
    profile-name {
      adjust-minimum rate;
      delay-buffer-rate (bps | cps cps | percent percentage);
      excess-rate (percent percentage | proportion value);
      guaranteed-rate (bps | percent percentage) <burst-size bytes>;
      overhead-accounting (frame-mode | cell-mode) <bytes byte-value>;
      scheduler-map map-name;
      shaping-rate (bps | percent percentage) <burst-size bytes>;
    }
  }
  tri-color;
}

class-of-service {
  interfaces {
    interface-name {
      excess-bandwith-share (equal | proportional value);
      input-excess-bandwith-share (equal | proportional value);
      input-scheduler-map map-name;
      input-shaping-rate bps;
```

```
    input-traffic-control-profile profile-name;
    output-forwarding-class-map map-name;
    output-traffic-control-profile profile-name;
    scheduler-map map-name;
    scheduler-map-chassis (map-name | derived);
    shaping-rate bps;
    unit (logical-unit-number | *){
      classifiers {
        dscp (classifier-name | default) {
          family [ inet mpls ];
        }
        dscp-ipv6 (classifier-name | default) {
          family [ inet mpls ];
        }
        exp (classifier-name | default);
        ieee-208.1 (classifier-name | default)  <vlan-tag (inner | outer)>;
        ieee-208.1ad (classifier-name | default);
        inet-precedence (classifier-name | default);
      }
      forwarding-class class-name;
      input-scheduler-map map-name;
      input-shaping-rate bps;
      input-traffic-control-profile profile-name shared-instance instance-name;
      loss-priority-maps {
        (map-name | default);
      }
      loss-priority-rewrites {
        (map-name | default);
      }
      output-forwarding-class-map map-name;
      output-traffic-control-profile profile-name shared-instance instance-name;
      rewrite-rules {
        dscp (rule-name | default) <protocol mpls>;
        dscp-ipv6 (rule-name | default);
        exp (rule-name | default) <protocol [ mpls-any | mpls-inet-both |
          mpls-inet-both-non-vpn ]>;
        exp-push-push-push default;
        exp-swap-push-push default;
        ieee-802.1 (rewrite-name | default) <vlan-tag (outer | outer-and-inner)>;
        ieee-802.1ad (rewrite-name | default) <vlan-tag (outer | outer-and-inner)>;
        inet-precedence (rewrite-name | default) <protocol mpls>;
      }
      scheduler-map map-name;
      shaping-rate bps;
      translation-table (to-dscp-from-dscp | to-dscp-ipv6-from-dscp-ipv6 |
        to-exp-from-exp | to-inet-precedence-from-inet-precedence) table-name;
    }
  }
  interface-set interface-set-name {
    excess-bandwith-share (equal | proportional value);
    input-excess-bandwith-share (equal | proportional value);
    input-traffic-control-profile profile-name;
    input-traffic-control-profile-remaining profile-name;
    internal-node;
    output-traffic-control-profile profile-name;
    output-traffic-control-profile-remaining profile-name;
```

```
        }
    }
}
```

Related Documentation

- Notational Conventions Used in Junos OS Configuration Hierarchies

transparent

Syntax transparent;

Hierarchy Level [edit class-of-service interfaces *interface-name* unit *logical-unit-number* classifiers ieee802.1 vlan-tag]

Release Information Statement introduced in Junos OS Release 11.2

Description Packet classification based on the transparent VLAN tag.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

[edit firewall] Hierarchy Level

Several statements in the **[edit firewall]** hierarchy are valid at numerous locations within the hierarchy. To make the complete hierarchy easier to read, the repeated statements are listed in the following sections, which are referenced at the appropriate locations in "Complete [edit firewall] Hierarchy" on page 32.

Common Firewall Actions

This section lists statements that are valid at the following hierarchy levels, and is referenced at those levels in "Complete [edit firewall] Hierarchy" on page 32 instead of the statements being repeated.

- **[edit firewall family (any | ccc | ethernet-switching | inet | inet6 | mpls | vpls) filter *filter-name* term *term-name* then]**
- **[edit firewall filter *filter-name* term *term-name* then]**

The common firewall actions are as follows:

```
count counter-name;
forwarding-class class-name;
```



```
loss-priority (high | low | medium-high | medium-low);
next term;
policer policer-name;
three-color-policer policer-name {
   (single-rate single-rate-policer-name | two-rate two-rate-policer-name);
}
```

Common IP Firewall Actions

This section lists statements that are valid at the following hierarchy levels, and is referenced at those levels in "Complete [edit firewall] Hierarchy" on page 32 instead of the statements being repeated.

- **[edit firewall family inet filter *filter-name* term *term-name* then]**
- **[edit firewall family inet6 filter *filter-name* term *term-name* then]**
- **[edit firewall filter *filter-name* term *term-name* then]**

The common IP firewall actions are as follows:

```
log;
logical-system logical-system-name <routing-instance routing-instance-name>
   <topology topology-name>;
port-mirror;
port-mirror-instance instance-name;
routing-instance routing-instance-name> <topology topology-name>;
sample;
service-filter-hit;
syslog;
topology topology-name;
```

Common IPv4 Firewall Actions

This section lists statements that are valid at the following hierarchy levels, and is referenced at those levels in "Complete [edit firewall] Hierarchy" on page 32 instead of the statements being repeated.

- **[edit firewall family inet filter *filter-name* term *term-name* then]**
- **[edit firewall filter *filter-name* term *term-name* then]**

The common IP version 4 (IPv4) firewall actions are as follows:

```
(accept | discard <accounting collector-name> | reject <administratively-prohibited |
   bad-host-tos | bad-network-tos | fragmentation-needed | host-prohibited |
   host-unknown | host-unreachable | network-prohibited | network-unknown |
   network-unreachable | port-unreachable | precedence-cutoff | precedence-violation |
   protocol-unreachable | source-host-isolated | source-route-failed | tcp-reset>);
ipsec-sa sa-name;
load-balance sa-name;
next-hop-group group-name;
prefix-action action-name;
```


Common IP Firewall Match Conditions

This section lists statements that are valid at the following hierarchy levels, and is referenced at those levels in "Complete [edit firewall] Hierarchy" on page 32 instead of the statements being repeated.

- **[edit firewall family inet dialer-filter *filter-name* term *term-name* from]** (with the exceptions noted at this level in "Complete [edit firewall] Hierarchy" on page 32)
- **[edit firewall family inet filter *filter-name* term *term-name* from]**
- **[edit firewall family inet6 dialer-filter *filter-name* term *term-name* from]** (with the exceptions noted at this level in "Complete [edit firewall] Hierarchy" on page 32)
- **[edit firewall family inet6 filter *filter-name* term *term-name* from]**
- **[edit firewall filter *filter-name* term *term-name* from]**

The common IP firewall match conditions are as follows:

```
address {
    ip-prefix</prefix-length> <except>;
}
destination-address {
    ip-prefix</prefix-length> <except>;
}
destination-class [ class-names ] | destination-class-except [ class-names ]);
(destination-port [ port-names ] | destination-port-except [ port-names ]);
destination-prefix-list {
    list-name <except>;
}
(forwarding-class [ class-names ] | forwarding-class-except [ class-names ]);
(icmp-code [ codes ] | icmp-code-except [ codes ]);
(icmp-type [ types ] | icmp-type-except [ types ]);
interface interface-name;
(interface-group [ group-names ] | interface-group-except [ group-names ]);
interface-set set-name;
(loss-priority [ priorities ] | loss-priority-except [ priorities ]);
(packet-length [ values ] | packet-length-except [ values ]);
(port [ port-names ] | port-except [ port-names ]);
prefix-list {
    list-name <except>;
}
service-filter-hit;
source-address {
    ip-prefix</prefix-length> <except>;
}
(source-class [ class-names ] | source-class-except [ class-names ]);
(source-port [ port-names ] | source-port-except [ port-names ]);
source-prefix-list {
    list-name <except>;
}
tcp-established;
tcp-flags flag;
tcp-initial;
```

Common IPv4 Firewall Match Conditions

This section lists statements that are valid at the following hierarchy levels, and is referenced at those levels in "Complete [edit firewall] Hierarchy" on page 32 instead of the statements being repeated.

- **[edit firewall family inet dialer-filter *filter-name* term *term-name* from]** (with the exceptions noted at this level in "Complete [edit firewall] Hierarchy" on page 32)
- **[edit firewall family inet filter *filter-name* term *term-name* from]**
- **[edit firewall filter *filter-name* term *term-name* from]**

The common IPv4 firewall match conditions are as follows:

```
(ah-spi [ values ] | ah-spi-except [ values ]);
(dscp [ code-point-values ] | dscp-except [ code-point-values ]);
(esp-spi [ values ] | esp-spi-except [ values ]);
first-fragment;
fragment-flags flag;
(fragment-offset [ offsets ] | fragment-offset-except [ offsets ]);
(ip-options [ option-names ] | ip-options-except [ option-names ]);
is-fragment;
(precedence [ precedence-names ] | precedence-except [ precedence-names ]);
(protocol [ protocol-names ] | protocol-except [ protocol-names ]);
(ttl [ ttl-values ] | ttl-except [ ttl-values ]);
```

Common Layer 2 Firewall Match Conditions

This section lists statements that are valid at the following hierarchy levels, and is referenced at those levels in "Complete [edit firewall] Hierarchy" on page 32 instead of the statements being repeated.

- **[edit firewall family ethernet-switching filter *filter-name* term *term-name* from]**
- **[edit firewall family vpls filter *filter-name* term *term-name* from]**

The common Layer 2 firewall match conditions are as follows:

```
destination-mac-address {
    mac-address <except>;
}
(destination-port [ port-names ] | destination-port-except [ port-names ]);
(dscp [ code-point-values ] | dscp-except [ code-point-values ]);
(ether-type [ protocol-types ] | ether-type-except [ protocol-types ]);
(forwarding-class [ class-names ] | forwarding-class-except [ class-names ]);
(icmp-code [ codes ] | icmp-code-except [ codes ]);
(icmp-type [ types ] | icmp-type-except [ types ]);
(interface-group [ group-names ] | interface-group-except [ group-names ]);
ip-address {
    ip-prefix</prefix-length> <except>;
}
ip-destination-address {
    ip-prefix</prefix-length> <except>;
}
```



```
(ip-precedence [ precedence-names ] | ip-precedence-except [ precedence-names ]);
(ip-protocol [ protocol-names ] | ip-protocol-except [ protocol-names ]);
ip-source-address ip-prefix</prefix-length>;
(learn-vlan-1p-priority [ priorities ] | learn-vlan-1p-priority [ priorities ]);
(learn-vlan-id [ vlan-ids ] | learn-vlan-id-except [ vlan-ids ]);
(loss-priority [ priorities ] | loss-priority-except [ priorities ]);
(port [ port-names ] | port-except [ port-names ]);
source-mac-address {
  mac-address <except>;
}
(source-port [ port-names ] | source-port-except [ port-names ]);
tcp-flags flag;
(traffic-type [ broadcast known-unicast multicast unknown-unicast ] |
  traffic-type-except [ broadcast known-unicast multicast unknown-unicast ]);
(user-vlan-1p-priority [ priorities ] | user-vlan-1p-priority [ priorities ]);
(user-vlan-id [ vlan-ids ] | user-vlan-id-except [ vlan-ids ]);
(vlan-ether-type [ protocol-types ] | vlan-ether-type-except [ protocol-types ]);
```

Complete [edit firewall] Hierarchy

```
firewall {
  family (any | ccc | ethernet-switching | inet | inet6 | mpls | vpls) {
    ... the family subhierarchies appear after the main [edit firewall] hierarchy ...
  }
  filter filter-name {
    accounting-profile [ profile-names ];
    enhanced-mode;
    interface-shared-with;
    interface-specific;
    physical-interface-policer;
    term term-name {
      filter filter-name;
      from {
        ... statements in Common IP Firewall Match Conditions on page 29 AND
          statements in Common IPv4 Firewall Match Conditions on page 30 ...
      }
      then {
        ... statements in Common Firewall Actions on page 27 AND
          statements in Common IP Firewall Actions on page 28 AND
          statements in Common IPv4 Firewall Actions on page 28 ...
      }
    }
  }
  hierarchical-policer policer-name {
    aggregate {
      if-exceeding {
        bandwidth-limit bps;
        burst-size-limit bytes;
      }
      then {
        discard;
        forwarding-class class-name;
        loss-priority (high | low | medium-high | medium-low);
      }
    }
    logical-interface-policer;
    physical-interface-policer;
    premium {
      if-exceeding {
        bandwidth-limit bps;
        burst-size-limit bytes;
      }
      then {
        discard;
      }
    }
  }
  shared-bandwidth-policer;
    interface-set interface-set-name {
    interface-name;
  }
  load-balance-group group-name {
    next-hop-group [ group-names ];
```

```
    }
    policer policer-name {
      filter-specific;
      if-exceeding {
        (bandwidth-limit bps | bandwidth-percent percentage);
        burst-size-limit bytes;
      }
      logical-bandwidth-policer;
      logical-interface-policer;
      physical-interface-policer;
      then {
        discard;
        forwarding-class class-name;
        loss-priority (high | low | medium-high | medium-low);
      }
    }
    three-color-policer policer-name {
      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;
      }
    }
}

firewall {
    family any {
      filter filter-name {
        interface-shared;
        term term-name {
          from {
            (forwarding-class [ class-names ] | forwarding-class-except [ class-names ]);
            interface interface-name;
            interface-set set-name;
            (loss-priority [ priorities ] | loss-priority-except [ priorities ]);
            (packet-length [ values ] | packet-length-except [ values ]);
          }
          then {
            ... statements in Common Firewall Actions on page 27 PLUS ...
            (accept | discard);
          }
```

```
        }
      }
    }
  }


firewall {
    family ccc {
        filter filter-name {
            accounting-profile [ profile-names ];
            physical-interface-filter;
            interface-specific;
            term term-name {
                filter filter-name;
                from {
                    (forwarding-class [ class-names ] | forwarding-class-except [ class-names ]);
                    (interface-group [ group-names ] | interface-group-except [ group-names ]);
                    (learn-vlan-1p-priority [ priorities ] | learn-vlan-1p-priority [ priorities ]);
                    (loss-priority [ priorities ] | loss-priority-except [ priorities ]);
                    (user-vlan-1p-priority [ priorities ] | user-vlan-1p-priority [ priorities ]);
                }
                then {
                    ... statements in Common Firewall Actions on page 27 PLUS ...
                    (accept | discard);
                    port-mirror-instance instance-name;
                }
            }
        }
    }
}

firewall {
    family ethernet-switching {
        filter filter-name {
            interface-specific;
            term term-name {
                from {
                    destination-address {
                        ip-prefix</prefix-length>;
                    }
                    destination-mac-address {
                        mac-address;
                    }
                    destination-port [ port-names ];
                    destination-prefix-list {
                        list-name;
                    }
                    dot1q-tag [ tag-values ];
                    dot1q-user-priority [ priority-values ];
                    dscp [ code-point-values ];
                    ether-type [ protocol-names ];
                    fragment-flags flag;
                    icmp-code [ codes ];
                    icmp-type [ types ];
                    interface interface-name;
                    is-fragment;
```

```
            precedence [ precedence-names ];
            protocol [ protocol-names ];
            source-address {
              ip-prefix</prefix-length>;
            }
            source-mac-address {
              mac-address;
            }
            source-port [ port-names ];
            source-prefix-list {
              list-name;
            }
            tcp-established;
            tcp-flags flag;
            tcp-initial;
            vlan [ vlan-names ];
          }
          then {
            (accept | discard);
            analyzer analyzer-name;
            count counter-name;
            forwarding-class class-name;
            interface interface-name;
            log;
            loss-priority (high | low);
            policer policer-name;
            syslog;
            vlan vlan-name;
          }
        }
      }
    }
  }

  firewall {
    family inet {
      dialer-filter filter-name {
        accounting-profile [ profile-names ];
        term term-name {
          from {
            ... statements in Common IP Firewall Match Conditions on page 29 AND
              statements in Common IPv4 Firewall Match Conditions on page 30 EXCEPT
              FOR ...
              (ah-spi [ values ] | ah-spi-except [ values ]);   # NOT valid at this level
              (destination-class [ class-names ] |
                destination-class-except [ class-names ]);   # NOT valid at this level
              interface interface-name;   # NOT valid at this level
              (loss-priority [ priorities ] | loss-priority-except [ priorities ]);   # NOT valid at
                this level
              service-filter-hit;   # NOT valid at this level
              (source-class [ class-names ] | source-class-except [ class-names ]);   # NOT
                valid at this level
          }
          then {
            (ignore | note);
            log;
```

```
        sample;
        syslog;
      }
    }
  }
  filter filter-name {
    accounting-profile [ profile-names ];
    interface-specific;
    term term-name {
      filter filter-name;
      from {
        ... statements in Common IP Firewall Match Conditions on page 29 AND
          statements in Common IPv4 Firewall Match Conditions on page 30 ...
      }
      then {
        ... statements in Common Firewall Actions on page 27 AND
          statements in Common IP Firewall Actions on page 28 AND
          statements in Common IPv4 Firewall Actions on page 28 ...
      }
    }
  }
  prefix-action name {
    count;
    destination-prefix-length prefix-length;
    filter-specific;
    policer policer-name;
    source-prefix-length prefix-length;
    subnet-prefix-length prefix-length;
  }
  service-filter filter-name {
    term term-name {
      from {
        address {
          ip-prefix</prefix-length>;
        }
        (ah-spi [ values ] | ah-spi-except [ values ]);
        destination-address {
          ip-prefix</prefix-length>;
        }
        (destination-port [ port-names ] | destination-port-except [ port-names ]);
        destination-prefix-list {
          list-name;
        }
        (esp-spi [ values ] | esp-spi-except [ values ]);
        first-fragment;
        fragment-flags flag;
        (fragment-offset [ offsets ] | fragment-offset-except [ offsets ]);
        (interface-group [ group-names ] | interface-group-except [ group-names ]);
        (ip-options [ option-names ] | ip-options-except [ option-names ]);
        is-fragment;
        (loss-priority [ priorities ] | loss-priority-except [ priorities ]);
        (port [ port-names ] | port-except [ port-names ]);
        prefix-list {
          list-name;
        }
        (protocol [ protocol-names ] | protocol-except [ protocol-names ]);
```

```
            source-address {
              ip-prefix</prefix-length>;
            }
            (source-port [ port-names ] | source-port-except [ port-names ]);
            source-prefix-list {
              list-name;
            }
            tcp-flags flag-name;
          }
          then {
            count counter-name;
            log;
            port-mirror;
            sample;
            (service | skip);
          }
        }
      }
      simple-filter filter-name {
        term term-name {
          from {
            destination-address ip-prefix</prefix-length>;
            destination-port port-name;
            forwarding-class [ class-names ];
            protocol protocol-name;
            source-address ip-prefix</prefix-length>;
            source-port port-name;
          }
          then {
            forwarding-class class-name;
            loss-priority (high | low | medium-high | medium-low);
            policer policer-name;
          }
        }
      }
    }
  }

  firewall {
    family inet6 {
      dialer-filter filter-name {
        accounting-profile [ profile-names ];
        term term-name {
          from {
            ... statements in Common IP Firewall Match Conditions on page 29 PLUS ...
              (next-header [ protocol-types ] | next-header-except [ protocol-types ]);
            ... BUT NOT ...
              (destination-class [ class-names ] |
                destination-class-except [ class-names ]);   # NOT valid at this level
              (forwarding-class [ class-names ] |
                forwarding-class-except [ class-names ]);   # NOT valid at this level
              interface interface-name;   # NOT valid at this level
              (interface-group [ group-names ] | interface-group-except [ group-names ]);  #
                NOT valid at this level
              (loss-priority [ priorities ] | loss-priority-except [ priorities ]);   # NOT valid at
                this level
```

```
            service-filter-hit;   # NOT valid at this level
            (source-class [ class-names ] | source-class-except [ class-names ]);   # NOT
              valid at this level
            tcp-established;   # NOT valid at this level
            tcp-flags flag;   # NOT valid at this level
            tcp-initial;   # NOT valid at this level
        }
        then {
          (ignore | note);
          log;
          sample;
          syslog;
        }
      }
    }
    filter filter-name {
      accounting-profile [ profile-names ];
      interface-specific;
      term term-name {
        filter filter-name;
        from {
          ... statements in Common IP Firewall Match Conditions on page 29 PLUS ...
          (next-header [ protocol-types ] | next-header-except [ protocol-types ]);
          (traffic-class [ code-point-values ] | traffic-class-except [ code-point-values ]);
        }
        then {
          ... statements in Common Firewall Actions on page 27 AND
            statements in Common IP Firewall Actions on page 28 PLUS ...
          (accept | discard | reject <address-unreachable | administratively-prohibited |
            beyond-scope | fragmentation-needed | no-route | port-unreachable |
            tcp-reset>);
        }
      }
    }
    service-filter filter-name {
      term term-name {
        from {
          address {
            ip-prefix</prefix-length>;
          }
          (ah-spi [ values ] | ah-spi-except [ values ]);
          destination-address {
            ip-prefix</prefix-length>;
          }
          (destination-port [ port-names ] | destination-port-except [ port-names ]);
          destination-prefix-list {
            list-name;
          }
          (esp-spi [ values ] | esp-spi-except [ values ]);
          (interface-group [ group-names ] | interface-group-except [ group-names ]);
          (next-header [ protocol-types ] | next-header-except [ protocol-types ]);
          (port [ port-names ] | port-except [ port-names ]);
          prefix-list {
            list-name;
          }
          source-address {
```

```
            ip-prefix</prefix-length>;
          }
          (source-port [ port-names ] | source-port-except [ port-names ]);
          source-prefix-list {
            list-name;
          }
          tcp-flags flag-name;
        }
        then {
          count counter-name;
          log;
          port-mirror;
          sample;
          (service | skip);
        }
      }
    }
  }
}

firewall {
  family mpls {
    filter filter-name {
      accounting-profile [ profile-names ];
      interface-specific;
      physical-interface-filter;
      term term-name {
        from {
          (exp [ exp-bits ] | exp-except [ exp-bits ]);
        }
        then {
          (ignore | note);
          log;
          sample;
          syslog;
        }
      }
    }
    filter filter-name {
      accounting-profile [ profile-names ];
      interface-specific;
      physical-interface-filter;
      term term-name {
        filter filter-name;
        from {
          (exp [ exp-bits ] | exp-except [ exp-bits ]);
          (forwarding-class [ class-names ] | forwarding-class-except [ class-names ]);
          interface interface-name;
          interface-set set-name;
          (loss-priority [ priorities ] | loss-priority-except [ priorities ]);
        }
        then {
          ... statements in Common Firewall Actions on page 27 PLUS ...
          (accept | discard);
          sample;
        }
```

```
        }
      }
    }
  }

  firewall {
    family vpls {
      filter filter-name {
        accounting-profile [ profile-names ];
        interface-specific;
        term term-name {
          filter filter-name;
          from {
            ... statements in Common Layer 2 Firewall Match Conditions on page 30 ...
          }
          then {
            ... statements in Common Firewall Actions on page 27 PLUS ...
            (accept | discard);
            port-mirror;
            port-mirror-instance instance-name;
          }
        }
      }
    }
  }
```

Related Documentation

- Notational Conventions Used in Junos OS Configuration Hierarchies

dscp (Multifield Classifier)

Syntax dscp [0 | *value*];

Hierarchy Level [edit firewall family *family-name* filter *filter-name* term *term-name* then]

Release Information Statement introduced in Junos OS Release 7.4.

Description For M320 and T Series routers, set the DSCP field of incoming or outgoing packets to **000000**. On the same packets, you can use a behavior aggregate (BA) classifier and a rewrite rule to rewrite the MPLS EXP field.

For MX Series routers with MPCs and EX Series switches, the DSCP field can be set from a numeric range.

For MX Series routers and EX Series switches, if you configure a firewall filter with a DSCP action or traffic-class action on a DPC, the commit does not fail, but the filter is not applied to the interface, a warning displays, and an entry is made in the syslog.

Options *value*—For MX Series routers with MPCs and EX Series switches, specify the field of incoming or outgoing packets in the range from **0** through **63**.

Required Privilege Level firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Applying Tricolor Marking Policers to Firewall Filters

family (Multifield Classifier)

Syntax

```
family family-name {
  filter filter-name {
    term term-name {
      from {
        match-conditions;
      }
      then {
        dscp 0;
        forwarding-class class-name;
        loss-priority (high | low);
        three-color-policer {
          (single-rate | two-rate) policer-name;
        }
      }
    }
  }
}
```

Hierarchy Level [edit firewall]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure a firewall filter for IP version 4 (IPv4) or IP version 6 (IPv6) traffic.

Options *family-name*—Protocol family:

- **ccc**—Circuit cross-connect parameters
- **inet**—IPv4 parameters
- **inet6**—IPv6 protocol parameters
- **iso**—OSI ISO protocol parameters
- **mpls**—MPLS protocol parameters
- **tcc**—Translational cross-connect parameters
- **vpls**—Virtual private LAN service parameters.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Multifield Classifiers on page 9

filter (Configuring)

Syntax

```
filter filter-name {
  accounting-profile name;
  enhanced-mode;
  interface-shared;
  interface-specific;
  physical-interface-filter;
  term term-name {
    filter filter-name;
    from {
      match-conditions;
    }
    then {
      actions;
    }
  }
}
```

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 at the **[edit dynamic-profiles ... family family-name]** hierarchy level introduced in Junos OS Release 11.4.
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 (" "). In Junos OS Release 9.0 and later, you can no longer use special characters within the name of a firewall filter. 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.

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 Standard Firewall Filters
- Guidelines for Applying Standard Firewall Filters
- Configuring Multifield Classifiers on page 9
- Using Multifield Classifiers to Set PLP

- simple-filter

forwarding-class (Multifield Classifiers)

Syntax forwarding-class *class-name*;

Hierarchy Level [edit firewall family *family-name* filter *filter-name* term *term-name* then]

Release Information Statement introduced before Junos OS Release 7.4.

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
- Configuring Multifield Classifiers on page 9

from

Syntax
```
from {
  applications [ application-name ];
  application-sets [ set-name ];
  destination-address (CoS) address;
  source-address address;
}
```

Hierarchy Level [edit services cos rule *rule-name* term *term-name*]

Release Information Statement introduced in Junos OS Release 8.1.

Description Specify input conditions for a CoS term.

Options The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring CoS Rule Sets

loss-priority (Normal Filter)

Syntax loss-priority (high | low);

Hierarchy Level [edit firewall family *family-name* filter *filter-name* term *term-name* then]

Release Information Statement introduced before Junos OS Release 7.4.

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
- Configuring Multifield Classifiers on page 9

loss-priority (Simple Filter)

Syntax loss-priority (high | low | medium);

Hierarchy Level [edit firewall family *family-name* simple-filter *filter-name* term *term-name* then]

Release Information Statement introduced in Junos OS Release 7.6.

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
- Configuring Multifield Classifiers on page 9

simple-filter (Configuring)

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 filter *filter-name*]

Release Information Statement introduced in Junos OS Release 7.6.

Description Define a simple filter. Simple filters are recommended for metropolitan Ethernet applications.

Options

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-name—Name that identifies the term. 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 (“ ”).

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.

The remaining statements are explained separately. Only **forwarding-class** and **loss-priority** are valid 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

- Configuring Multifield Classifiers on page 9
- filter (Applying to an Interface) on page 60
- simple-filter (Applying to an Interface) on page 61

term (Simple Filter)

Syntax

```
term term-name {
  from {
    match-conditions;
  }
  then {
    forwarding-class class-name;
    loss-priority (high | low | medium);
  }
}
```

Hierarchy Level [edit firewall family inet simple-filter *filter-name*]

Release Information Statement introduced in Junos OS Release 7.6.

Description Define a simple filter term.

Options

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-name—Name that identifies the term. 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 (“ ”).

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. For CoS, only the actions listed are allowed. These statements are explained separately.

Required Privilege Level
firewall—To view this statement in the configuration.
firewall-control—To add this statement to the configuration.

Related Documentation

- Multifield Classification
- Simple Filter Overview
- Standard Firewall Filter Match Conditions for IPv4 Traffic
- Standard Firewall Filter Match Conditions for IPv6 Traffic

then

Syntax

```
then {
  application-profile profile-name;
  dscp (alias | bits);
  forwarding-class class-name;
  syslog;
  (reflexive | reverse) {
    application-profile profile-name;
    dscp (alias | bits);
    forwarding-class class-name;
    syslog;
  }
}
```

Hierarchy Level [edit services cos rule *rule-name* term *term-name*]

Release Information Statement introduced in Junos OS Release 8.1.

Description Define the CoS term actions.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Actions in a CoS Rule
- Configuring Actions in CoS Rules

[edit interfaces] Hierarchy Level

The following statement hierarchy can also be included at the **[edit logical-systems *logical-system-name*]** hierarchy level.

```
interfaces {
  interface-name {
    ... the "interface-name" subhierarchy appears after the main [edit interfaces] hierarchy
      level ...
  }
  interface-set interface-set-name {
    interface interface-name {
      (unit unit-number | vlan-tags-outer vlan-tag);
    }
  }
  irb (Interfaces) {
    accounting-profile name;
    description  text;
    disable;

    (gratuitous-arp-reply | no-gratuitous-arp-reply);
    hold-time up milliseconds down milliseconds;
```

```
mtu bytes;
no-gratuitous-arp-request;

traceoptions {
  flag flag;
}
(traps | no-traps);
unit logical-unit-number {
  accounting-profile name;
  bandwidth rate;
  description text;
  disable;
  encapsulation type;
  family inet {
    accounting {
      destination-class-usage;
      source-class-usage {
        input;
        output;
      }
    }
    address ipv4-address {
      arp ip-address (mac | multicast-mac) mac-address <publish>;
      broadcast address;
      preferred;
      primary;
      vrrp-group group-id {
        (accept-data | no-accept-data);
        advertise-interval seconds;
        advertisements-threshold number;
        authentication-key key;
        authentication-type authentication;
        fast-interval milliseconds;
        (preempt | no-preempt) {
          hold-time seconds;
        }
        priority number;
        track {
          interface interface-name {
            bandwidth-threshold bits-per-second priority-cost priority;
            priority-cost priority;
          }
          priority-hold-time seconds;
          route prefix/prefix-length routing-instance instance-name priority-cost priority;
        }
        virtual-address [ addresses ];
        vrrp-inherit-from vrrp-group;
      }
    }
    filter {
      input filter-name;
      output filter-name;
    }
    mtu bytes;
    no-neighbor-learn;
```

```
    no-redirects;
    primary;
    rpf-check {
      fail-filter filter-name;
      mode {
        loose;
      }
    }
    targeted-broadcast {
      forward-and-send-to-re;
      forward-only;
    }
  }
  family inet6 {
    accounting {
      destination-class-usage;
      source-class-usage {
        input;
        output;
      }
    }
    address address {
      eui-64;
      ndp ip-address (mac | multicast-mac) mac-address <publish>;
      preferred;
      primary;
      vrrp-inet6-group group-id {
        accept-data | no-accept-data;
        advertisements-threshold number;
        authentication-key key;
        authentication-type authentication;
        fast-interval milliseconds;
        inet6-advertise-interval milliseconds;
        preempt | no-preempt {
          hold-time seconds;
        }
        priority number;
        track {
          interface interface-name {
            bandwidth-threshold bandwidth priority-cost number;
            priority-cost number;
          }
          priority-hold-time seconds;
          route ip-address/mask routing-instance instance-name priority-cost cost;
        }
        virtual-inet6-address [addresses];
        virtual-link-local-address ipv6–address;
        vrrp-inherit-from {
          active-group group-number;
          active-interface interface-name;
        }
      }
    }
    (dad-disable | no-dad-disable);
    filter {
      input filter-name;
```



```
            output filter-name;
          }
          mtu bytes;
          nd6-stale-time seconds;
          no-neighbor-learn;
          no-redirects;
          policer {
            input policer-name;
            output policer-name;
          }
          rpf-check {
            fail-filter filter-name;
            mode {
              loose;
            }
          }
        }
        family iso {
          address interface-address;
          mtu bytes;
        }
        family mpls {
          filter {
            input filter-name;
            output filter-name;
          }
          mtu bytes;
          policer {
            input policer-name;
            output policer-name;
          }
        }
        native-inner-vlan-id vlan-id;
        proxy-arp (restricted | unrestricted);
        (traps | no-traps);
        vlan-id-list [vlan-id's];
        vlan-id-range [vlan-id-range];
      }
    }
    traceoptions {
      file <filename> <files number> <match regular-expression> <size maximum-file-size>
        <world-readable | no-world-readable>;
      flag flag <disable>;
      no-remote-trace;
    }
}

interfaces {
    interface-name {
      disable;
      accounting-profile name;
      aggregated-ether-options {
        ethernet-switch-profile {
          tag-protocol-id [ hexadecimal-identifiers ];
        }
        (flow-control | no-flow-control);
```

```
    lacp {
      (active | passive);
      admin-key key;
      fast-failover;
      link-protection {
        disable;
        (revertive | non-revertive);
      }
      periodic (fast | slow);
      system-id mac-address;
      system-priority priority;
    }
    (link-protection | no-link-protection);
    link-speed (100m | 1g | 8g | 10g | 40g | 50g | 80g | 100g | oc192);
    logical-interface-fpc-redundancy;
    (loopback | no-loopback);
    mc-ae {
      chassis-id chassis-id;
      events {
        iccp-peer-down {
          force-icl-down;
          prefer-status-control-active;
        }
      }
      mc-ae-id mc-ae-id;
      mode (active-active | active-standby);
      redundancy-group group-id;
      status-control (active | standby);
    }
    minimum-links number;
    rebalance-periodic {
      start-time time;
      interval number;
    }
    source-address-filter {
      mac-address;
    }
    (source-filtering | no-source-filtering);
  }
  auto-configure {
    remove-when-no-subscribers;
    stacked-vlan-ranges {
      access-profile profile-name;
      authentication {
        password password-string;
        username-include {
          circuit-type;
          delimiter delimiter-character;
          domain-name domain-name-string;
          interface-name;
          mac-address;
          option-82 ( circuit-id | remote-id);
          radius-realm radius-realm-string;
          user-prefix user-prefix-string;
        }
      }
```

```
      dynamic-profile profile-name {
        accept (any | dhcp-v4 | dhcp-v6 | inet | inet6);
        ranges (any | low-tag–high-tag),(any | low-tag–high-tag);
      }
    }
    vlan-ranges {
      access-profile profile-name;
      authentication {
        password password-string;
        username-include {
          circuit-type;
          delimiter delimiter-character;
          domain-name domain-name-string;
          interface-name;
          mac-address;
          option-82;
          radius-realm radius-realm-string;
          user-prefix user-prefix-string;
        }
      }
      dynamic-profile profile-name {
        accept (any | dhcp-v4 | dhcp-v6 | inet | inet6);
        ranges (any | low-tag)–(any | high-tag);
      }
    }
    override tag vlan-tag dynamic-profile profile name;
  }
  encapsulation (ethernet-bridge | ethernet-vpls | extended-vlan-bridge |
    extended-vlan-vpls | flexible-ethernet-services | vlan-vpls);
  ether-options {
    802.3ad {
      aex;
      (backup | primary);
      lacp {
        force-up;
        port-priority
      }
    }
    asynchronous-notification;
    (auto-negotiation | no-auto-negotiation);
    ethernet-switch-profile {
      ethernet-policer-profile {
        input-priority-map {
          ieee802.1p premium [ values ];
        }
        output-priority-map {
          classifier {
            premium {
              forwarding-class class-name {
                loss-priority (high | low);
              }
            }
          }
        }
        policer cos-policer-name {
          aggregate {
```

```
            bandwidth-limit bps;
            burst-size-limit bytes;
          }
          premium {
            bandwidth-limit bps;
            burst-size-limit bytes;
          }
        }
        tag-protocol-id;
      }
      (mac-learn-enable | no-mac-learn-enable);
    }
    (flow-control | no-flow-control);
    ignore-l3-incompletes;
    link-mode (automatic | full-duplex | half-duplex);
    (lloopback | no-loopback);
    keepalives <interval seconds> <down-count number> <up-count number>;
    speed (1g | 10m | 100m | 10m-100m | auto-negotiation);
    source-address-filter {
      mac-address;
    }
    source-filtering | no-source-filtering;
  }
  flexible-vlan-tagging;
  (gratuitous-arp-reply | no-gratuitous-arp-reply);
  hold-time (up milliseconds | down milliseconds);
  interface-transmit-statistics;
  (keepalives <down-count number> <interval seconds> <up-count number> |
    no-keepalives);
  layer2-policer {
    apply-groups [ group-names ];
    apply-groups-except [ group-names ];
  }
  link-mode (automatic | full-duplex);
  mac mac-address;
  mtu bytes;
  multi-chassis-protection peer-ip-address {
    interface interface-name;
  }
  native-vlan-id number;
  no-gratuitous-arp-request;
  optics-options {
    alarm low–light–alarm {
      (link-down | syslog);
    }
    warning low–light–warning {
      (link-down | syslog);
    }
    wavelength nm;
  }
  passive-monitor-mode;
  per-unit-scheduler;
  speed (10m | 100m | 1g | auto | oc3 | oc12 | oc48);
  stacked-vlan-tagging;
  traceoptions {
    flag flag;
```

```
    }
    transmit-bucket {
      overflow discard;
      rate percentage;
      threshold bytes;
    }
    (traps | no-traps);
    unidirectional;
    vlan-tagging;
  }


  interface-name {
    unit logical-unit-number {
      disable;
      accept-source-mac {
        mac-address mac-address {
          policer {
            input policer-name;
            output policer-name;
          }
        }
      }
      account-layer2-overhead (Interface Level) {
        value;
        egress bytes;
        ingress bytes;
      }
      accounting-profile name;
      advisory-options {
        downstream-rate rate;
        upstream-rate rate;
      }
      arp-resp (restricted|unrestricted);
      bandwidth rate;
      clear-dont-fragment-bit;
      copy-tos-to-outer-ip-header;
      demux-destination family;
      encapsulation (vlan-bridge | vlan-vpls);
      epd-threshold cells plp1 cells;
      filter filter-name;
      inner-vlan-id-range start start-id end end-id;
      input-vlan-map {
        (pop | pop-pop | pop-swap | push | push-push | swap | swap-push | swap-swap);
        inner-tag-protocol-id tpid;
        inner-vlan-id number;
        tag-protocol-id tpid;
        vlan-id number;
      }
      interface-shared-with psdnumerical-index;
      layer2-policer {
        input-hierarchical-policer policer-name;
        input-policer policer-name;
        input-three-color policer-name;
        output-policer policer-name;
        output-three-color policer-name;
```

```
    }
    multi-chassis-protection peer-ip-address {
      interface interface-name;
    }
    native-inner-vlan-id number;
    output-vlan-map {
      (pop | pop-pop | pop-swap | push | push-push | swap | swap-push | swap-swap);
      inner-tag-protocol-id tpid;
      inner-vlan-id number;
      tag-protocol-id tpid;
      vlan-id number;
    }
    peer-interface interface-name;
    peer-unit unit-number;
    plp-to-clp;
    proxy-arp <restricted | unrestricted>;
    rpm {
      (client | server);
      twamp-server;
    }
    swap-by-poppush;
    vlan-id number;
    vlan-id-list [ vlan-id vlan-id-vlan-id ];
    vlan-id-range number-number;
    vlan-tags (inner <tpid.>vlan-id | inner-list [vlan-id vlan-id–vlan-id ] |
      inner-range <tpid.>vlan-id–vlan-id) outer <tpid.>vlan-id;
}

unit logical-unit-number {
    family ethernet-switching {
      filter{
        group filter-group-number;
        (input filter-name | input-list [ filter-names ]);
      (output filter-name | output-list [ filter-names ]);
      (inner-vlan-id-list [ vlan-ids ] | vlan-id number | vlan-id-list [ number
        number–number ]);
      interface-mode (access | trunk);
      policer {
        input policer-name;
        output policer-name;
      }
      vlan-rewrite {
        translate old-vlan-id new-vlan-id;
      }
      vlan {
        members [ all vlan-identifiers ];
      }
    }
    family inet {
      filter {
        group filter-group-number;
        (input filter-name | input-list [ filter-names ]);
        (output filter-name | output-list [ filter-names ]);
      }
      input-hierarchical-policer policer-name;
      mac-validate (loose | strict);
```



```
    mtu bytes;
    no-neighbor-learn;
    no-redirects;
    policer {
      arp policer-template-name;
      input policer-name;
      output policer-name;
    }
    primary;
    receive-options-packets;
    receive-ttl-exceeded;
    rpf-check {
      fail-filter filter-name;
      mode loose;
    }
    sampling {
      (input | output | input output);
    }
    simple-filter {
      input filter-name;
    }
    targeted-broadcast {
      forward-and-send-to-re;
      forward-only;
    }
    unnumbered-address interface-name <destination address>
      <destination-profile profile-name> <preferred-source-address address>;

  }

  family inet6 {
    address ipv6-address {
      destination destination-address;
      eui-64;
      ndp ipv6-address <l2-interface interface-name> <(mac mac-address |
        multicast-mac multicast-mac-address) <publish>>;
      preferred;
      primary;
      vrrp-inet6-group group-number {
        (accept-data | no-accept-data);
        fast-interval milliseconds;
        inet6-advertise-interval seconds;
        (no-preempt; | ... the following preempt statement ...)
        preempt {
          hold-time seconds;
        }
        priority number;
        track {
          interface interface-name {
            bandwidth-threshold bits-per-second priority-cost priority;
            priority-cost priority;
          }
          priority-hold-time seconds;
          route ip-address-prefix/prefix-length routing-instance instance-name
            priority-cost priority;
        }
```

```
            virtual-inet6-address [ addresses ];
            virtual-link-local-address ipv6-address;
            vrrp-inherit-from {
              active-group group-number;
              active-interface interface-name;
            }
          }
        }
        (dad-disable | no-dad-disable);
        filter {
          group filter-group-number;
          (input filter-name | input-list [ filter-names ]);
          (output filter-name | output-list [ filter-names ]);
        }
        input-hierarchical-policer policer-name;
        mtu bytes;
        nd6-stale-time seconds;
        no-neighbor-learn;
        policer {
          input policer-name;
          output policer-name;
        }
        rpf-check {
          fail-filter filter-name;
          mode loose;
        }
        sampling {
          (input | output | input output);
        }
        unnumbered-address interface-name preferred-source-address address;

      }

      family iso {
        address iso-address;
        mtu bytes;
      }

      family mlfr-end-to-end {
        bundle logical-interface-name;
      }



      family mpls {
        filter {
          group filter-group-number;
          (input filter-name | input-list [ filter-names ]);
          (output filter-name | output-list [ filter-names ]);
        }
        input-hierarchical-policer policer-name;
        maximum-labels maximum-labels;
        mtu bytes;
        policer {
          input policer-name;
          output policer-name;
```

```
        }
      }


      family vpls {
        core-facing;
        filter {
          group filter-group-number;
          (input filter-name | input-list [ filter-names ]);
          (output filter-name | output-list [ filter-names ]);
        }
        policer {
          input policer-name;
          output policer-name;
        }
      }

    }
  }
}
```

Related Documentation

- Notational Conventions Used in Junos OS Configuration Hierarchies

filter (Applying to an Interface)

Syntax

```
filter {
    input filter-name;
    output filter-name;
}
```

Hierarchy Level

[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.

Description

Apply a filter to an interface. You can also use filters for encrypted traffic. When you configure filters, you can configure the family **inet**, **inet6**, **mpls**, or **vpls** only.

Options

input *filter-name*—Name of one filter to evaluate when packets are received on the interface.

output *filter-name*—Name of one filter to evaluate when packets are 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

- simple-filter on page 61
- Applying Firewall Filter Tricolor Marking Policers to Interfaces
- Example: Classifying Packets Based on Their Destination Address
- Example: Configuring and Verifying a Complex Multifield Filter on page 14
- Example: Writing Different DSCP and EXP Values in MPLS-Tagged IP Packets
- Example: Configuring a Simple Filter on page 19
- Example: Configuring a Logical Bandwidth Policer on page 20
- Example: Two-Color Policers and Shaping Rate Changes on page 21

simple-filter (Applying to an Interface)

Syntax

```
simple-filter {
    input filter-name;
}
```

Hierarchy Level

[edit interfaces *interface-name* unit *logical-unit-number* family inet],
[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number* family inet]

Release Information

Statement introduced in Junos OS Release 7.6.

Description

Apply a simple filter to an interface. You can apply simple filters to the family **inet** only, and only in the input direction.

Options

input *filter-name*—Name of one filter to evaluate when packets are received 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

- Configuring Multifield Classifiers on page 9
- filter on page 60