Engineering
Simplicity

Junos® OS

Class of Service User Guide for Routers

Published
2025-12-15

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

YEAR 2000 NOTICE

END USER LICENSE AGREEMENT

Table of Contents

About This Guide

Use this guide to understand and configure class of service (CoS) features in Junos OS to define service levels that provide different delay, jitter, and packet loss characteristics to particular applications served by specific traffic flows. Applying CoS features to each device in your network ensures quality of service (QoS) for traffic throughout your entire network. This guide applies to all Juniper routing devices.

RELATED DOCUMENTATION

Day One: Junos QoS for IOS Engineers

Day One: Deploying Basic QoS

Advanced Junos CoS Troubleshooting Cookbook

1

PART

Overview

CHAPTER 1

How Class of Service Manages Congestion and Defines Traffic Forwarding Behavior

IN THIS CHAPTER

How Class of Service Manages Congestion and Controls Service Levels in the Network

IN THIS SECTION

Usually, IP devices forward packets independently and without any control on throughput or delay. This is known as *best-effort* service. This service is as good as the network equipment and links, and the result is satisfactory for many traditional IP applications emphasizing data delivery, such as e-mail or Web browsing. However, IP applications such as real-time video and audio (or voice) require lower delay, jitter, and loss parameters than simple best-effort networks can provide during times of network congestion.

When a network experiences congestion and delay, some packets must be dropped. The Juniper Networks Junos operating systems (Junos OS and Junos OS Evolved) *class of service* (CoS) enables you to assign traffic to classes and offer various levels of throughput and packet loss when congestion occurs.

CoS is the assignment of traffic flows to different service levels. Service providers can use router-based CoS features to define service levels that provide different delay, *jitter* (delay variation), and packet loss characteristics to particular applications served by specific traffic flows.

A router cannot compromise best-effort forwarding performance in order to deliver CoS features, because this merely trades one problem for another. When CoS features are enabled, they must allow routers to better process critical packets as well as best-effort traffic flows, even during times of congestion. Network throughput is determined by a combination of available bandwidth and delay. CoS guarantees a minimum bandwidth dedicated to a service class.

The main impact of CoS on network delay is in queuing delays, when packets are normally queued for output in the order of arrival, regardless of service class. Queuing delays increase with network congestion and often result in lost packets when queue buffers overflow. The other two elements of overall network delay, serial transmission delays determined by link speeds and propagation delays determined by media type, are not determined by CoS settings.

For interfaces that carry IPv4, IPv6, and MPLS traffic, you can configure the Junos CoS features to provide multiple classes of service for different applications. On the routing device, you can configure multiple forwarding classes for transmitting packets, define which packets are placed into each output queue, schedule the transmission service level for each queue, and manage congestion using a random early detection (RED) algorithm.

The Junos CoS features provide a set of mechanisms that you can use to provide differentiated services when best-effort traffic delivery is insufficient. In designing CoS applications, you must give careful consideration to your service needs, and you must thoroughly plan and design your CoS configuration to ensure consistency across all routing devices in a CoS domain. You must also consider all the routing devices and other networking equipment in the CoS domain to ensure interoperability among all equipment.

CoS Applications

You can configure CoS features to meet the needs of multiple applications. Because the components are generic, you can use a single CoS configuration syntax across multiple routing devices. CoS mechanisms are useful for two broad classes of applications. These applications can be referred to as *in the box* and *across the network*.

In-the-box applications use CoS mechanisms to provide special treatment for packets passing through a single node on the network. You can monitor the incoming traffic on each interface, using CoS to provide preferred service to some interfaces (that is, to some customers) while limiting the service provided to other interfaces. You can also filter outgoing traffic by the packet's destination, thus providing preferred service to some destinations.

Across-the-network applications use CoS mechanisms to provide differentiated treatment to different classes of packets across a set of nodes in a network. In these types of applications, you typically control the ingress and egress routing devices to a routing domain and all the routing devices within the domain. You can use the Junos CoS features to modify packets traveling through the domain to indicate the packet's priority across the domain.

Specifically, you modify the CoS code points in packet headers, remapping these bits to values that correspond to levels of service. When all devices in the domain are configured to associate the precedence bits with specific service levels, packets with the same code points traveling across the domain receive the same level of service from the ingress point to the egress point. For CoS to work in this case, the mapping between the code points and service levels must be identical across all routing devices in the domain.

The Junos CoS applications support the following range of mechanisms:

- Differentiated Services (DiffServ)—The CoS application supports DiffServ, which uses a 6-bit differentiated services code point (DSCP) in the differentiated services field of the IPv4 and IPv6 packet header. For IPv6, DSCP is referred to as *traffic class*. The configuration uses DSCP values to determine the forwarding class associated with each packet. IPv4 traffic can also use the 3-bit IP precedence bits to classify traffic.

- Layer 2 to Layer 3 CoS mapping—The CoS application supports mapping of Layer 2 (IEEE 802.1p) packet headers to routing device forwarding class and loss-priority values.

  Layer 2 to Layer 3 CoS mapping involves setting the forwarding class and loss priority based on information in the Layer 2 header. Output involves mapping the forwarding class and loss priority to a Layer 2-specific marking. You can mark the Layer 2 and Layer 3 headers simultaneously.

- MPLS EXP—Supports configuration of mapping of MPLS experimental (EXP) bit settings to routing device forwarding classes and vice versa.

- VPN outer-label marking—Supports setting of outer-label EXP bits, also known as CoS bits, based on MPLS EXP mapping.

CoS Standards

The standards for Junos CoS capabilities are defined in the following RFCs:

- RFC 2474, *Definition of the Differentiated Services Field in the IPv4 and IPv6 Headers*

- RFC 2475, *An Architecture for Differentiated Services*
- RFC 2597, *Assured Forwarding PHB Group*
- RFC 2598, *An Expedited Forwarding PHB*
- RFC 2698, *A Two Rate Three Color Marker*
- RFC 2983, *Differentiated Service and Tunnels*
- RFC 3260, *New Terminology and Clarifications for DiffServ*
- RFC 3317, *Differentiated Services Quality of Service Policy Information Base*

RELATED DOCUMENTATION

Junos CoS Components Used to Manage Congestion and Control Service Levels | 6

How CoS Applies to Packet Flow Across a Network

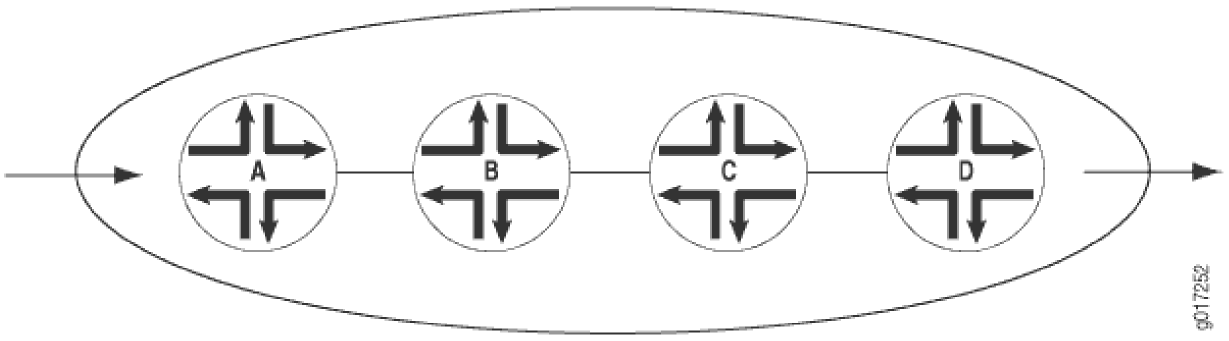
CoS works by examining traffic entering at the edge of your network. The edge routing devices classify traffic into defined service groups to provide the special treatment of traffic across the network. For example, voice traffic can be sent across certain links, and data traffic can use other links. In addition, the data traffic streams can be serviced differently along the network path to ensure that higher-paying customers receive better service. As the traffic leaves the network at the far edge, you can reclassify the traffic.

To support CoS, you must configure each device in the network. Generally, each device examines the packets that enter it to determine their CoS settings. These settings then dictate which packets are first transmitted to the next downstream device. In addition, the devices at the edges of the network might be required to alter the CoS settings of the packets that enter the network from the customer or peer networks.

In Figure 1 on page 6, Router A is receiving traffic from a customer network. As each packet enters, Router A examines the packet’s current CoS settings and classifies the traffic into one of the groupings defined by the Internet service provider (ISP). This definition allows Router A to prioritize its resources for servicing the traffic streams it is receiving. In addition, Router A might alter the CoS settings (forwarding class and loss priority) of the packets to better match the ISP’s traffic groups. When Router B receives the packets, it examines the CoS settings, determines the appropriate traffic group, and processes the packet according to those settings. It then transmits the packets to Router C, which performs the same actions. Router D also examines the packets and determines the appropriate group.

Because Router D sits at the far end of the network, the ISP might decide once again to alter the CoS settings of the packets before Router D transmits them to the neighboring network.

Figure 1: Packet Flow Across the Network



RELATED DOCUMENTATION

Junos CoS Components Used to Manage Congestion and Control Service Levels | 6

Junos CoS Components Used to Manage Congestion and Control Service Levels

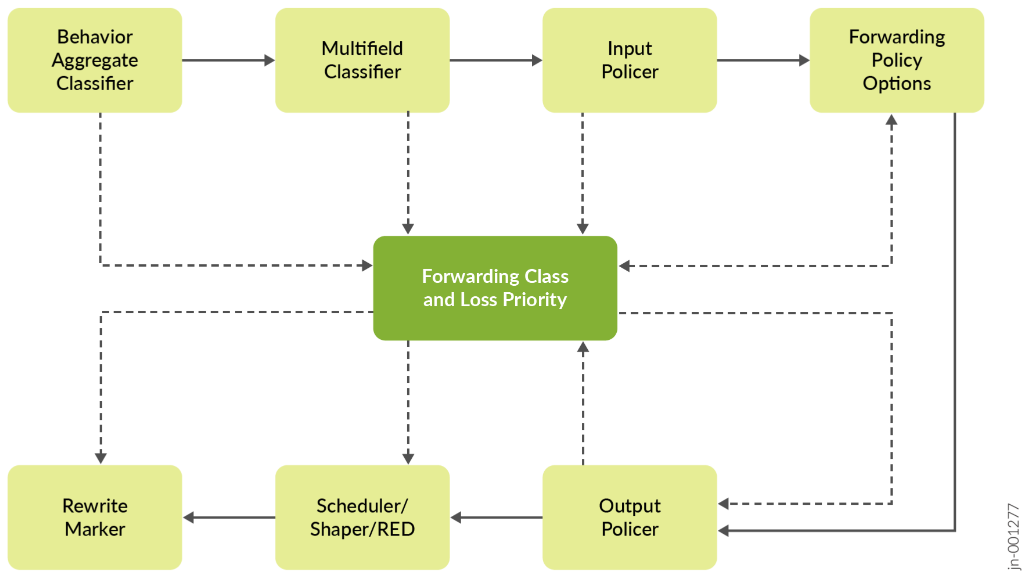
Any CoS implementation must work consistently end to end through the network. A standards-based, vendor-neutral CoS implementation satisfies this requirement best. Junos CoS features interoperate with other vendors' CoS implementations because they are based on IETF Differentiated Services (DiffServ) standards. Junos CoS consists of many components that you can combine and tune to provide the level of services required by customers.

DiffServ specifications establish a six-bit field in the IPv4 and IPv6 packet header to indicate the service class that should be applied to the packet. The bit values in the DiffServ field form DiffServ code points (DSCPs) that can be set by the application or a router on the edge of a DiffServ-enabled network.

Although CoS methods such as DiffServ specify the position and length of the DSCP in the packet header, the implementation of the router mechanisms to deliver DiffServ internally is vendor-specific. CoS functions in Junos OS are configured through a series of mechanisms that you can configure individually or in combination to define particular service offerings.

Figure 2 on page 7 shows the components of the Junos CoS features, illustrating the sequence in which they interact.

Figure 2: Packet Flow Through CoS-Configurable Components



You can configure one or more of the following Junos CoS mechanisms:

- Classifiers—*Packet classification* refers to the examination of an incoming packet. This function associates the packet with a particular CoS servicing level. In Junos, classifiers associate incoming packets with a forwarding class and loss priority and, based on the associated forwarding class, assign packets to output queues. Two general types of classifiers are supported:

  - Behavior aggregate classifiers—A *behavior aggregate* (BA) is a method of classification that operates on a packet as it enters the routing device. The CoS value in the packet header is examined, and this single field determines the CoS settings applied to the packet. BA classifiers allow you to set the forwarding class and loss priority of a packet based on the Differentiated Services code point (DSCP) value, DSCP IPv6 value, IP precedence value, MPLS EXP bits, and IEEE 802.1p value. The default classifier is based on the IP precedence value.

    (You can also configure *code-point aliases* which assign a name to a pattern of code-point bits. You can use this name instead of the bit pattern when you configure other CoS components, such as classifiers, drop-profile maps, and *rewrite rules*.)

    See "Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic" on page 60 for more information on BA classifiers.

  - Multifield traffic classifiers—A *multifield* classifier is a second method for classifying traffic flows. Unlike a behavior aggregate, a multifield classifier can examine multiple fields in the packet. Examples of some fields that a multifield classifier can examine include the source and destination address of the packet as well as the source and destination port numbers of the packet. With multifield classifiers, you set the forwarding class and loss priority of a packet based on *firewall filter* rules. Multifield classification is usually done at the edge of the network for packets that do not have valid or trusted behavior aggregate code points.

See "Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields" on page 148 for more information on multifield classifiers.

- Forwarding classes—The *forwarding classes* affect the forwarding, scheduling, and marking policies applied to packets as they transit a routing device. Known as ordered aggregates in the DiffServ architecture, the forwarding class plus the loss priority determine the router's per-hop behavior (PHB in DiffServ) for CoS. Four categories of forwarding classes are supported: best effort, assured forwarding, expedited forwarding, and network control. Up to 16 forwarding classes are supported, so you can classify packets more granularly. For example, you can configure multiple classes of expedited forwarding (EF) traffic: EF, EF1, and EF2.

  See "Understanding How Forwarding Classes Assign Classes to Output Queues" on page 286 for more information on forwarding classes.

- Loss priorities—*Loss priorities* allow you to set the priority of dropping a packet. Loss priority affects the scheduling of a packet without affecting the packet's relative ordering. You can use the packet loss priority (PLP) bit as part of a congestion control strategy. You can use the loss priority setting to identify packets that have experienced congestion. Typically you mark packets exceeding some service level with a high loss priority. You set loss priority by configuring a classifier or a policer. The loss priority is used later in the workflow to select one of the drop profiles used by RED.

  See "Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows" on page 471 for more information on packet loss priorities.

- Forwarding policy options—These options allow you to associate forwarding classes with next hops. Forwarding policy options also allow you to create classification overrides, which assign forwarding classes to sets of prefixes.

  See "Forwarding Policy Options Overview" on page 304 for more information on forwarding policy options.

- Transmission scheduling and rate control—These parameters provide you with a variety of tools to manage traffic flows:

  - Queuing—After a packet is sent to the outgoing interface on a routing device, it is queued for transmission on the physical media. The amount of time a packet is queued on the device is determined by the availability of the outgoing physical media as well as the amount of traffic using the interface.

  - Schedulers—An individual device interface has multiple queues assigned to store packets. The device determines which queue to service based on a particular method of scheduling. This process often involves a determination of which type of packet should be transmitted before another. Junos schedulers allow you to define the priority, bandwidth, delay buffer size, rate control status, and RED drop profiles to be applied to a particular queue for packet transmission.

See "How Schedulers Define Output Queue Properties" on page 342 for more information on schedulers.

- Policers for traffic classes—*Policers* allow you to limit traffic of a certain class to a specified bandwidth and burst size. Packets exceeding the policer limits can be discarded (hard policing), or can be assigned to a different forwarding class, a different loss priority, or both (soft policing). You define policers with filters that can be associated with input or output interfaces.

  See *Controlling Network Access Using Traffic Policing Overview* for more information on policers.

- Rewrite rules—A *rewrite rule* sets the appropriate CoS bits in the outgoing packet. This allows the next downstream device to classify the packet into the appropriate service group. Rewriting, or marking, outbound packets is useful when the routing device is at the border of a network and must alter the CoS values to meet the policies of the targeted peer.

  Typically, rewrites of the DSCPs on outgoing packets are done once, when packets enter the DiffServ portion of the network, either because the packets do not arrive from the customer with the proper DSCP bit set or because the service provider wants to verify that the customer has set the DSCP properly. CoS schemes that accept the DSCP and classify and schedule traffic solely on DSCP value perform behavior aggregate (BA) DiffServ functions and do not usually rewrite the DSCP. DSCP rewrites typically occur in multifield (MF) DiffServ scenarios.

  See "Rewriting Packet Headers to Ensure Forwarding Behavior" on page 558 for more information on rewrite rules.

RELATED DOCUMENTATION

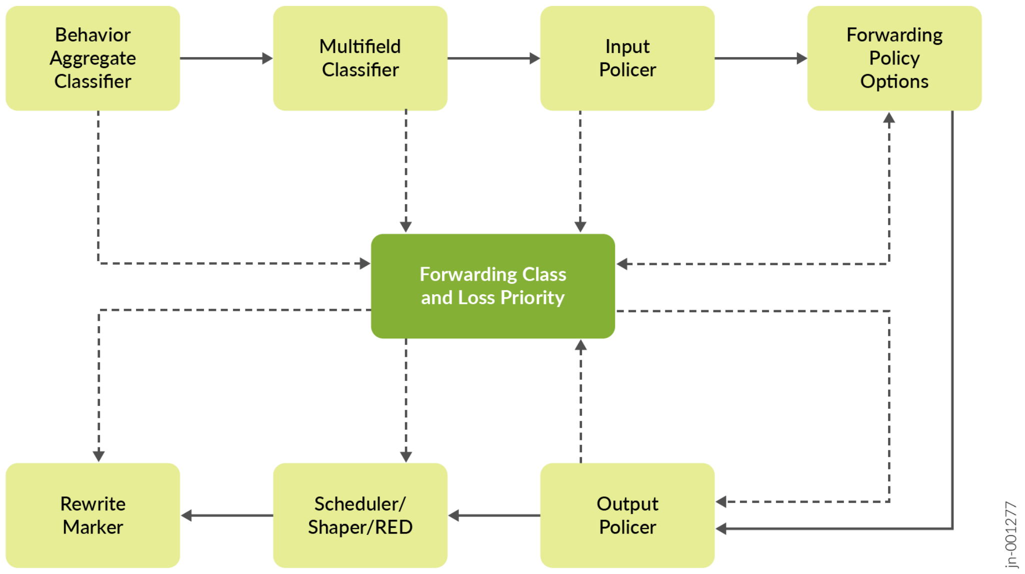
How Class of Service Manages Congestion and Controls Service Levels in the Network | **2**

Mapping CoS Component Inputs to Outputs

Some CoS components map one set of values to another set of values. Each mapping contains one or more inputs and one or more outputs.

Figure 3 on page 10 shows the components of the Junos CoS features, illustrating the sequence in which they interact.

Figure 3: Packet Flow Through CoS-Configurable Components



TIP: Component mapping enables you to define forwarding classes and packet loss priorities for various traffic flows and then map those forwarding classes to output queues with specific shaping and scheduling characteristics.

When you configure a mapping, you set the outputs for a given set of inputs, as shown in Table 1 on page 10.

Table 1: CoS Mappings—Inputs and Outputs

CoS Mappings	Inputs	Outputs	Comments
`classifiers`	`code-points`	`forwarding-class` `loss-priority`	The map sets the forwarding class and PLP for a specific set of code points.
`drop-profile-map`	`loss-priority` `protocol`	`drop-profile`	The map sets the drop profile for a specific PLP and protocol type.
`scheduler-maps`	`forwarding-class`	`scheduler`	This map assigns a forwarding class to a specific scheduler.

Table 1: CoS Mappings—Inputs and Outputs *(Continued)*

CoS Mappings	Inputs	Outputs	Comments
`rewrite-rules`	`forwarding-class` `loss-priority`	`code-points`	The map sets the code points for a specific forwarding class and PLP.

Following are sample configurations for classifiers, drop-profile maps, scheduler maps, and rewrite rules.

In the following classifier sample configuration, packets with EXP bits `000` are assigned to the `data-queue` forwarding class with a `low` loss priority, and packets with EXP bits `001` are assigned to the `data-queue` forwarding class with a `high` loss priority.

```
[edit class-of-service]
classifiers {
    exp exp_classifier {
        forwarding-class data-queue {
            loss-priority low code-points 000;
            loss-priority high code-points 001;
        }
    }
}
```

See "Configuring Behavior Aggregate Classifiers" on page 80 for more information on setting the forwarding class and loss priority for a specific set of code-point aliases and bit patterns

In the following drop-profile map sample configuration, the scheduler includes two drop-profile maps, which specify that packets are evaluated by the `low-drop` drop profile if they have a `low` loss priority and are from any protocol. Packets are evaluated by the `high-drop` drop profile if they have a `high` loss priority and are from any protocol.

```
[edit class-of-service]
schedulers {
    best-effort {
        drop-profile-map loss-priority low protocol any drop-profile low-drop;
        drop-profile-map loss-priority high protocol any drop-profile high-drop;
    }
}
```

See "Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers" on page 469 for more information on mapping drop profiles to a scheduler.

In the following scheduler maps configuration sample, each of the default forwarding classes is mapped to a scheduler specifically designed for that forwarding class.

```
scheduler-maps {
    basic {
        forwarding-class best-effort scheduler be;
        forwarding-class assured-forwarding scheduler af;
        forwarding-class expedited-forwarding scheduler ef;
        forwarding-class network-control scheduler nc;
    }
}
```

See "Configuring Scheduler Maps" on page 348 for more information on mapping forwarding classes to schedulers.

In the following rewrite rule configuration sample, packets in the `be` forwarding class with `low` loss priority are assigned the EXP bits `000`, and packets in the `be` forwarding class with `high` loss priority are assigned the EXP bits `001`.

```
[edit class-of-service]
rewrite-rules {
    exp exp-rw {
        forwarding-class be {
            loss-priority low code-point 000;
            loss-priority high code-point 001;
        }
    }
}
```

See "Configuring Rewrite Rules" on page 564 for more information on setting the code-point aliases and bit patterns for specific forwarding classes and loss priorities as packets leave the device.

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | **60**

Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers | **469**

Configuring Scheduler Maps | **348**

Default Junos CoS Settings

If you do not configure any CoS settings on your router, the software performs some CoS functions to ensure that user traffic and protocol packets are forwarded with minimum delay when the network is experiencing congestion. Some default mappings are automatically applied to each *logical interface* that you configure. Other default mappings, such as explicit default classifiers and *rewrite rules*, are in operation only if you explicitly associate them with an interface.

You can display default CoS settings by issuing the `show class-of-service` *operational mode command*. This section includes sample output displaying the default CoS settings. The sample output is truncated for brevity.

show class-of-service

```
user@host> show class-of-service
```

NOTE: Some platforms require an argument after the `show class-of-service` command. The argument is to select a portion of the following output to display.

Default Forwarding Classes

```
Forwarding class           Queue
  best-effort                    0
  expedited-forwarding           1
  assured-forwarding             2
  network-control                3
```

Default Code-Point Aliases

```
Code point type: dscp
  Alias              Bit pattern
  af11               001010
  af12               001100
...
Code point type: dscp-ipv6
...
```

```
Code point type: exp
...
Code point type: ieee-802.1
...
Code point type: inet-precedence
...
Code point type: ieee-802.1ad
...
```

Default Classifiers

```

Classifier: dscp-default, Code point type: dscp, Index: 7
...

Classifier: dscp-ipv6-default, Code point type: dscp-ipv6, Index: 8
...

Classifier: exp-default, Code point type: exp, Index: 9
...

Classifier: ieee8021p-default, Code point type: ieee-802.1, Index: 10
...

Classifier: ipprec-default, Code point type: inet-precedence, Index: 11
...

Classifier: ipprec-compatibility, Code point type: inet-precedence, Index: 12
...

Classifier: ieee8021ad-default, Code point type: ieee-802.1ad, Index: 41
...
```

Default Rewrite Rules

```

Rewrite rule: dscp-default, Code point type: dscp, Index: 24
  Forwarding class                    Loss priority       Code point
  best-effort                         low                 000000
  best-effort                         high                000000
...

Rewrite rule: dscp-ipv6-default, Code point type: dscp-ipv6, Index: 25
```

```
...

Rewrite rule: exp-default, Code point type: exp, Index: 26
...

Rewrite rule: ieee8021p-default, Code point type: ieee-802.1, Index: 27
...

Rewrite rule: ipprec-default, Code point type: inet-precedence, Index: 28
...

Rewrite rule: ieee8021ad-default, Code point type: ieee-802.1ad, Index: 42
...
```

Default Drop Profile

```
Drop profile: <default-drop-profile>, Type: discrete, Index: 1
  Fill level    Drop probability
         100                 100
```

Default Schedulers

```
Scheduler map: <default>, Index: 2

  Scheduler: <default-be>, Forwarding class: best-effort, Index: 17
    Transmit rate: 95 percent, Rate Limit: none, Buffer size: 95 percent, Priority: low
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             Any             1    <default-drop-profile>
      High            Any             1    <default-drop-profile>
...
```

RELATED DOCUMENTATION

Default Forwarding Classes | 288

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 60

Default Schedulers Overview | 346

Forwarding Classes and Fabric Priority Queues | 319

Packet Flow Through the Junos CoS Process

IN THIS SECTION

Perhaps the best way to understand Junos CoS is to examine how a packet is treated on its way through the CoS process. This topic includes a description of each step and figures illustrating the process.

The following steps describe the CoS process:

1. A *logical interface* has one or more classifiers of different types applied to it (at the `[edit class-of-service interfaces]` hierarchy level). The types of classifiers are based on which part of the incoming packet the classifier examines (for example, EXP bits, IEEE 802.1p bits, or DSCP bits). You can use a translation table to rewrite the values of these bits on ingress.

2. The classifier assigns the packet to a forwarding class and a loss priority (at the `[edit class-of-service classifiers]` hierarchy level).

3. Each forwarding class is assigned to a queue (at the `[edit class-of-service forwarding-classes]` hierarchy level).

4. Input (and output) policers meter traffic and might change the forwarding class and loss priority if a traffic flow exceeds its service level.

5. The physical or logical interface has a scheduler map applied to it (at the `[edit class-of-service interfaces]` hierarchy level).

   At the `[edit class-of-service interfaces]` hierarchy level, the `scheduler-map` and `rewrite-rules` statements affect the outgoing packets, and the `classifiers` statement affects the incoming packets.

6. The scheduler defines how traffic is treated in the output queue—for example, the transmit rate, buffer size, priority, and drop profile (at the `[edit class-of-service schedulers]` hierarchy level).

7. The scheduler map assigns a scheduler to each forwarding class (at the `[edit class-of-service scheduler-maps]` hierarchy level).

8. The drop-profile defines how aggressively to drop packets that are using a particular scheduler (at the `[edit class-of-service drop-profiles]` hierarchy level).

9. The rewrite rule takes effect as the packet leaves a logical interface that has a rewrite rule configured (at the `[edit class-of-service rewrite-rules]` hierarchy level). The rewrite rule writes information to the

packet (for example, EXP or DSCP bits) according to the forwarding class and loss priority of the packet.

Figure 4 on page 18 and Figure 5 on page 19 show the components of the Junos CoS features, illustrating the sequence in which they interact.

Figure 4: CoS Classifier, Queues, and Scheduler

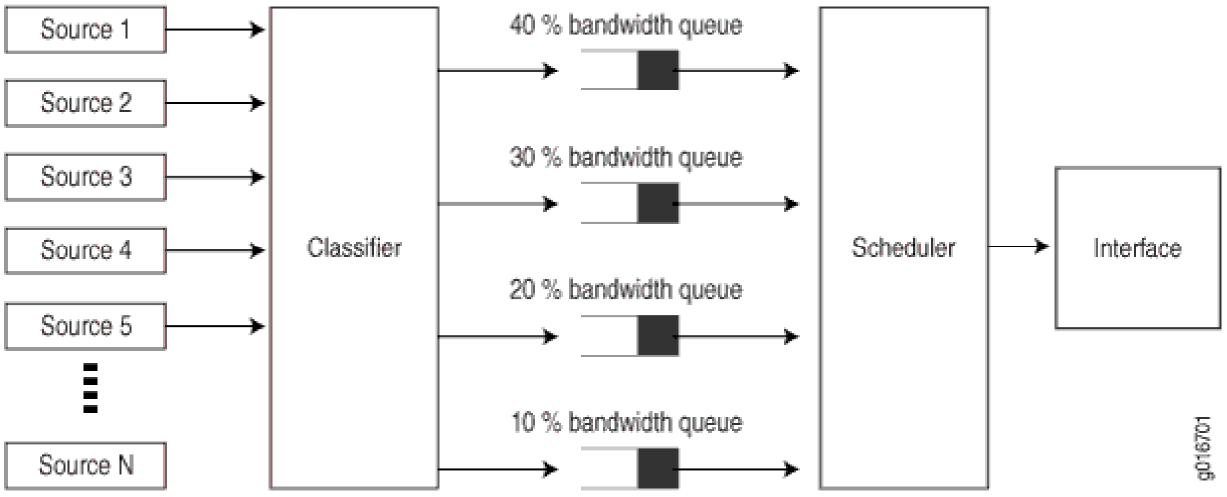
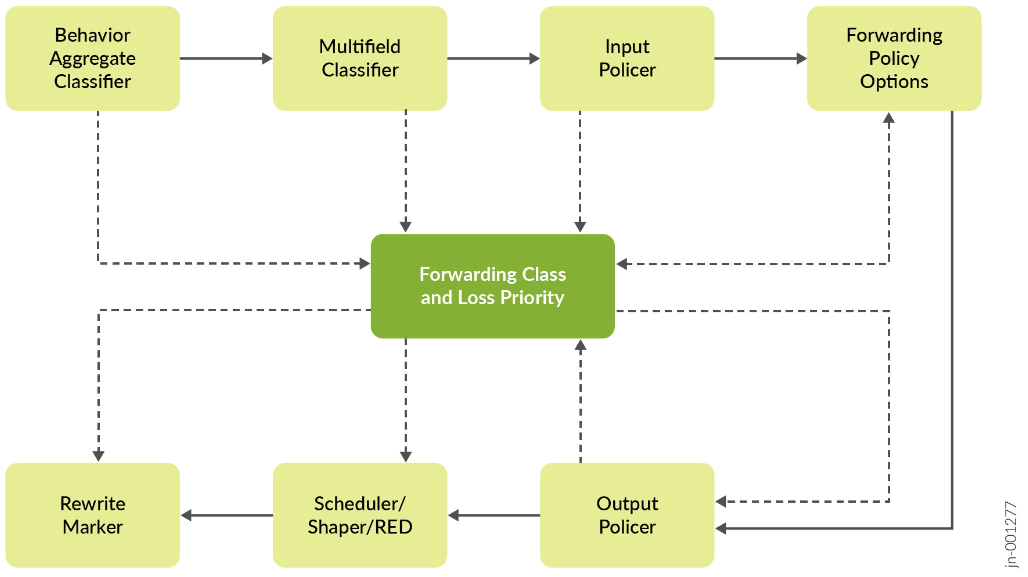


Figure 5: Packet Flow Through CoS- Configurable Components



Each outer box in Figure 5 on page 19 represents a process component. The components in the upper row apply to inbound packets, and the components in the lower row apply to outbound packets. The arrows with the solid lines point in the direction of packet flow.

The middle box (forwarding class and loss priority) represents two data values that can either be inputs to or outputs of the process components. The arrows with the dotted lines indicate inputs and outputs (or settings and actions based on settings). For example, the multifield classifier sets the forwarding class and loss priority of incoming packets. This means that the forwarding class and loss priority are outputs of the classifier; thus, the arrow points away from the classifier. The scheduler receives the forwarding class and loss priority settings, and queues the outgoing packet based on those settings. This means that the forwarding class and loss priority are inputs to the scheduler; thus, the arrow points to the scheduler.

Typically, only a combination of some components (not all) is used to define a CoS service offering.

Packet Flow Within Routers Overview

Although the architecture of Juniper Networks devices differ in detail, the overall flow of a packet within each device remains consistent.

When a packet enters a Juniper Networks device, the PIC or other interface type receiving the packet retrieves it from the network and verifies that the link-layer information is valid. The packet is then passed to the concentrator device such as a Flexible PIC Concentrator (FPC), where the data link and network layer information is verified. In addition, the FPC is responsible for segmenting the packet into

64-byte units called J-cells. These cells are then written into packet storage memory while a notification cell is sent to the route lookup engine. The destination address listed in the notification cell is located in the forwarding table, and the next hop of the packet is written into the result cell. This result cell is queued on the appropriate outbound FPC until the outgoing interface is ready to transmit the packet. The FPC then reads the J-cells out of memory, re-forms the original packet, and sends the packet to the outgoing PIC, where it is transmitted back into the network.

RELATED DOCUMENTATION

Configuring Basic Packet Flow Through the Junos CoS Process | **20**

Configuring Basic Packet Flow Through the Junos CoS Process

IN THIS SECTION

Figure 6 on page 21 and Figure 7 on page 22 show the components of the Junos CoS features, illustrating the sequence in which they interact.

Figure 6: CoS Classifier, Queues, and Scheduler

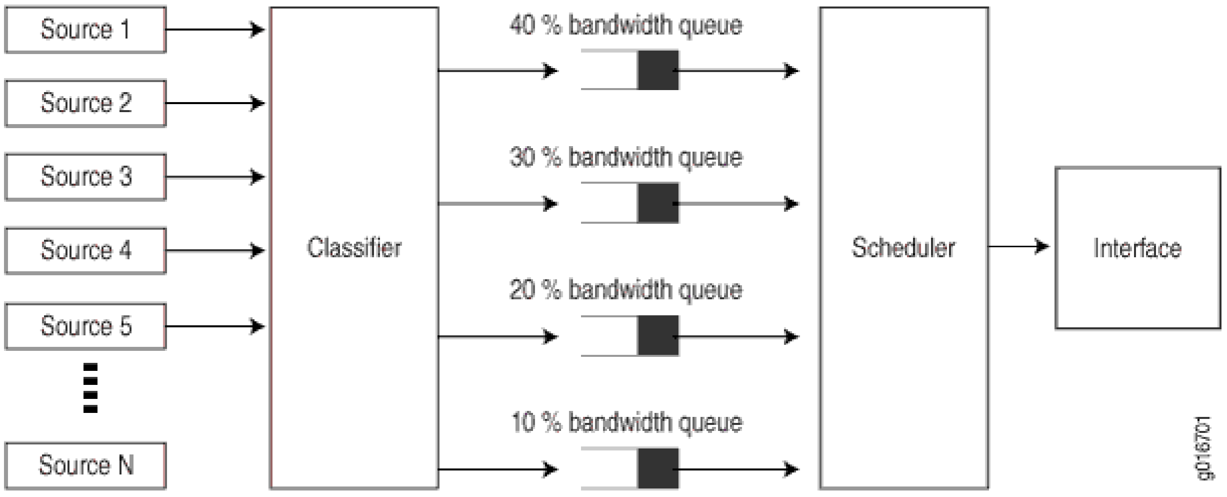
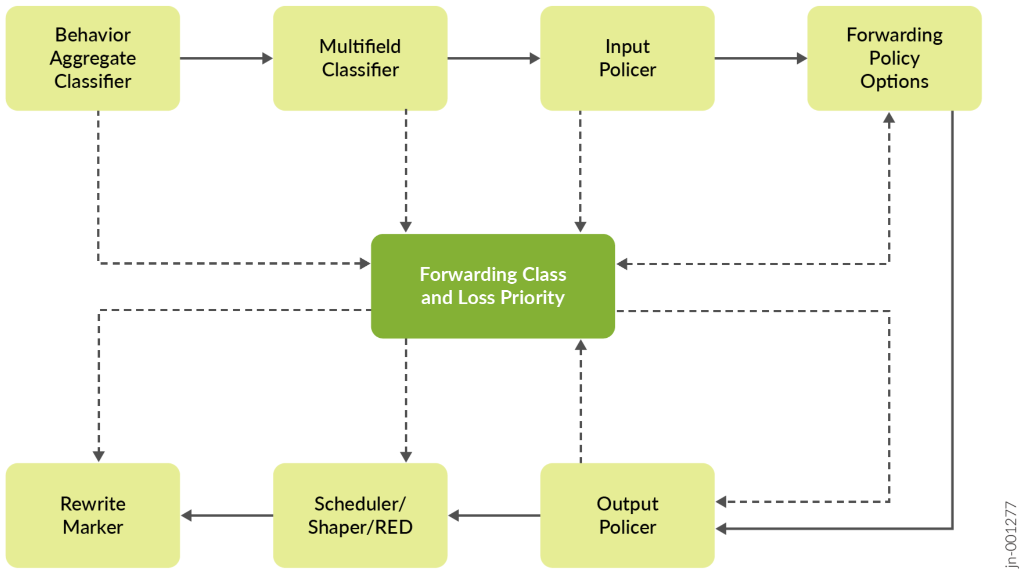


Figure 7: Packet Flow Through CoS- Configurable Components



The following configuration demonstrates the packet flow through the CoS process:

Define Classifiers

If you trust the CoS values in the packet headers, you can use behavior aggregate classification to map those values to a forwarding class and drop priority. For example:

```
[edit class-of-service]
classifiers {
exp exp_classifier {
    forwarding-class data-queue {
        loss-priority high code-points 001;
        loss-priority low code-points 000;
    }
    forwarding-class nc-queue {
        loss-priority high code-points 111;
        loss-priority low code-points 110;
    }
    forwarding-class video-queue {
        loss-priority high code-points 011;
        loss-priority low code-points 010;
    }
    forwarding-class voice-queue {
        loss-priority high code-points 101;
        loss-priority low code-points 100;
    }
}
```

If you do not trust the CoS values in the packet headers, you can use the more complex multifield classification to map ingress traffic to a forwarding class and drop priority. For example:

```
[edit firewall]
family inet {
    filter classify {
        term sip {
            from {
                protocol [ udp tcp ];
                port 5060;
            }
            then {
                forwarding-class nc-queue;
                loss-priority low;
                accept;
            }
        }
    }
}
```

SEE ALSO

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 60

Assign CoS Levels to Packets Based on Multiple Packet Header Fields | 148

Apply Classifiers to Incoming Packets on Interfaces

You apply behavior aggregate classifiers to logical interfaces at the `[edit class-of-service interfaces]` hierarchy level. For example:

```
[edit class-of-service]
interfaces {
    so-* {
        unit 0 {
            classifiers {
                exp exp_classifier;
            }
        }
    }
    t3-* {
        unit 0 {
            classifiers {
                exp exp_classifier;
            }
        }
    }
}
```

You apply multifield classifiers as input filters to logical interfaces at the `[edit interfaces]` hierarchy level. For example:

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

Define Policers to Limit Traffic and Control Congestion

If you need to rate-limit a traffic flow, either by discarding excess traffic (hard policing) or reassign excess traffic to a different forwarding class and/or loss priority (soft policing), define a policier and apply the policer to a firewall filter for that traffic flow. For example:

```
[edit firewall]
policer be-lp {
    if-exceeding {
        bandwidth-limit 10m;
        burst-size-limit 62500;
    }
    then loss-priority high;
}
family inet {
    filter be-lp {
        term t1 {
            from {
                protocol tcp;
                port 80;
            }
            then policer be-lp;
            then loss-priority low;
            then accept;
        }
    }
}
```

SEE ALSO

Controlling Network Access Using Traffic Policing Overview

Define Drop Profiles

Use drop profiles to define the drop probabilities across the range of delay-buffer occupancy, supporting the random early detection (RED) process.

```
[edit class-of-service]
drop-profiles {
    be-red {
        fill-level 20 drop-probability 25;
        fill-level 30 drop-probability 50;
        fill-level 40 drop-probability 75;
        fill-level 50 drop-probability 100;
    }
}
```

Assign Each Forwarding Class to a Queue

To provide differentiated services to each forwarding class, assign each forwarding class to it’s own output queue. For example:

```
[edit class-of-service]
forwarding-classes {
    queue 0 data-queue;
    queue 1 video-queue;
    queue 2 voice-queue;
    queue 3 nc-queue;
}
```

SEE ALSO

Understanding How Forwarding Classes Assign Classes to Output Queues | 286

Define Schedulers

Define the scheduler characteristics for each forwarding class. For example:

```
[edit class-of-service]
schedulers {  #
    data-scheduler {
        buffer-size percent 50;
        drop-profile-map loss-priority high protocol any drop-profile be-red;
        priority low;
        transmit-rate percent 50;
    }
    nc-scheduler {
        buffer-size percent 5;
        priority high;
        transmit-rate percent 5;
    }
    video-scheduler {
        buffer-size percent 25;
        priority strict-high;
        transmit-rate percent 25;
    }
    voice-scheduler {
        buffer-size percent 20;
        priority high;
        transmit-rate percent 20;
    }
}
```

SEE ALSO

How Schedulers Define Output Queue Properties | 342

Define Scheduler Maps

Use scheduler maps to map schedulers to forwarding classes. For example:

```
[edit class-of-service]
scheduler-maps {
    sched1 {
        forwarding-class data-queue scheduler data-scheduler;
        forwarding-class nc-queue scheduler nc-scheduler;
        forwarding-class video-queue scheduler video-scheduler;
        forwarding-class voice-queue scheduler voice-scheduler;
    }
}
```

SEE ALSO

Configuring Scheduler Maps | 348

Define CoS Header Rewrite Rules

Use rewrite rules to redefine the CoS bit pattern of outgoing packets. For example:

```
[edit class-of-service]
rewrite-rules {
    inet-precedence inet-rewrite {
        forwarding-class data-queue {
            loss-priority high code-point 001;
            loss-priority low code-point 000;
        }
        forwarding-class nc-queue {
            loss-priority high code-point 111;
            loss-priority low code-point 110;
        }
        forwarding-class video-queue {
            loss-priority high code-point 101;
            loss-priority low code-point 100;
        }
        forwarding-class voice-queue {
            loss-priority high code-point 011;
            loss-priority low code-point 010;
        }
    }
}
```

SEE ALSO

Rewriting Packet Headers to Ensure Forwarding Behavior | 558

Apply Scheduler Maps and Rewrite Rules to Egress Interfaces

```
[edit class-of-service]
interfaces {
    ge-* {
        scheduler-map sched1;
        unit * {
            rewrite-rules {
                inet-precedence inet-rewrite;
            }
        }
    }
}
```

SEE ALSO

Applying Scheduler Maps Overview | 349

Applying Rewrite Rules to Output Logical Interfaces | 577

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
21.2	Starting with Junos OS Release 21.2, Junos OS displays class of service configuration in alphabetical order regardless of configuration order.

RELATED DOCUMENTATION

Packet Flow Through the Junos CoS Process | 16

Example: Classify all Traffic from a Remote Device with Fixed Interface-Based Classification

IN THIS SECTION

This example shows the configuration of fixed classification based on the incoming interface. Fixed classification can be based on the physical interface or a logical interface.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos. The example shown here was tested and verified on SRX Series Firewalls running a supported Junos OS release. The SRX Series Firewalls are configured to run as routers.

TIP: If you are performing tests on SRX Series Firewalls, you might need to configure the devices to run as unsecured routers in your test environment. You do not typically do this in a production environment.

Overview

IN THIS SECTION

A fixed interface classifier is the simplest way to classify all packets from a specific interface to a forwarding class. You typically use this approach on edge routers to classify all traffic from a remote device to a certain forwarding class and queue. A fixed interface classifier simply looks at the ingress interface on which the packet arrives and assigns all traffic received on that interface to a certain service class.

The fixed interface classifier cannot set the locally-meaningful packet-loss-priority, which is used by rewrite rules and drop profiles. The implicit packet-loss-priority is low for all fixed interface classifiers.

A fixed interface classifier is inadequate for scenarios in which interfaces receive traffic that belongs to multiple service classes. However, interface-based classification can be useful when it is combined with other classification processes. Filtering based on the inbound interface can improve the granularity of classification, for example, when combined with filtering based on code point markings. Combining the processes for interface and code point marking classification allows a single code point marking to have different meanings, depending on the interface on which the packet is received. If you want to combine a fixed interface classifier with a code point classifier, this is in effect a multifield classifier.

More Granular Alternative to Fixed Interface Classifier

In Junos, you can combine interface-based classification and code-point classification by using a multifield classifier, as follows:

```
[edit firewall family inet filter MF_CLASSIFIER term 1]
from {
    dscp ef;
    interface ge-0/0/0.0;
}
then forwarding-class Voice;
```

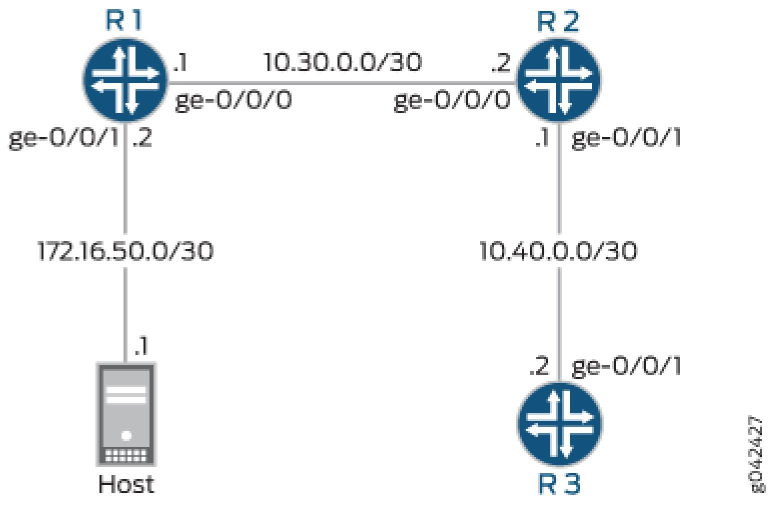
The following Juniper Networks Learning Byte video describes classifiers in more detail.

Video: Class of Service Basics, Part 2: Classification Learning Byte

Topology

Figure 8 on page 31 shows the sample network.

Figure 8: Fixed-Interface Classifier Scenario



To simulate voice traffic, this example shows TCP packets sent from the host to a downstream device. On Device R2, a fixed interface classifier routes the packets into the queue defined for voice traffic.

The classifier is assigned to interface ge-0/0/0 on Device R2. As always, verification of queue assignment is done on the egress interface, which is ge-0/0/1 on Device R2.

"CLI Quick Configuration" on page 32 shows the configuration for all of the Juniper Networks devices in Figure 8 on page 31. The section "Step-by-Step Procedure" on page 33 describes the steps on Device R2.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces ge-0/0/0 description to-R2
set interfaces ge-0/0/0 unit 0 family inet address 10.30.0.1/30
set interfaces ge-0/0/1 description to-host
set interfaces ge-0/0/1 unit 0 family inet address 172.16.50.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.1/32
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

Device R2

```
set interfaces ge-0/0/0 unit 0 family inet address 10.30.0.2/30
set interfaces ge-0/0/1 unit 0 family inet address 10.40.0.1/30
set interfaces lo0 unit 0 family inet address 192.168.0.2/32
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set class-of-service forwarding-classes queue 0 BE-data
set class-of-service forwarding-classes queue 1 Premium-data
set class-of-service forwarding-classes queue 2 Voice
set class-of-service forwarding-classes queue 3 NC
set class-of-service interfaces ge-0/0/0 unit 0 forwarding-class Voice
```

Device R3

```
set interfaces ge-0/0/0 unit 0 family inet address 10.50.0.1/30
set interfaces ge-0/0/1 unit 0 family inet address 10.40.0.2/30
set interfaces lo0 unit 0 family inet address 192.168.0.3/32
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To enable the default DSCP BA classifier:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@R2# set ge-0/0/0 unit 0 family inet address 10.30.0.2/30
   user@R2# set ge-0/0/1 unit 0 family inet address 10.40.0.1/30
   user@R2# set lo0 unit 0 family inet address 192.168.0.2/32
   ```

2. Configure an interior gateway protocol (IGP) or static routes.

   ```
   [edit protocols ospf area 0.0.0.0]
   user@R2# set interface ge-0/0/0.0
   user@R2# set interface ge-0/0/1.0
   user@R2# set interface lo0.0 passive
   ```

3. Configure a set of forwarding classes.

   ```
   [edit class-of-service forwarding-classes]
   user@R2# set queue 0 BE-data
   user@R2# set queue 1 Premium-data
   user@R2# set queue 2 Voice
   user@R2# set queue 3 NC
   ```

4. Map all traffic that arrives on ge-0/0/0.0 into the Voice queue.

   ```
   [edit class-of-service interfaces ge-0/0/0 unit 0]
   user@R2# set forwarding-class Voice
   ```

Results

From configuration mode, confirm your configuration by entering the `show interfaces` and `show class-of-service` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
ge-0/0/0 {
    unit 0 {
        family inet {
            address 10.30.0.2/30;
        }
    }
}
ge-0/0/1 {
    unit 0 {
        family inet {
            address 10.40.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@R2# show protocols
ospf {
    area 0.0.0.0 {
        interface ge-0/0/0.0;
        interface ge-0/0/1.0;
        interface lo0.0 {
            passive;
        }
    }
}
```

```
user@R2# show class-of-service
forwarding-classes {
    queue 0 BE-data;
    queue 1 Premium-data;
    queue 2 Voice;
    queue 3 NC;
}
interfaces {
    ge-0/0/0 {
        unit 0 {
            forwarding-class Voice;
        }
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying a Fixed-Interface Classifier

Purpose

Verify that the fixed interface classifier is enabled on the Device R2's ingress interface. Keep in mind that although the classifier operates on incoming packets, you view the resulting queue assignment on the outgoing (egress) interface.

Action

1. Clear the interface statistics on Device R2's egress interface.

```
user@R2> clear interface statistics ge-0/0/1
```

2. Using a packet generator, send TCP packets to a device that is downstream of Device R2.

   This example uses the packet generator hping.

```
root@host> sudo hping3 10.40.0.2 -c 25 -fast

HPING 10.40.0.2 (eth0 10.40.0.2): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=10.40.0.2 ttl=62 id=8619 sport=0 flags=RA seq=0 win=0 rtt=1.9 ms
len=46 ip=10.40.0.2 ttl=62 id=8620 sport=0 flags=RA seq=1 win=0 rtt=2.8 ms
len=46 ip=10.40.0.2 ttl=62 id=8621 sport=0 flags=RA seq=2 win=0 rtt=1.9 ms
len=46 ip=10.40.0.2 ttl=62 id=8623 sport=0 flags=RA seq=3 win=0 rtt=1.8 ms
len=46 ip=10.40.0.2 ttl=62 id=8624 sport=0 flags=RA seq=4 win=0 rtt=7.1 ms
len=46 ip=10.40.0.2 ttl=62 id=8625 sport=0 flags=RA seq=5 win=0 rtt=1.8 ms
len=46 ip=10.40.0.2 ttl=62 id=8626 sport=0 flags=RA seq=6 win=0 rtt=1.8 ms
len=46 ip=10.40.0.2 ttl=62 id=8627 sport=0 flags=RA seq=7 win=0 rtt=1.9 ms
len=46 ip=10.40.0.2 ttl=62 id=8628 sport=0 flags=RA seq=8 win=0 rtt=2.0 ms
len=46 ip=10.40.0.2 ttl=62 id=8634 sport=0 flags=RA seq=9 win=0 rtt=7.4 ms
len=46 ip=10.40.0.2 ttl=62 id=8635 sport=0 flags=RA seq=10 win=0 rtt=1.8 ms
len=46 ip=10.40.0.2 ttl=62 id=8636 sport=0 flags=RA seq=11 win=0 rtt=2.0 ms
len=46 ip=10.40.0.2 ttl=62 id=8637 sport=0 flags=RA seq=12 win=0 rtt=7.8 ms
len=46 ip=10.40.0.2 ttl=62 id=8639 sport=0 flags=RA seq=13 win=0 rtt=7.0 ms
len=46 ip=10.40.0.2 ttl=62 id=8640 sport=0 flags=RA seq=14 win=0 rtt=1.8 ms
len=46 ip=10.40.0.2 ttl=62 id=8641 sport=0 flags=RA seq=15 win=0 rtt=7.2 ms
len=46 ip=10.40.0.2 ttl=62 id=8642 sport=0 flags=RA seq=16 win=0 rtt=2.1 ms
len=46 ip=10.40.0.2 ttl=62 id=8643 sport=0 flags=RA seq=17 win=0 rtt=2.0 ms
len=46 ip=10.40.0.2 ttl=62 id=8644 sport=0 flags=RA seq=18 win=0 rtt=7.3 ms
len=46 ip=10.40.0.2 ttl=62 id=8645 sport=0 flags=RA seq=19 win=0 rtt=1.7 ms
len=46 ip=10.40.0.2 ttl=62 id=8646 sport=0 flags=RA seq=20 win=0 rtt=7.1 ms
len=46 ip=10.40.0.2 ttl=62 id=8647 sport=0 flags=RA seq=21 win=0 rtt=2.0 ms
len=46 ip=10.40.0.2 ttl=62 id=8648 sport=0 flags=RA seq=22 win=0 rtt=1.7 ms
len=46 ip=10.40.0.2 ttl=62 id=8649 sport=0 flags=RA seq=23 win=0 rtt=1.8 ms
len=46 ip=10.40.0.2 ttl=62 id=8651 sport=0 flags=RA seq=24 win=0 rtt=1.8 ms
```

3. On Device R2, verify that the Voice queue is incrementing.

```
user@R2> show interfaces extensive ge-0/0/1 | find "queue counters"
  Queue counters:       Queued packets  Transmitted packets  Dropped packets
    0 BE-data                        0                    0                0
    1 Premium-data                   0                    0                0
    2 Voice                         25                   25                0
    3 NC                             3                    3                0
  Queue number:         Mapped forwarding classes
    0                   BE-data
    1                   Premium-data
    2                   Voice
    3                   NC
...
```

Meaning

The output shows that the Voice queue has incremented by 25 packets after sending 25 packets through the ge-0/0/0 interface on Device R2.

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 60

Default Aliases for CoS Value Bit Patterns | 71

Example: Configure DiffServ for IPv6

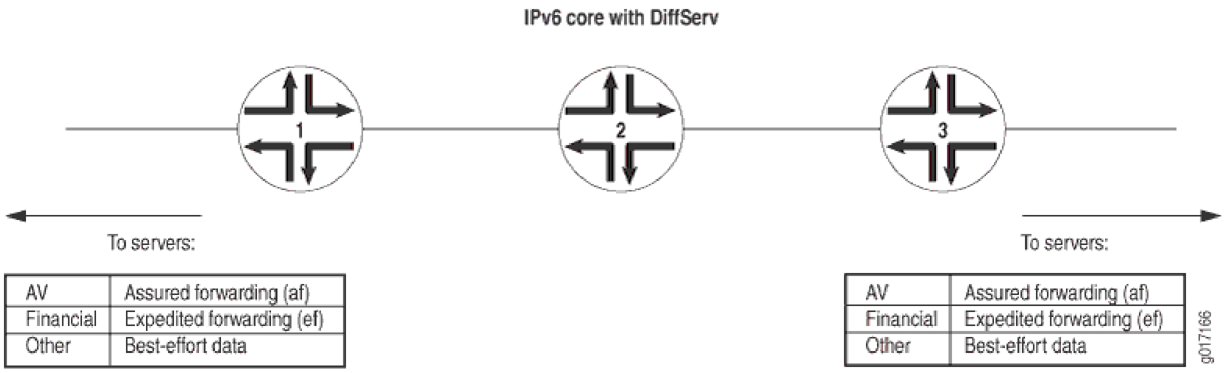
IN THIS SECTION

Configuration

The example assigns expedited forwarding to Q1 and a subset of the assured forwarding classes (**af1*x***) to Q2, and distributes resources among all four forwarding classes.

Figure 9 on page 38 shows the topology of the three routers and links that are used as a case study in this chapter.

Figure 9: Basic IPv6 DiffServ Topology


IPv6 core with DiffServ
1
2
3
To servers:
AV | Assured forwarding (af)
Financial | Expedited forwarding (ef)
Other | Best-effort data
To servers:
AV | Assured forwarding (af)
Financial | Expedited forwarding (ef)
Other | Best-effort data
g017166

In this case study, the service provider has agreed to provide high-priority delivery of packets for two applications between the customer's servers at two sites. The first application generates streams of high-definition audiovisual (television) packet flows and the second generates large quantities of time-sensitive financial information. In all cases, the packet flow is from server to server. The service provider marks the packets appropriately as they enter the network from either site, configures special queues and forwarding classes for this traffic on the three routers, and uses DiffServ for IPv6 for this purpose.

Routers 1 and 3 use multifield (MF) classifiers on the customer-facing interfaces to detect high-priority packets and rewrite the Differentiated Services code points (DSCPs) appropriately. Best-effort data and network control packets are not affected. All three routers are configured with consistent schedulers and resources to handle high-priority packets properly.

Figure 10: IPv6 DiffServ Configuration

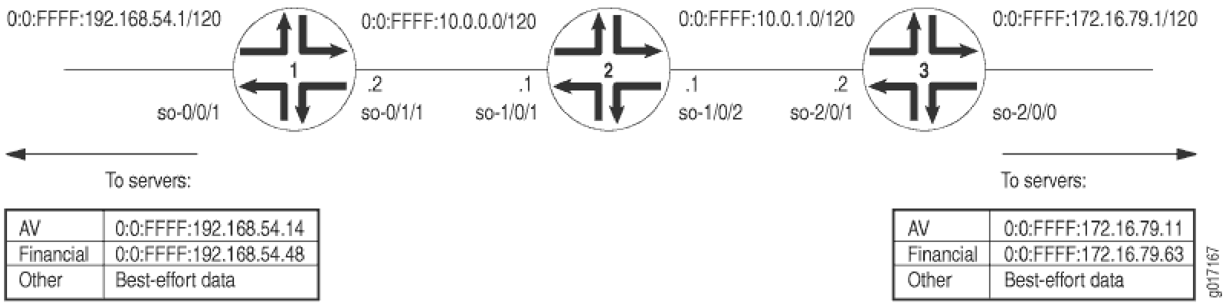


Figure 10 on page 38 shows the complete topology for IPv6 DiffServ, complete with interfaces and IPv6 addresses. The IPv4-mapped IPv6 address format described in RFC 5952 is used.

Begin your configuration on Router 2, the core router. This ensures that when DiffServ is enabled on the edge routers, class of service (CoS) is enabled end to end through the network. The core router configuration is a little simpler because no MF classification is configured in the core.

Router 2

```
[edit]
class-of-service {
    classifiers { # Router 2 classifiers.
        dscp-ipv6 IPv6-classifier {
            import default; # Uses the DSCP default map.
            forwarding-class be-DATA-class {
                loss-priority high code-points 000001;
            }
            forwarding-class ef-FIN-class {
                loss-priority high code-points 101111;
            }
            forwarding-class af-AV-class {
                loss-priority high code-points 001100;
            }
            forwarding-class nc-CONTROL-class {
                loss-priority high code-points 110001;
            }
        }
    }
    drop-profiles {         # Router 2 drop profiles.
        af-AV-normal {
            interpolate {
                fill-level [95 100];
                drop-probability [0 100];
            }
        }
        af-AV-with-PLP {
            interpolate {
                fill-level [60 70 80 90 95];
                drop-probability [80 90 95 97 100];
            }
        }
    }
    forwarding-classes {     # Router 2 forwarding classes.
```

```
        queue 0 be-DATA-class;
        queue 1 ef-FIN-class;
        queue 2 af-AV-class;
        queue 3 nc-CONTROL-class;
    }
    interfaces {           # Router 2 class-of-service interfaces.
        so-1/0/1 { # Connected to R1.
            scheduler-map diffserv-cos-map;
            unit 0 {
                classifiers {
                    dscp-ipv6 IPv6-classifier;
                }
                rewrite-rules {
                    dscp-ipv6 rewrite-IPv6-dscp;
                }
            }
        }
        so-1/0/2 { # Connected to R3.
            scheduler-map diffserv-cos-map;
            unit 0 {
                classifiers {
                    dscp-ipv6 IPv6-classifier;
                }
                rewrite-rules {
                    dscp-ipv6 rewrite-IPv6-dscp;
                }
            }
        }
    }
    rewrite-rules rewrite-IPv6-dscps {     # Router 2 rewrite rules.
        forwarding-class be-DATA-class {
            loss-priority low code points 000000;
            loss-priority high code points 000001;
        }
        forwarding-class ef-FIN-class {
            loss-priority low code points 101110;
            loss-priority high code points 101111;
        }
        forwarding-class af-AV-class {
            loss-priority low code points 001010;
            loss-priority high code points 001100;
        }
        forwarding-class nc-CONTROL-class {
            loss-priority low code points 110000;
            loss-priority high code points 110001;
        }
    }
    scheduler-maps {                 # Router 2 scheduler maps.
                        diffserv-cos-map {
            forwarding-class be-DATA-class scheduler be-DATA-scheduler;
            forwarding-class ef-FIN-class scheduler ef-FIN-scheduler;
            forwarding-class af-AV-class scheduler af-AV-scheduler;
            forwarding-class nc-CONTROL-class scheduler nc-CONTROL-scheduler;
        }
    }
    schedulers {                     # Router 2 schedulers.
                        be-DATA-scheduler {
            transmit-rate percent 40;
            buffer-size percent 40;
            priority low;
        }
                        ef-FIN-scheduler {
            transmit-rate percent 10;
            buffer-size percent 10;
            priority high;
        }
                        af-AV-scheduler {
            transmit-rate percent 45;
            buffer-size percent 45;
            priority high;
            drop-profile-map loss-priority low protocol any drop-profile af-AV-normal;
            drop-profile-map loss-priority high protocol any drop-profile af-AV-with-PLP;
        }
                        nc-CONTROL-scheduler {
            transmit-rate percent 5;
            buffer-size percent 5;
            priority low;
        }
    }
}
interfaces {             # R2 interfaces.
    so-1/0/1 { # Connected to R1.
        unit 0 {
            family inet {
                address 10.0.0.1/24;
            }
            family inet6 {
                address 0:0:FFFF:10.0.0.1/120;
            }
        }
    }
    so-1/0/2 { # Connected to R3.
        unit 0 {
            family inet {
                address 10.0.1.1/24;
            }
            family inet6 {
                address 0:0:FFFF:10.0.1.1/120;
            }
        }
    }
}
```

Continue your configuration on Router 1 and Router 3, the edge routers. These routers get firewall-filter-based MF classifiers and rewrite rules for markers as well as schedulers and drop profiles on the core-facing interfaces.

Router 1

```
[edit]
class-of-service {
    classifiers {            # Router 1 classifiers.
        dscp-ipv6 IPv6-classifier {
            import default; # Uses the DSCP default map.
            forwarding-class be-DATA-class {
                loss-priority high code-points 000001;
            }
            forwarding-class ef-FIN-class {
                loss-priority high code-points 101111;
            }
            forwarding-class af-AV-class {
                loss-priority high code-points 001100;
            }
            forwarding-class nc-CONTROL-class {
                loss-priority high code-points 110001;
            }
        }
    }
```

```
drop-profiles {         # Router 1 drop profiles.
                       af-AV-normal {
        interpolate {
            fill-level [95 100];
            drop-probability [0 100];
        }
    }
                       af-AV-with-PLP {
        interpolate {
            fill-level [60 70 80 90 95];
            drop-probability [80 90 95 97 100];
        }
    }
}
forwarding-classes {     # Router 1 forwarding classes.
    queue 0 be-DATA-class;
    queue 1 ef-FIN-class;
    queue 2 af-AV-class;
    queue 3 nc-CONTROL-class;
}
interfaces {            # Router 1 class-of-service interfaces.
    so-0/1/1 {  # To servers.
        scheduler-map diffserv-cos-map;
        unit 0 {
            classifiers {
                dscp-ipv6 IPv6-classifier;
            }
            rewrite-rules {
                dscp-ipv6 rewrite-IPv6-dscp;
            }
        }
    }
    rewrite-rules rewrite-IPv6-dscps {        # Router 1 rewrite rules.
        forwarding-class be-DATA-class {
            loss-priority low code points 000000;
            loss-priority high code points 000001;
        }
        forwarding-class ef-FIN-class {
            loss-priority low code points 101110;
            loss-priority high code points 101111;
        }
        forwarding-class af-AV-class {
            loss-priority low code points 001010;
```

```
                loss-priority high code points 001100;
            }
            forwarding-class nc-CONTROL-class {
                loss-priority low code points 110000;
                loss-priority high code points 110001;
            }
        }
        scheduler-maps {      # Router 1 scheduler map.
                                 diffserv-cos-map {
                forwarding-class be-DATA-class scheduler be-DATA-scheduler;
                forwarding-class ef-FIN-class scheduler ef-FIN-scheduler;
                forwarding-class af-AV-class scheduler af-AV-scheduler;
                forwarding-class nc-CONTROL-class scheduler nc-CONTROL-scheduler;
            }
        }
        schedulers {          # Router 1 schedulers.
                                 be-DATA-scheduler {
                transmit-rate percent 40;
                buffer-size percent 40;
                priority low;
            }
                                 ef-FIN-scheduler {
                transmit-rate percent 10;
                buffer-size percent 10;
                priority high;
            }
                                 af-AV-scheduler {
                transmit-rate percent 45;
                buffer-size percent 45;
                priority high;
                drop-profile-map loss-priority low protocol any drop-profile af-AV-normal;
                drop-profile-map loss-priority high protocol any drop-profile af-AV-with-PLP;
            }
                                 nc-CONTROL-scheduler {
                transmit-rate percent 5;
                buffer-size percent 5;
                priority low;
            }
        }
    }
    firewall {                   # Router 1 firewall policer and filter.
        policer ef-FIN-Policer-Profile {
            if-exceeding {
```

```
            bandwidth-percent 10;
            burst-size-limit 2k;
        }
        then loss-priority high;
    }
    family inet6 {
        filter mf-classifier {
            filter-specific;
            term AV {
                from {
                    destination-address {
                        0:0:FFFF:172.16.79.11;
                    }
                }
                then {
                    loss-priority low;
                    forwarding-class af-AV-class;
                }
            }
            term Finance {
                from {
                    destination-address {
                        O:0:FFFF:172.16.79.63;
                    }
                }
                then {
                    policer ef-FIN-Policer-Profile;
                    forwarding-class ef-FIN-class;
                }
            }
            term Network-Control {
                from {
                    traffic-class 192; # 192 is the 110000 traffic class.
                }
                then {
                    forwarding-class nc-CONTROL-class; # This is network control traffic.
                }
            }
            term Data {
                then forwarding-class be-DATA-class; # The rest is data.
            }
        }
    }
    }
    interfaces {             # Router 1 interfaces.
        so-0/0/1 {  # To servers.
            unit 0 {
                family inet {
                    address 192.168.54.1/24;
                }
                family inet6 {
                    filter {
                        input mf-classifier;
                    }
                    address 0:0:FFFF:192.168.54.1/120;
                }
            }
        }
        so-0/1/1 {  # Connected to R2.
            unit 0 {
                family inet {
                    address 10.0.0.2/24;
                }
                family inet6 {
                    address 0:0:FFFF:10.0.0.2/120;
                }
            }
        }
    }
}
```

Router 3

```
[edit]
class-of-service {
    classifiers {          # Router 3 classifiers.
        dscp-ipv6 IPv6-classifier {
            import default; # Uses the DSCP default map.
            forwarding-class be-DATA-class {
                loss-priority high code-points 000001;
            }
            forwarding-class ef-FIN-class {
                loss-priority high code-points 101111;
            }
            forwarding-class af-AV-class {
                loss-priority high code-points 001100;
            }
            forwarding-class nc-CONTROL-class {
                loss-priority high code-points 110001;
            }
        }
    }
    drop-profiles {          # Router 3 drop profiles.
                       af-AV-normal {
            interpolate {
                fill-level [95 100];
                drop-probability [0 100];
            }
        }
                       af-AV-with-PLP {
            interpolate {
                fill-level [60 70 80 90 95];
                drop-probability [80 90 95 97 100];
            }
        }
    }
    forwarding-classes {   # Router 3 forwarding classes.
        queue 0 be-DATA-class;
        queue 1 ef-FIN-class;
        queue 2 af-AV-class;
        queue 3 nc-CONTROL-class;
    }
    interfaces {          # Router 3 class-of-service interfaces.
        so-2/0/1 {  # To servers.
            scheduler-map diffserv-cos-map;
            unit 0 {
                classifiers {
                    dscp-ipv6 IPv6-classifier;
                }
                rewrite-rules {
                    dscp-ipv6 rewrite-IPv6-dscp;
                }
            }
        }
        rewrite-rules rewrite-IPv6-dscps {        # Router 3 rewrite rules.
            forwarding-class be-DATA-class {
                loss-priority low code points 000000;
                loss-priority high code points 000001;
            }
            forwarding-class ef-FIN-class {
                loss-priority low code points 101110;
                loss-priority high code points 101111;
            }
            forwarding-class af-AV-class {
                loss-priority low code points 001010;
                loss-priority high code points 001100;
            }
            forwarding-class nc-CONTROL-class {
                loss-priority low code points 110000;
                loss-priority high code points 110001;
            }
        }
        scheduler-maps {                    # Router 3 scheduler map.
                                 diffserv-cos-map {
                forwarding-class be-DATA-class scheduler be-DATA-scheduler;
                forwarding-class ef-FIN-class scheduler ef-FIN-scheduler;
                forwarding-class af-AV-class scheduler af-AV-scheduler;
                forwarding-class nc-CONTROL-class scheduler nc-CONTROL-scheduler;
            }
        }
        schedulers {                        # Router 3 schedulers.
                                 be-DATA-scheduler {
                transmit-rate percent 40;
                buffer-size percent 40;
                priority low;
            }
                                 ef-FIN-scheduler {
                transmit-rate percent 10;
                buffer-size percent 10;
                priority high;
            }
                                 af-AV-scheduler {
                transmit-rate percent 45;
                buffer-size percent 45;
                priority high;
                drop-profile-map loss-priority low protocol any drop-profile af-AV-normal;
                drop-profile-map loss-priority high protocol any drop-profile af-AV-with-PLP;
            }
                                 nc-CONTROL-scheduler {
                transmit-rate percent 5;
                buffer-size percent 5;
                priority low;
            }
        }
        firewall {               # Router 3 firewall policer and filter.
            policer ef-FIN-Policer-Profile {
                if-exceeding {
                    bandwidth-percent 10;
                    burst-size-limit 2k;
                }
                then loss-priority high;
            }
            family inet6 {
                filter mf-classifier {
                    filter-specific;
                    term AV {
                        from {
                            destination-address {
                                0:0:FFFF:172.16.79.11;
                            }
                        }
                        then {
                            loss-priority low;
                            forwarding-class af-AV-class;
                        }
                    }
                    term Finance {
                        from {
                            destination-address {
                                O:0:FFFF:172.16.79.63;
                            }
                        }
                        then {
                            policer ef-FIN-Policer-Profile;
                            forwarding-class ef-FIN-class;
                        }
                    }
                    term Network-Control {
                        from {
                            traffic-class 192; # 192 is the 110000 traffic class.
                        }
                        then {
                            forwarding-class nc-CONTROL-class; # This is network control traffic.
                        }
```

```
                    }
                    term Data {
                        then forwarding-class be-DATA-class; # The rest is data.
                    }
                }
            }
        }
        interfaces {              # Router 3 interfaces.
            so-2/0/0 { # To servers.
                unit 0 {
                    family inet {
                        address 1172.16.79.1/24;
                    }
                    family inet6 {
                        filter {
                            input mf-classifier;
                        }
                        address 0:0:FFFF:172.16.79.1/120;
                    }
                }
            }
            so-2/0/1 { # to R2
                unit 0 {
                    family inet {
                        address 10.0.1.2/24;
                    }
                    family inet6 {
                        address 0:0:FFFF:10.0.1.2/120;
                    }
                }
            }
        }
    }
}
```

Verification

To verify that your CoS using IPv6 DiffServ configuration is correct, use the following commands:

- `show class-of-service classifier type dscp-ipv6`
- `show class-of-service rewrite-rule type dscp-ipv6`

- `show class-of-service interface`
- `show class-of-service forwarding-table classifier mapping`
- `show class-of-service forwarding-table rewrite-rule mapping`
- `show class-of-service scheduler-map` ***scheduler-map-name***
- `show class-of-service forwarding-table scheduler-map`

The following section shows the output of these commands used with the configuration example.

DiffServ Classifiers

```
user@R1> show class-of-service classifier type dscp-ipv6
Classifier: dscp-ipv6-default, Code point type: dscp-ipv6, Index: 4
  Code point         Forwarding class                    Loss priority
  000000             be-DATA-class                       low
  000001             be-DATA-class                       low
  000010             be-DATA-class                       low
  000011             be-DATA-class                       low
  000100             be-DATA-class                       low
  000101             be-DATA-class                       low
  000110             be-DATA-class                       low
  000111             be-DATA-class                       low
  001000             be-DATA-class                       low
  001001             be-DATA-class                       low
  001010             af-AV-class                         low
  001011             be-DATA-class                       low
  001100             af-AV-class                         high
  001101             be-DATA-class                       low
  001110             af-AV-class                         high
  001111             be-DATA-class                       low
  010000             be-DATA-class                       low
  010001             be-DATA-class                       low
  010010             be-DATA-class                       low
  010011             be-DATA-class                       low
  010100             be-DATA-class                       low
  010101             be-DATA-class                       low
  010110             be-DATA-class                       low
  010111             be-DATA-class                       low
  011000             be-DATA-class                       low
  011001             be-DATA-class                       low
  011010             be-DATA-class                       low
  011011            be-DATA-class                     low
  011100            be-DATA-class                     low
  011101            be-DATA-class                     low
  011110            be-DATA-class                     low
  011111            be-DATA-class                     low
  100000            be-DATA-class                     low
  100001            be-DATA-class                     low
  100010            be-DATA-class                     low
  100011            be-DATA-class                     low
  100100            be-DATA-class                     low
  100101            be-DATA-class                     low
  100110            be-DATA-class                     low
  100111            be-DATA-class                     low
  101000            be-DATA-class                     low
  101001            be-DATA-class                     low
  101010            be-DATA-class                     low
  101011            be-DATA-class                     low
  101100            be-DATA-class                     low
  101101            be-DATA-class                     low
  101110            ef-FIN-class                      low
  101111            be-DATA-class                     low
  110000            nc-CONTROL-class                  low
  110001            be-DATA-class                     low
  110010            be-DATA-class                     low
  110011            be-DATA-class                     low
  110100            be-DATA-class                     low
  110101            be-DATA-class                     low
  110110            be-DATA-class                     low
  110111            be-DATA-class                     low
  111000            nc-CONTROL-class                  low
  111001            be-DATA-class                     low
  111010            be-DATA-class                     low
  111011            be-DATA-class                     low
  111100            be-DATA-class                     low
  111101            be-DATA-class                     low
  111110            be-DATA-class                     low
  111111            be-DATA-class                     low
Classifier: IPv6-classifier, Code point type: dscp-ipv6, Index: 18301
  Code point        Forwarding class                  Loss priority
  000000            be-DATA-class                     low
  000001            be-DATA-class                     high
  000010            be-DATA-class                     low
  000011            be-DATA-class                     low
```

```
000100             be-DATA-class                       low
000101             be-DATA-class                       low
000110             be-DATA-class                       low
000111             be-DATA-class                       low
001000             be-DATA-class                       low
001001             be-DATA-class                       low
001010             af-AV-class                         low
001011             be-DATA-class                       low
001100             af-AV-class                         high
001101             be-DATA-class                       low
001110             af-AV-class                         high
001111             be-DATA-class                       low
010000             be-DATA-class                       low
010001             be-DATA-class                       low
010010             be-DATA-class                       low
010011             be-DATA-class                       low
010100             be-DATA-class                       low
010101             be-DATA-class                       low
010110             be-DATA-class                       low
010111             be-DATA-class                       low
011000             be-DATA-class                       low
011001             be-DATA-class                       low
011010             be-DATA-class                       low
011011             be-DATA-class                       low
011100             be-DATA-class                       low
011101             be-DATA-class                       low
011110             be-DATA-class                       low
011111             be-DATA-class                       low
100000             be-DATA-class                       low
100001             be-DATA-class                       low
100010             be-DATA-class                       low
100011             be-DATA-class                       low
100100             be-DATA-class                       low
100101             be-DATA-class                       low
100110             be-DATA-class                       low
100111             be-DATA-class                       low
101000             be-DATA-class                       low
101001             be-DATA-class                       low
101010             be-DATA-class                       low
101011             be-DATA-class                       low
101100             be-DATA-class                       low
101101             be-DATA-class                       low
101110             ef-FIN-class                        low
```

```
  101111            ef-FIN-class                        high
  110000            nc-CONTROL-class                    low
  110001            nc-CONTROL-class                    high
  110010            be-DATA-class                       low
  110011            be-DATA-class                       low
  110100            be-DATA-class                       low
  110101            be-DATA-class                       low
  110110            be-DATA-class                       low
  110111            be-DATA-class                       low
  111000            nc-CONTROL-class                    low
  111001            be-DATA-class                       low
  111010            be-DATA-class                       low
  111011            be-DATA-class                       low
  111100            be-DATA-class                       low
  111101            be-DATA-class                       low
  111110            be-DATA-class                       low
  111111            be-DATA-class                       low
```

Rewrite Rules

```
user@R1> show class-of-service rewrite-rule type dscp-ipv6
Rewrite rule: dscp-ipv6-default, Code point type: dscp-ipv6, Index: 20
  Forwarding class                    Loss priority       Code point
  be-DATA-class                       low                 000000
  be-DATA-class                       high                000000
  ef-FIN-class                        low                 101110
  ef-FIN-class                        high                101110
  af-AV-class                         low                 001010
  af-AV-class                         high                001100
  nc-CONTROL-class                    low                 110000
  nc-CONTROL-class                    high                111000
Rewrite rule: rewrite-IPv6-dscp, Code point type: dscp-ipv6, Index: 58077
  Forwarding class                    Loss priority       Code point
  be-DATA-class                       low                 000000
  be-DATA-class                       high                000001
  ef-FIN-class                        low                 101110
  ef-FIN-class                        high                101111
  af-AV-class                         low                 001010
  af-AV-class                         high                001100
  nc-CONTROL-class                    low                 110000
  nc-CONTROL-class                    high                110001
```

Class-of-Service Interfaces

```
user@R1> show class-of-service interface
...
Physical interface: so-0/0/1, Index: 141
Queues supported: 4, Queues in use: 4
  Scheduler map: diffserv-cos-map, Index: -543019056
  Logical interface: so-0/0/1.0, Index: 68
    Object        Name                   Type                       Index
    Rewrite       rewrite-IPv6-dscp      dscp-ipv6                  58077
    Rewrite       exp-default            exp                           21
    Classifier    IPv6-classifier        dscp-ipv6                  18301
    Classifier    exp-default            exp                            5
...
Physical interface: so-0/1/1, Index: 144
Queues supported: 4, Queues in use: 4
  Scheduler map: <default>, Index: -113795564

  Logical interface: so-0/1/1.0, Index: 69
    Object        Name                   Type                       Index
    Rewrite       exp-default            exp                           21
    Classifier    exp-default            exp                            5
    Classifier    ipprec-compatibility   ip                             8
```

Classifier Mapping

```
user@R1> show class-of-service forwarding-table classifier mapping
                      Table Index/
Interface      Index      Q num      Table type
so-0/0/1.0        68      18301      IPv6 DSCP
so-0/1/1.0        69          8      IPv4 precedence
```

Rewrite Rule Mapping

```
user@R1> show class-of-service forwarding-table rewrite-rule mapping
Interface      Index   Table index  Type
so-0/1/1.0        68      58077     IPv6 DSCP
```

Scheduler Map

```
user@R1> show class-of-service scheduler-map diffserv-cos-map
Scheduler map: diffserv-cos-map, Index: 1094596010
  Scheduler: be-DATA-scheduler, Forwarding class: be-DATA-class, Index: 14343
    Transmit rate: 40 percent, Rate Limit: none, Buffer size: 40 percent,
    Priority: low
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             non-TCP         1    <default-drop-profile>
      Low             TCP             1    <default-drop-profile>
      High            non-TCP         1    <default-drop-profile>
      High            TCP             1    <default-drop-profile>
  Scheduler: ef-FIN-scheduler, Forwarding class: ef-FIN-class, Index: 21707
    Transmit rate: 10 percent, Rate Limit: none, Buffer size: 10 percent,
    Priority: high
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             non-TCP         1    <default-drop-profile>
      Low             TCP             1    <default-drop-profile>
      High            non-TCP         1    <default-drop-profile>
      High            TCP             1    <default-drop-profile>
  Scheduler: af-AV-scheduler, Forwarding class: af-AV-class, Index: 51704
    Transmit rate: 45 percent, Rate Limit: none, Buffer size: 45 percent,
    Priority: high
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             non-TCP     61474    af-AV-normal
      Low             TCP         61474    af-AV-normal
      High            non-TCP     65199    af-AV-with-PLP
      High            TCP         65199    af-AV-with-PLP
  Scheduler: nc-CONTROL-scheduler, Forwarding class: nc-CONTROL-class, Index: 50404
    Transmit rate: 5 percent, Rate Limit: none, Buffer size: 5 percent,
    Priority: low
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             non-TCP         1    <default-drop-profile>
      Low             TCP             1    <default-drop-profile>
      High            non-TCP         1    <default-drop-profile>
      High            TCP             1    <default-drop-profile>

user@R1>  show class-of-service forwarding-table scheduler-map
```

```
...
Interface: so-0/0/1 (Index: 141, Map index: -543019056, Map type: FINAL,
Num of queues: 4):
  Entry 0 (Scheduler index: 14343, Queue #: 0):
    Tx rate: 0 Kb (40%), Buffer size: 40 percent
Priority low
    PLP high: 1, PLP low: 1, TCP PLP high: 1, TCP PLP low: 1
  Entry 1 (Scheduler index: 21707, Queue #: 1):
    Tx rate: 0 Kb (10%), Buffer size: 10 percent
Priority high
    PLP high: 1, PLP low: 1, TCP PLP high: 1, TCP PLP low: 1
  Entry 2 (Scheduler index: 51704, Queue #: 2):
    Tx rate: 0 Kb (45%), Buffer size: 45 percent
Priority high
    PLP high: 65199, PLP low: 61474, TCP PLP high: 65199, TCP PLP low: 61474
  Entry 3 (Scheduler index: 50404, Queue #: 3):
    Tx rate: 0 Kb (5%), Buffer size: 5 percent
Priority low
    PLP high: 1, PLP low: 1, TCP PLP high: 1, TCP PLP low: 1
...
```

2

PART

Configuring Class of Service

CHAPTER 2

Assign Service Levels with Behavior Aggregate Classifiers

IN THIS CHAPTER

How Behavior Aggregate Classifiers Prioritize Trusted Traffic

The idea behind class of service (CoS) is that packets are not treated identically by the routers or switches on the network. In order to selectively apply service classes to specific packets, the packets of interest must be classified in some fashion.

The simplest way to classify a packet is to use behavior aggregate (BA) classification, also called the *CoS value* in this document. The DSCP, DSCP IPv6, or IP precedence bits of the IP header convey the behavior aggregate class information. The information might also be found in the MPLS EXP bits, IEEE 802.1ad, or IEEE 802.1p CoS bits.

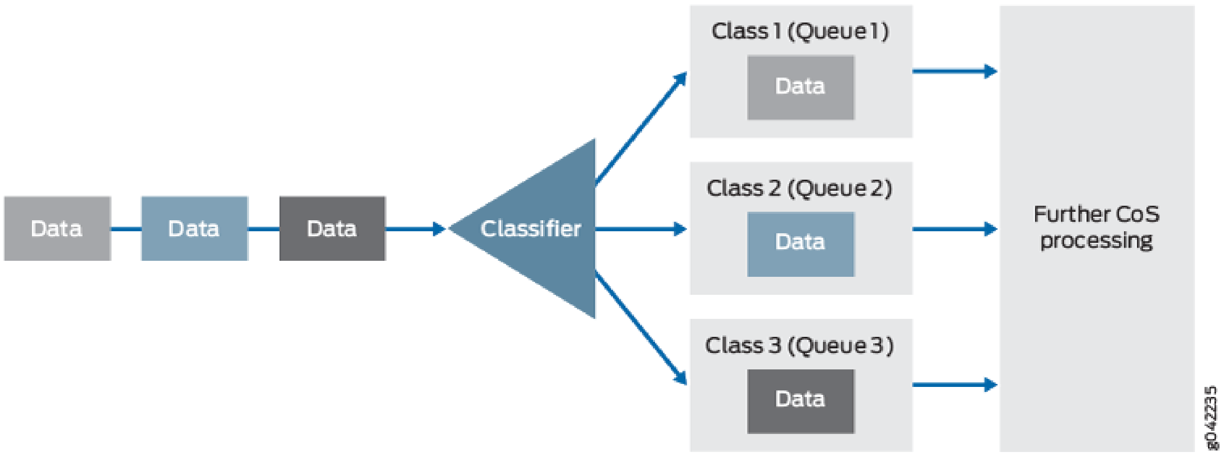
NOTE: When you have both a class of service (CoS) and firewall filter, and both include DSCP or forwarding class filter actions, the criteria in the firewall filter takes precedence over the CoS settings. The same is true when creating new configurations; that is, where the same settings exist, the firewall filter takes precedence over the CoS, regardless of which was created first.

BA classification is useful if the traffic comes from a trusted source and the CoS value in the packet header is trusted. If the traffic is untrusted, multifield classifiers (see "Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields" on page 148) are used to classify packets based on multiple packet fields. It is common to use multifield classifiers to classify traffic at the ingress of a network, rewrite the packet headers (see "Rewriting Packet Headers to Ensure Forwarding Behavior" on page 558), then use the more efficient BA classification for traversing the network.

The BA classifier maps a CoS value in the packet header 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 detection (RED) algorithm to control packet discard during periods of congestion.

Figure 11 on page 61 provides a high-level illustration of how a classifier works.

Figure 11: How a Classifier Works



The types of BA classifiers are based on which part of the incoming packet the classifier examines:

- DSCP, DSCP IPv6, or IP precedence—IP packet classification (L3 headers)
- MPLS EXP—MPLS packet classification (L2 headers)
- IEEE 802.1p—Packet classification (L2 headers)
- IEEE 802.1ad—Packet classification for IEEE 802.1ad formats (including DEI bit)

Unlike multifield classifiers (which are discussed in "Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields" on page 148), BA classifiers are based on fixed-length fields. This makes BA classifiers computationally more efficient than multifield classifiers. For this reason, core devices are normally configured to perform BA classification, because of the higher traffic volumes core devices handle.

In most cases, you need to rewrite a given marker (IP precedence, DSCP, IEEE 802.1p, IEEE 802.1ad, or MPLS EXP settings) at the egress node to accommodate BA classification by core and egress devices. For more information about rewrite markers, see "Rewriting Packet Headers to Ensure Forwarding Behavior" on page 558.

NOTE: If you apply an IEEE 802.1 classifier to a logical interface, this classifier takes precedence over any other classifier type. Classifiers for IEEE 802.1, IP (DSCP or IP precedence), and MPLS (EXP) can coexist on a logical interface.

If you carry more classes of traffic than the device can forward independently, you must configure the additional classes to be aggregated into one of the available classes. You use the BA classifier to configure class aggregation.

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

If you apply both a protocol-specific BA classifier and a protocol-independent firewall filter together, the protocol-independent filter is processed before the protocol-specific BA classifier, regardless or protocol family. `firewall family any filter` is protocol independent and will be always processed before protocol-specific BA classifiers.

Fixed classification is protocol independent as well, hence, it is executed before any firewall filter.

For more information about multifield classifiers, see "Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields" on page 148. For more information about protocol-independent filters, see *Guidelines for Configuring Firewall Filters*. For more information about fixed classification, see "Applying Forwarding Classes to Interfaces" on page 315.

If you do nothing to configure or assign classifiers, Junos automatically assigns a default IP precedence classifier to all logical interfaces. The default IP precedence classifier maps IP precedence code points to `best-effort` and `network-control` forwarding classes (mapped to queue 0 and queue 3 on routing devices, respectively). The default Junos CoS policy reserves 5 percent of available bandwidth for `network-control` traffic and 95 percent for `best-effort` traffic. Junos provides a range of default BA classifiers that you can apply to logical interfaces and that map various CoS values to `assured-forwarding`, `expedited-forwarding`, `best-effort`, and `network-control` forwarding classes. You can also define custom BA classifiers that map any CoS value to any classifier you define.

NOTE: The default Junos CoS policy assigning 95 percent of the bandwidth for queue 0 and 5 percent for queue 3 on routing devices is in effect regardless of any custom BA classifier or forwarding class definitions until you configure a custom scheduler. See "Default Schedulers Overview" on page 346 and "Configuring Schedulers" on page 348.

If you enable the MPLS protocol family on a *logical interface*, a default MPLS EXP classifier is automatically applied to that logical interface. This default EXP classifier maps the eight possible EXP code point values into a combination of the four default forwarding classes and loss priority values to be directly compatible with the default EXP rewrite rule. See "Default MPLS EXP Classifier" on page 66 and "Rewriting MPLS and IPv4 Packet Headers" on page 579.

Other default classifiers (such as those for IEEE 802.1p bits and DSCP) require that you explicitly associate a default classification table with a logical interface. When you explicitly associate a default classifier with a logical interface, you are in effect overriding the implicit default classifier with an explicit default classifier.

NOTE: Only the IEEE 802.1p classifier is supported in Layer 2-only interfaces. You must explicitly apply this classifier to the interface as shown in "Default IEEE 802.1p Classifier" on page 68.

NOTE: Although several CoS values map to the expedited-forwarding (`ef`) and assured-forwarding (`af`) classes, by default no resources are assigned to these forwarding classes. All `af` classes other than `af1x` are mapped to best-effort, because RFC 2597, *Assured Forwarding PHB Group*, prohibits a node from aggregating classes.

You can apply IEEE 802.1p classifiers to interfaces that are part of VPLS routing instances.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
13.3R7	Support was added for filtering on Differentiated Services Code Point (DSCP) and forwarding class for Routing Engine sourced packets, including IS-IS packets encapsulated in generic routing encapsulation (GRE).

RELATED DOCUMENTATION

Default IP Precedence Classifier | 64
Default DSCP and DSCP IPv6 Classifiers | 65
Default MPLS EXP Classifier | 66
Default IEEE 802.1p Classifier | 68
Default IEEE 802.1ad Classifier | 70
Configure Behavior Aggregate Classifiers | 80
Apply Behavior Aggregate Classifiers to Interfaces | 83
Assign CoS Levels to Packets Based on Multiple Packet Header Fields | 148
Rewriting Packet Headers to Ensure Forwarding Behavior | 558

Default IP Precedence Classifier

By default, Junos automatically assigns an implicit IP precedence classifier called `ipprec-compatibility` to all logical interfaces. The `ipprec-compatibility` IP precedence classifier maps IP precedence bits to forwarding classes and packet loss priorities (PLPs), as shown in Table 2 on page 64.

Table 2: Default IP Precedence (ipprec-compatibility) Classifier

IP Precedence Bits	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`

The other default IP precedence classifier (called `ipprec-default`) overrides the `ipprec-compatibility` classifier when you explicitly associate it with a logical interface. To associate `ipprec-default` with a logical interface, include the `default` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `classifiers inet-precedence]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number classifiers inet-
precedence]
default;
```

Table 3 on page 65 shows the forwarding class and PLP that Junos assigns to the IP precedence bits when you apply the default IP precedence classifier.

Table 3: Default IP Precedence (ipprec-default) Classifier

IP Precedence Bits	Forwarding Class	PLP
000	`best-effort`	`low`
001	`assured-forwarding`	`low`
010	`best-effort`	`low`
011	`best-effort`	`low`
100	`best-effort`	`low`
101	`expedited-forwarding`	`low`
110	`network-control`	`low`
111	`network-control`	`high`

RELATED DOCUMENTATION

Apply Behavior Aggregate Classifiers to Interfaces | 83

Default DSCP and DSCP IPv6 Classifiers

To enable the default DiffServ code point (DSCP) classifier, include the `default` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`unit-number`* `classifiers dscp]` hierarchy level.

To enable the default DSCP IPv6 classifier, include the `default` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`unit-number`* `classifiers dscp-ipv6]` hierarchy level.

Table 4 on page 66 shows the forwarding class and packet loss priority (PLP) that Junos assigns to each well-known DSCP when you apply the explicit default DSCP or DSCP IPv6 classifier.

Table 4: Default DSCP and DSCP IPv6 Classifiers

DSCP and DSCP IPv6 Code Point	Forwarding Class	PLP
`000000`	`best-effort`	`low`
`001010`	`assured-forwarding`	`low`
`001100`	`assured-forwarding`	`high`
`001110`	`assured-forwarding`	`high`
`101110`	`expedited-forwarding`	`low`
`110000`	`network-control`	`low`
`111000`	`network-control`	`low`
all other code points	`best-effort`	`low`

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 60

Default Aliases for CoS Value Bit Patterns | 71

Change the Default Queuing and Marking of Host Outbound Traffic | 328

classifiers (Logical Interface)

Default MPLS EXP Classifier

MPLS CoS works with the routing device’s general CoS functionality.

When IP traffic enters a label-switched path (LSP) tunnel, the ingress device marks all packets with a CoS value, which is used to place the traffic into a transmission queue. On the routing device, each physical interface has up to eight transmission queues. The CoS value is encoded as part of the MPLS header and remains in the packets until the MPLS header is removed when the packets exit the egress routing device. The routing devices within the LSP utilize the CoS value set at the ingress device. The CoS value is encoded by means of the CoS bits (also known as the EXP bits).

If you do not configure any CoS features, Junos applies the default general CoS settings. For MPLS CoS, you might want to prioritize how the transmission queues are serviced by configuring weighted round-robin (WRR) and also configuring congestion avoidance using random early detection (RED).

If you enable the MPLS protocol family on a logical interface, Junos automatically applies the default MPLS EXP classifier to that logical interface.

Table 5 on page 67 lists the default MPLS classifier mapping of EXP bits to forwarding classes and loss priorities..

Table 5: Default MPLS EXP Classification

MPLS EXP Bits	Forwarding Class	Loss Priority
`000`	`best-effort`	`low`
`001`	`best-effort`	`high`
`010`	`expedited-forwarding`	`low`
`011`	`expedited-forwarding`	`high`
`100`	`assured-forwarding`	`low`
`101`	`assured-forwarding`	`high`
`110`	`network-control`	`low`
`111`	`network-control`	`high`

Starting with Junos OS Release 21.1, PTX routers support two forwarding classes and four loss priorities for MPLS EXP default classification, as Table 6 on page 68 shows.

Table 6: Default MPLS EXP Classification for PTX Routers

MPLS EXP Bits	Forwarding Class	Loss Priority
`000`	`best-effort`	`low`
`001`	`best-effort`	`high`
`010`	`best-effort`	`medium-low`
`011`	`best-effort`	`medium-high`
`100`	`network-control`	`medium-low`
`101`	`network-control`	`medium-high`
`110`	`network-control`	`low`
`111`	`network-control`	`high`

RELATED DOCUMENTATION

Configuring Class of Service for MPLS LSPs

Default Aliases for CoS Value Bit Patterns | **71**

code-point-aliases

Default IEEE 802.1p Classifier

Table 7 on page 69 shows the forwarding class and PLP that Junos assigns to each IEEE 802.1p CoS value when you apply the explicit default IEEE 802.1p classifier. To apply the default IEEE 802.1p classifier, include the `default` statement at the `[edit class-of-service interfaces *interface-name* unit *logical-unit-number* classifiers ieee-802.1]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number classifiers ieee-802.1]
default;
```

NOTE: Junos supports only the IEEE 802.1p classifier in L2 interfaces. You must explicitly apply this classifier as shown.

Table 7: Default IEEE 802.1p Classifier

IEEE 802.1p CoS Value	Forwarding Class	PLP
000	best-effort	low
001	best-effort	high
010	expedited-forwarding	low
011	expedited-forwarding	high
100	assured-forwarding	low
101	assured-forwarding	high
110	network-control	low
111	network-control	high

RELATED DOCUMENTATION

Apply Behavior Aggregate Classifiers to Interfaces | 83

Default IEEE 802.1ad Classifier | 70

Default IEEE 802.1ad Classifier

Table 8 on page 70 shows the forwarding class and packet loss priority (PLP) that Junos assigns to each IEEE 802.1ad CoS value when you apply the explicit default IEEE 802.1ad classifier. The table is very similar to the IEEE 802.1p default table, but the DEI bit determines the loss priority. To apply the default table, include the `default` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `classifiers ieee-802.1]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number classifiers
ieee-802.1ad]
default;
```

Table 8: Default IEEE 802.1ad Classifier

IEEE 802.1ad CoS Value	Forwarding Class	PLP
0000	best- effort	low
0001	best-effort	high
0010	best- effort	low
0011	best-effort	high
0100	expedited-forwarding	low
0101	expedited-forwarding	high
0110	expedited-forwarding	low
0111	expedited-forwarding	high
1000	assured-forwarding	low
1001	assured-forwarding	high

Table 8: Default IEEE 802.1ad Classifier *(Continued)*

IEEE 802.1ad CoS Value	Forwarding Class	PLP
`1010`	`assured-forwarding`	`low`
`1011`	`assured-forwarding`	`high`
`1100`	`network-control`	`low`
`1101`	`network-control`	`high`
`1110`	`network-control`	`low`
`1111`	`network-control`	`high`

RELATED DOCUMENTATION

Configuring and Applying IEEE 802.1ad Classifiers | **755**

Default Aliases for CoS Value Bit Patterns

BA classifiers use CoS values—such as DSCP, DSCP IPv6, IP precedence, IEEE 802.1, and MPLS EXP bits—to associate incoming packets with a particular CoS service level (forwarding class and packet loss priority (PLP)). You can assign a meaningful name, or alias, to the CoS values and use this alias instead of bits when configuring CoS components. These aliases are not part of the specifications but are well known through usage. For example, the alias for DSCP `101110` is widely accepted as `ef` (for expedited forwarding).

The 21 well-known DSCPs establish 5 DiffServ service classes:

- **Best-effort (be)**—The router does not apply any special CoS handling to packets with `000000` in the DiffServ field, a backward compatibility feature. There exists a high probability that these packets will be dropped under congested network conditions.

- **Assured forwarding (af)**—The router offers a high level of assurance that the packets are delivered as long as the packet flow from the customer stays within a certain service profile (the service provider defines the values). The router accepts excess traffic, but applies a random early detection (RED) drop profile to decide if the excess packets should be dropped and not forwarded. Three drop probabilities (low, medium, and high) are defined for this service class.

- **Expedited forwarding (ef)**—The router delivers assured bandwidth, low loss, low delay, and low delay variation (jitter) end-to-end for packets in this service class. Routers accept excess traffic in this class, but in contrast to assured forwarding, out-of-profile expedited-forwarding packets can be forwarded out of sequence or dropped.

- **Conversational services (cs)**—The router delivers assured (usually low) bandwidth with low delay and jitter for packets in this service class. Packets can be dropped, but never delivered out of sequence. Packetized voice is a good example of a conversational service.

- **Network control (nc)**—The router delivers packets in this service class with a low priority as these packets are not delay-sensitive. Typically, these packets represent routing protocol hello or keepalive messages and the loss of these packets jeopardizes proper network operation, so delay is preferable to discard.

NOTE: CoS value aliases must begin with a letter and can be up to 64 characters long.

When you define classifiers, you can reference the CoS values by alias names. You can configure user-defined classifiers in terms of alias names. Changing the value of an alias alters the behavior of any classifier that references it.

Table 9 on page 72 shows the default mappings between the CoS values and standard aliases.

Table 9: Default CoS Value Aliases

Default CoS Value Alias	CoS Value
DSCP and DSCP IPv6 CoS Aliases and CoS Values	
`ef`	101110
`af11`	001010
`af12`	001100

Table 9: Default CoS Value Aliases *(Continued)*

Default CoS Value Alias	CoS Value
af13	001110
af21	010010
af22	010100
af23	010110
af31	011010
af32	011100
af33	011110
af41	100010
af42	100100
af43	100110
be	000000
cs1	001000
cs2	010000
cs3	011000
cs4	100000

Table 9: Default CoS Value Aliases *(Continued)*

Default CoS Value Alias	CoS Value
`cs5`	101000
`nc1/cs6`	110000
`nc2/cs7`	111000
MPLS EXP CoS Aliases and CoS Values	
`be`	000
`be1`	001
`ef`	010
`ef1`	011
`af11`	100
`af12`	101
`nc1/cs6`	110
`nc2/cs7`	111
IEEE 802.1 CoS Aliases and CoS Values	
`be`	000
`be1`	001

Table 9: Default CoS Value Aliases *(Continued)*

Default CoS Value Alias	CoS Value
`ef`	010
`ef1`	011
`af11`	100
`af12`	101
`nc1/cs6`	110
`nc2/cs7`	111
IEEE 802.1ad CoS Aliases and CoS Values	
`be`	0000
`be-dei`	0001
`be1`	0010
`be1-dei`	0011
`ef`	0100
`ef-dei`	0101
`ef1`	0110
`ef1-dei`	0111

Table 9: Default CoS Value Aliases *(Continued)*

Default CoS Value Alias	CoS Value
`af11`	1000
`af11-dei`	1001
`af12`	1010
`af12-dei`	1011
`nc1`	1100
`nc1-dei`	1101
`nc2`	1110
`nc2-dei`	1111
Legacy IP Precedence CoS Aliases and CoS Values	
`be`	000
`be1`	001
`ef`	010
`ef1`	011
`af11`	100
`af12`	101

Table 9: Default CoS Value Aliases *(Continued)*

Default CoS Value Alias	CoS Value
nc1/cs6	110
nc2/cs7	111

RELATED DOCUMENTATION

Define Aliases for CoS Value Bit Patterns | **77**
Default IP Precedence Classifier | **64**
Default DSCP and DSCP IPv6 Classifiers | **65**
Default MPLS EXP Classifier | **66**
Default IEEE 802.1p Classifier | **68**
Default IEEE 802.1ad Classifier | **70**
code-point-aliases

Define Aliases for CoS Value Bit Patterns

To define a CoS value alias, include the `code-point-aliases` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
code-point-aliases {
    (dscp | dscp-ipv6 | exp | ieee-802.1 | ieee-802.1ad | inet-precedence) {
        alias-name bit-pattern;
    }
}
```

The CoS marker types are as follows:

- `dscp`—Differentiated Services code point aliases for IPv4 packets.
- `dscp-ipv6`—Differentiated Services code point aliases for IPv6 packets.

- exp—Layer 2 CoS values for MPLS packets.
- ieee-802.1—Layer 2 IEEE 802.1 CoS values.
- ieee-802.1—Layer 2 IEEE 802.1ad (DEI) CoS values.
- inet-precedence—IP precedence for IPv4 packets. IP precedence mapping requires only the first three bits of the DSCP field.

For example, you might configure the following aliases:

```
[edit class-of-service]
code-point-aliases {
    dscp {
        my1 110001;
        my2 101110;
        be 000001;
        cs7 110000;
    }
}
```

To specify this configuration:

1. Specify the code-point-alias type as DSCP:

   ```
   [edit]
   user@host# edit class-of-service code-point-aliases dscp
   ```

2. Specify the alias names and DSCP 6-bit pattern.

   ```
   [edit class-of-service code-point-aliases dscp]
   user@host# set my1 110001
   user@host# set my2 101110
   user@host# set be 000001
   user@host# set cs7 110000
   ```

This configuration produces the following mapping:

```
user@host> show class-of-service code-point-aliases dscp
Code point type: dscp
Alias              Bit pattern
```

```
ef/my2             101110
af11               001010
af12               001100
af13               001110
af21               010010
af22               010100
af23               010110
af31               011010
af32               011100
af33               011110
af41               100010
af42               100100
af43               100110
be                 000001
cs1                001000
cs2                010000
cs3                011000
cs4                100000
cs5                101000
nc1/cs6/cs7        110000
nc2                111000
my1                110001
```

The following notes explain certain results in the mapping:

- `my1 110001`:
  - 110001 was not mapped to anything before, and `my1` is a new alias.
  - This statement changes nothing in the default mapping table.
- `my2 101110`:
  - 101110 now maps to `my2` as well as the default `ef`.
- `be 000001`:
  - `be` is now mapped to 000001.
  - The old value of `be`, 000000, is not associated with any alias. Packets with this DSCP value now map to the default forwarding class.
- `cs7 110000`:
  - `cs7` now maps to 110000, as well as `nc1` and `cs6`.

- The old value of `cs7`, 111000, still maps to `nc2`.

RELATED DOCUMENTATION

Default Aliases for CoS Value Bit Patterns | 71

Apply Behavior Aggregate Classifiers to Interfaces | 83

Configure Behavior Aggregate Classifiers

You can override the default IP precedence classifier (`ipprec-compatibility`) by defining a custom IP precedence classifier and applying it to a logical interface. You can also override the default IP precedence classifier by applying one of the other default BA classifiers to a logical interface.

The BA classifiers map sets the forwarding class and PLP for a specific set of code-point aliases or bit patterns. The inputs of the map are CoS values aliases or bit patterns. The outputs of the map are the forwarding class and the PLP. For more information about how CoS maps work, see "Mapping CoS Component Inputs to Outputs" on page 9.

The classifiers work as follows:

- `dscp`—Handles incoming IPv4 packets
- `dscp-ipv6`—Handles incoming IPv6 packets
- `exp`—Handles MPLS packets using L2 headers
- `ieee-802.1`—Handles L2 CoS
- `ieee-802.1ad`—Handles IEEE 802.1ad formats (including DEI bit)
- `inet-precedence`—Handles incoming IPv4 packets. IP precedence mapping requires only the upper three bits of the DSCP field.

A classifier takes a specified Cos value as either the literal bit pattern or as a defined alias and attempts to match it to the type of packet arriving on the interface. If the information in the packet's header matches the specified pattern, the packet is sent to the appropriate queue, defined by the forwarding class associated with the classifier.

NOTE: On MX Series and EX Series switches that do not have tricolor marking (TCM) enabled, the loss priority can be configured only by setting the PLP within a multifield classifier. This setting can then be used by the appropriate drop profile map and rewrite rule. For more information, see "Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows" on page 471.

Use the following configuration statements to define new classifiers for all CoS value types:

```
[edit class-of-service]
classifiers {
    (dscp | dscp-ipv6 | exp | ieee-802.1 | ieee-802.1ad | inet-precedence) classifier-name {
        import [classifier-name | default];
        forwarding-class class-name {
            loss-priority level code-points [ aliases ] [ bit-patterns ];
        }
    }
}
```

To define a new classifier for all CoS value types:

1. Specify the type and name of the new classifier. For example, to create a DSCP type classifier called `class1`:

   ```
   [edit]
   user@host# edit class-of-service classifiers dscp class1
   ```

2. (Optional) Specify the forwarding class associated with the classifier.

   ```
   [edit class-of-service classifiers dscp class1]
   user@host# edit forwarding-class class-name
   ```

3. (Optional) Specify the packet loss priority (PLP) value and for a specific set of code-point aliases and bit patterns.

   ```
   [edit class-of-service classifiers dscp class1 forwarding-class best-effort]
   user@host# set loss-priority level code-points (aliases | bit-patterns)
   ```

When *tricolor marking* is enabled, Junos supports four classifier PLP designations: `low`, `medium-low`, `medium-high`, and `high`. For example, in the following configuration, Junos assigns the `assured-forwarding` forwarding class and `medium-low` PLP to all packets entering the interface with the `101110` CoS value:

1. Map the `assured-forwarding` forwarding class and `medium-low` PLP to the CoS value of `101110`.

   ```
   [edit class-of-service classifiers dscp class1]
   user@host# set forwarding-class assured-forwarding loss-priority medium-low code-points 101110
   ```

2. Verify the configuration.

   ```
   [edit class-of-service classifiers dscp class1]
   user@host# show
   ```

   ```
   forwarding-class assured-forwarding {
       loss-priority medium-low code-points 101110;
   }
   ```

   To use this classifier, you must configure the settings for the `assured-forwarding` forwarding class at the `[edit class-of-service forwarding-classes queue` *`queue-number`* `assured-forwarding]` hierarchy level. For more information, see "Understanding How Forwarding Classes Assign Classes to Output Queues" on page 286.

You can use any table, including the default, in the definition of a new classifier by including the `import` statement. The imported classifier is used as a template and is not modified. Whenever you commit a configuration that assigns a new *`class-name`* and `loss-priority` value to a CoS value alias or bit pattern, it replaces that entry in the imported classifier template. As a result, you must explicitly specify every CoS value in every designation that requires modification. For instance, to import the default DSCP classifier:

1. Specify the type and name of the new classifier. For example, to create a new DSCP type classifier called class1:

   ```
   [edit]
   user@host# edit class-of-service classifiers dscp class1
   ```

2. Specify the default DSCP classifier.

   ```
   [edit class-of-service classifiers dscp class1]
   user@host# set import default
   ```

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 60

Apply Behavior Aggregate Classifiers to Interfaces | 83

Apply Behavior Aggregate Classifiers to Interfaces

This topic describes how to apply BA classifiers to interfaces.

When you apply BA classifiers to an interface, you can use interface wildcards for the *interface-name* and *logical-unit-number*.

Family-specific classifiers take precedence over IEEE 802.1p BA classifiers. For example, if you configure a logical interface to use both an MPLS EXP and an IEEE 802.1p classifier, the EXP classifier takes precedence. MPLS-labeled packets are evaluated by the EXP classifier, and all other packets are evaluated by the IEEE 802.1p classifier. The same is true about other classifiers when combined with IEEE 802.1p classifiers on the same logical interface.

You can configure user-defined DSCP-based BA classification for MPLS interfaces or VPLS or L3 VPN routing instances (LSI interfaces). Junos does not support DSCP-based classification for MPLS packets for L2 VPNs.

NOTE: If you do not apply a DSCP classifier, Junos applies the default EXP classifier to MPLS traffic. You might need to maintain the original classifier of the incoming packet, where you neither want to configure a custom classifier for the interface nor accept the default classifier, which would override the original classifier. In that case, you can apply the `no-default` option for the interface. For example:

```
[edit class-of-service]
interfaces interface-name unit unit-number {
    classifiers {
        no-default;
    }
}
```

You can apply DSCP classification for MPLS traffic in the following usage scenarios:

- In an L3 VPN using a label-switched interface (LSI) routing instance:
    - DSCP classifier applied under `[edit class-of-service routing-instances]` on the egress PE router.
- In VPLS using an LSI routing instance:

    - DSCP classifier applied under `[edit class-of-service routing-instances]` on the egress PE router.

- In an L3 VPN using a virtual tunnel (VT) routing instance:

    - DSCP classifier applied under `[edit class-of-service interfaces]` on the core-facing interface on the egress PE router.

- In VPLS using the VT routing instance

- MPLS forwarding:

    - DSCP classifier applied under `[edit class-of-service interfaces]` on the ingress core-facing interface on the provider (P) or egress PE router.

> **NOTE**: MPLS forwarding when the label stacking is greater than 2 is not supported.

> **NOTE**: If you don't configure a classifier on a routing-instance at a PE router, then the default EXP classifier still applies for MPLS traffic from the core. If you don't want the default routing instance classification, then configure the below option at the `[edit class-of-service]` hierarchy to override routing instance based default classifiers.
>
> ```
> set class-of-service routing-instances routing-instance-name classifiers no-default
> ```
>
> You need to configure this for each routing instance so that MPLS packets from the core are not classified at the routing instance level

You can apply BA classifiers to a routing instance or a logical interface, depending on where you want to classify the packets:

- To classify MPLS packets on the routing instance at the egress PE, include the `dscp` or `dscp-ipv6` statements at the `[edit class-of-service routing-instances` *`routing-instance-name`* `classifiers]` hierarchy level. For details, see "Applying MPLS EXP Classifiers to Routing Instances" on page 116.

- To classify MPLS packets at the core-facing interface, apply the classifier at the `[edit class-of-service interface` *`interface-name`* `unit` *`unit-name`* `classifiers (dscp | dscp-ipv6)` *`classifier-name`* `family mpls]` hierarchy level. The following procedure describes this method.

In the following example, you define a DSCP classifier for IPv4 named `dscp-ipv4-classifier` and a corresponding IPv6 DSCP classifier for the `fc-af11-class` forwarding class. You then apply the IPv4 classifier to MPLS traffic and the IPv6 classifier to Internet traffic on interface ge-2/0/3.0. Or you can apply the same classifier to both MPLS and IP traffic on interface ge-2/2/0. This example shows both of these methods.

1. Define the IPv4 classifier.

```
[edit class-of-service]
user@host# set classifiers dscp dscp-ipv4-classifier forwarding-class fc-af11-class loss-
priority low code-points 000100
```

2. Define the IPv6 classifier.

```
[edit class-of-service]
user@host# set classifiers dscp-ipv6 dscp-ipv6-classifier forwarding-class fc-af11-class loss-
priority low code-points af11
```

3. (Optional) Apply the IPv4 classifier to MPLS traffic and the IPv6 classifier to Internet traffic on interface ge-2/0/3.0.

```
[edit class-of-service]
user@host# set interfaces ge-2/0/3 unit 0 classifiers dscp dscp-ipv4-classifier family mpls
user@host# set interfaces ge-2/0/3 unit 0  classifiers dscp-ipv6 dscp-ipv6-classifier family
inet
```

4. Confirm the configuration.

```
[edit class-of-service]
user@host# show
```

```
classifiers {
    dscp dscp-ipv4-classifier {
        forwarding-class fc-af11-class {
            loss-priority low code-points 000100;
        }
    }
    dscp-ipv6 dscp-ipv6-classifer {
        forwarding-class fc-af11-class {
            loss-priority low code-points af11;
        }
    }
}
interfaces {
    ge-2/0/3 {
        unit 0 {
            classifiers {
                dscp dscp-ipv4-classifier {
                    family mpls;
                }
                dscp-ipv6 dscp-ipv6-classifier {
                    family inet;
                }
            }
        }
    }
}
```

5. (Optional) Apply the same classifier, named `dscp-mpls-and-inet`, to both MPLS and IP traffic on interface ge-2/2/0.

```
[edit class-of-service]
user@host# set interfaces ge-2/2/0 unit 0 classifiers dscp dscp-mpls-and-inet family [mpls
inet]
```

6. Confirm the configuration.

```
[edit class-of-services interface ge-2/2/0]
user@host# show
```

```
unit 0 {
    classifiers {
        dscp dscp-mpls-and-inet {
            family [ mpls inet ];
        }
    }
}
```

NOTE: This is not a complete configuration.

NOTE: You can apply DSCP and DSCP IPv6 classifiers to explicit null MPLS packets. The `family mpls` statement works the same on both explicit null and non-null MPLS labels.

RELATED DOCUMENTATION

Apply DSCP Classifiers to MPLS Traffic | 110

Example: Configure and Apply a Default DSCP BA Classifier

IN THIS SECTION

A Junos classifier identifies and separates traffic flows and provides the means to prioritize traffic later in the CoS process.

A BA classifier performs this function by associating well-known CoS values with forwarding classes and loss priorities. To enable a default classifier, you simply apply it to your device interfaces. If you do not apply a default classifier to an interface, the classifier does not take effect.

Junos provides multiple default BA classifier types, which you can combine and supplement with custom BA classifiers as needed to achieve your overall traffic classification goals. This example shows how to apply the default DSCP classifier and verify its functionality.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based, or it can be software running on a server or host machine. If you do not have access to a traffic generator, you can use extended ping for verification. We show this approach as well.

The functionality in this procedure is widely supported on devices that run Junos. The example shown here was tested and verified on MX Series routers.

Overview

IN THIS SECTION

The basis of Junos CoS is traffic differentiation. Assigning traffic to different service classes provides the necessary differentiation. From the view of a router, the service class assigned to a packet defines how the router behaves toward the packet. The concept of traffic differentiation is present in every CoS tool, and as a result, classes of service are present across the entire CoS design. A classifier has one input, the incoming packet, and it has *N* possible outputs, where *N* is the number of possible classes of service into which the packet can be classified.

BA classification is used when the traffic coming into your device already has trusted CoS values in the packet header. For example, the default DSCP BA classifier specifies that packets coming in with code points 000000 are assigned to the best-effort forwarding class and given a loss priority of low.

A forwarding class and loss priority are assigned by default to each well-known DSCP. To view this, run the `show class-of-service classifier` command.

```
user@host> show class-of-service classifier type dscp

Classifier: dscp-default, Code point type: dscp, Index: 7
  Code point         Forwarding class                   Loss priority
  000000             best-effort                        low
  000001             best-effort                        low
  000010             best-effort                        low
  000011             best-effort                        low
  000100             best-effort                        low
  000101             best-effort                        low
  000110             best-effort                        low
  000111             best-effort                        low
  001000             best-effort                        low
  001001             best-effort                        low
  001010             assured-forwarding                 low
  001011             best-effort                        low
  001100             assured-forwarding                 high
  001101             best-effort                        low
  001110             assured-forwarding                 high
  001111             best-effort                        low
```

```
010000             best-effort                         low
010001             best-effort                         low
010010             best-effort                         low
010011             best-effort                         low
010100             best-effort                         low
010101             best-effort                         low
010110             best-effort                         low
010111             best-effort                         low
011000             best-effort                         low
011001             best-effort                         low
011010             best-effort                         low
011011             best-effort                         low
011100             best-effort                         low
011101             best-effort                         low
011110             best-effort                         low
011111             best-effort                         low
100000             best-effort                         low
100001             best-effort                         low
100010             best-effort                         low
100011             best-effort                         low
100100             best-effort                         low
100101             best-effort                         low
100110             best-effort                         low
100111             best-effort                         low
101000             best-effort                         low
101001             best-effort                         low
101010             best-effort                         low
101011             best-effort                         low
101100             best-effort                         low
101101             best-effort                         low
101110             expedited-forwarding                low
101111             best-effort                         low
110000             network-control                     low
110001             best-effort                         low
110010             best-effort                         low
110011             best-effort                         low
110100             best-effort                         low
110101             best-effort                         low
110110             best-effort                         low
110111             best-effort                         low
111000             network-control                     low
111001             best-effort                         low
111010             best-effort                         low
```

```
  111011            best-effort                        low
  111100            best-effort                        low
  111101            best-effort                        low
  111110            best-effort                        low
  111111            best-effort                        low
```

The forwarding class determines the output queue. By default, all best-effort traffic uses queue 0.

To view the queues that are associated, by default, with each forwarding class, use the `show class-of-service forwarding-class` command. (For clarity, some of the output is excluded.)

```
user@host> show class-of-service forwarding-class

Forwarding class                      ID       Queue
  best-effort                          0        0
  expedited-forwarding                 1        1
  assured-forwarding                   2        2
  network-control                      3        3
```

The loss priority is used by schedulers in conjunction with the random early detection (RED) algorithm to control packet discard during periods of congestion. When you are thinking about loss priorities, keep in mind that unless you configure them, they have no meaning. The default drop behavior is to wait until the queue is 100 percent full and then begin dropping packets indiscriminately. When the queue dips below 100 percent full, packets stop dropping.

The default drop behavior is shown in the `show class-of-service drop-profile` command.

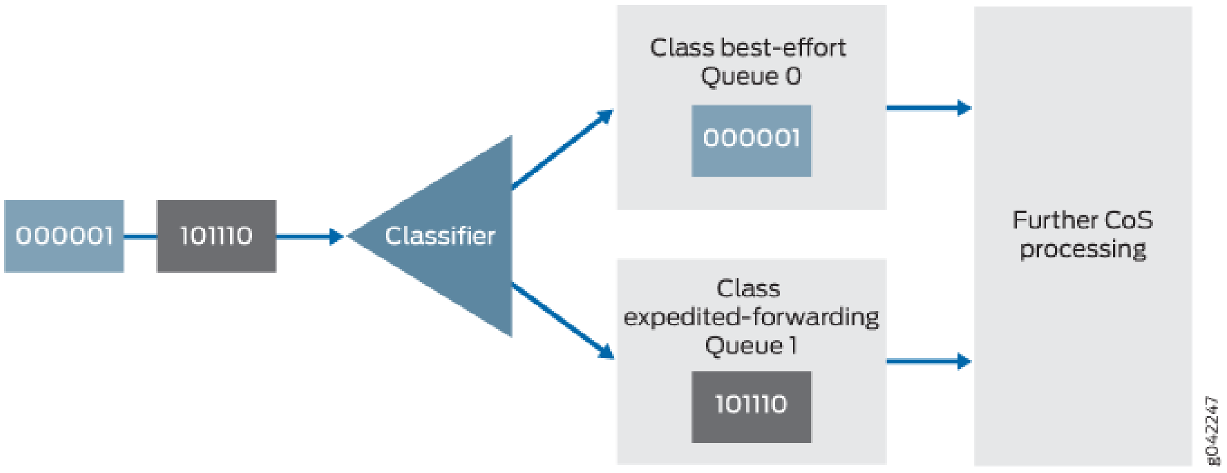
```
user@host> show class-of-service drop-profile

Drop profile: <default-drop-profile>, Type: discrete, Index: 1
  Fill level    Drop probability
         100                 100
```

To create meanings for the various loss priorities, you must configure custom drop profiles. For example, you might define the low loss priority to mean a 10 percent drop probability when the queue is 75 percent full and a 40 percent drop probability when the queue fill level is 95 percent. You might define the high loss priority to mean a 50 percent drop probability when the fill level is 25 percent and a 90 percent drop probability when the fill level is 50 percent. Custom drop profiles are not included in this example, but are mentioned here for clarity because classifiers assign loss priorities. It is important to understand that these assignments are meaningless until you create drop profiles.

The default classifier operation is shown in Figure 12 on page 91. The figure shows two IPv4 packets entering an interface and being classified according to the DSCP code points in the packet headers.

Figure 12: Behavior Aggregate Classifier with Two Queues



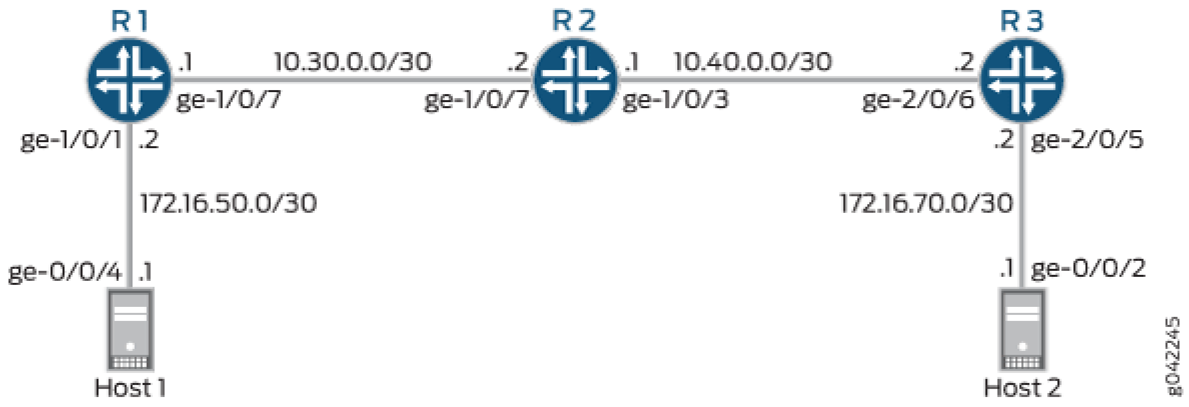
Classifiers are described in more detail in the following Juniper Networks Learning Byte video.

Video: Class of Service Basics, Part 2: Classification Learning Byte

Topology

Figure 13 on page 91 shows the sample network.

Figure 13: Behavior Aggregate Classifier Scenario



It is important to apply your class-of-service configuration across the topology, instead of applying it to a single device. Furthermore, even though classification takes effect on incoming interfaces, you should apply BA classifiers to all core and core-facing interfaces. This is because a single interface can be either incoming or outgoing, depending on the direction of the traffic. For example, as traffic flows from Host 1 to Host 2, the incoming interfaces are ge-1/0/7 on Device R2 and ge-2/0/6 on Device R3. As traffic flows in the other direction, from Host 2 to Host R1, the incoming interfaces are ge-1/0/3 on Device R2 and ge-1/0/7 on Device R1.

The BA classifier is not applied to ge-1/0/1 on Device R1 or ge-2/0/5 on Device R3, because these interfaces are not core facing. Generally, at the edge-facing interfaces, you would use a multifield classifier, not a BA classifier.

"CLI Quick Configuration" on page 92 shows the configuration for all of the Juniper Networks devices in Figure 13 on page 91. The section "Step-by-Step Procedure" on page 93 describes the steps on Device R2.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
set interfaces ge-1/0/7 unit 0 family inet address 10.30.0.1/30
set class-of-service interfaces ge-1/0/7 unit 0 classifiers dscp default
```

Device R2

```
set interfaces ge-1/0/3 unit 0 family inet address 10.40.0.1/30
set interfaces ge-1/0/7 unit 0 family inet address 10.30.0.2/30
set class-of-service interfaces ge-1/0/3 unit 0 classifiers dscp default
set class-of-service interfaces ge-1/0/7 unit 0 classifiers dscp default
```

Device R3

```
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/6 unit 0 family inet address 10.40.0.2/30
set class-of-service interfaces ge-2/0/6 unit 0 classifiers dscp default
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To enable the default DSCP behavior aggregate classifier:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@R2# set ge-1/0/3 unit 0 family inet address 10.40.0.1/30
   user@R2# set ge-1/0/7 unit 0 family inet address 10.30.0.2/30
   ```

2. Enable the default DSCP classifier on the interfaces.

   ```
   [edit class-of-service interfaces]
   user@R2# set ge-1/0/3 unit 0 classifiers dscp default
   user@R2# set ge-1/0/7 unit 0 classifiers dscp default
   ```

Results

From configuration mode, confirm your configuration by entering the `show interfaces` and `show class-of-service` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R2# show interfaces
ge-1/0/3 {
    unit 0 {
        family inet {
            address 10.40.0.1/30;
```

```
        }
    }
}
ge-1/0/7 {
    unit 0 {
        family inet {
            address 10.30.0.2/30;
        }
    }
}
```

```
user@R2# show class-or-service
interfaces {
    ge-1/0/3 {
        unit 0 {
            classifiers {
                dscp default;
            }
        }
    }
    ge-1/0/7 {
        unit 0 {
            classifiers {
                dscp default;
            }
        }
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Behavior Aggregate Classifiers

Purpose

Verify that the default behavior aggregate classifier is enabled on the device interfaces. Keep in mind that although the classifier operates on incoming packets, you view the resulting queue assignment on the outgoing interface.

Action

1. Clear the interface statistics on Device R2.

   ```
   user@R2> clear interface statistics ge-1/0/3
   ```

2. Using extended ping from Device R1 or a packet generator running on a host or server, send packets with the code point set to 001010.

   Both methods are shown here. The packet generator used is hping.

   - When you are using extended ping to set the DSCP code points in the IPv4 packet header, the type-of-service (ToS) decimal value (in this case, 40) is required in the `tos` option of the `ping` command.

   - When you are using hping to set the DSCP code points in the IPv4 packet header, the ToS hex value (in this case, 28) is required in the `--tos` option of the `hping` command.

   If your binary-to-hex or binary-to-decimal conversion skills are rusty, you can use an online calculator, such as http://www.mathsisfun.com/binary-decimal-hexadecimal-converter.html .

   NOTE: When you convert a binary DSCP code point value, be sure to add two extra zeros at the end. So instead of 001010, use 00101000. These 0 values (the 7th and 8th bits) are reserved and ignored, but if you do not include them in the conversion, your hex and decimal values will be incorrect.

   Extended Ping Sent from Device R1

   ```
   user@R1> ping 172.16.70.1 tos 40 rapid count 25

   PING 172.16.70.1 (172.16.70.1): 56 data bytes
   !!!!!!!!!!!!!!!!!!!!!!!!!
   --- 172.16.70.1 ping statistics ---
   ```

```
25 packets transmitted, 25 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.430/0.477/0.847/0.079 ms
```

hping Sent from Host 1

```
root@host1> hping 172.16.70.1 --tos 28 -c 25

HPING 172.16.70.1 (eth1 172.16.70.1): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.3 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=1 win=0 rtt=0.6 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=2 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=3 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=4 win=0 rtt=0.6 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=5 win=0 rtt=0.3 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=6 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=7 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=8 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=9 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=10 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=11 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=12 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=13 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=14 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=15 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=16 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=17 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=18 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=19 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=20 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=21 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=22 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=23 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=24 win=0 rtt=0.4 ms
```

3. On Device R2, verify that queue 2 is incrementing.

   Code point 001010 is associated with assured-forwarding, which uses queue 2 by default.

```
user@R2> show interfaces extensive ge-1/0/3 | find "queue counters"
```

```
Queue counters:      Queued packets  Transmitted packets      Dropped packets
    0                    0               0                        0
    1                    0               0                        0
    2                    50              25                       0
    3                    3               3                        0
  Queue number:              Mapped forwarding classes
    0                        best-effort
    1                        expedited-forwarding
    2                        assured-forwarding
    3                        network-control
```

Meaning

The output shows that queue 2 has incremented by 50 packets after sending 50 packets through the router.

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | **60**

Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows | **471**

Example: Configure and Apply Custom BA Classifiers

IN THIS SECTION

This example shows how to configure BA classifiers for a device to determine forwarding treatment of packets.

Requirements

Before you begin, determine the default forwarding class and PLP for each well-known DSCP for which you want to configure for a BA classifier. See *Default Behavior Aggregate Classification*.

Overview

You configure BA classifiers to classify packets that contain valid DSCPs to appropriate queues. Once configured, you must apply the BA classifier to the correct interfaces. You can override the default IP precedence classifier by defining a classifier and applying it to a logical interface. To define new classifiers for all code point types, include the `classifiers` statement at the `[edit class-of-service]` hierarchy level.

In this example, you set the DSCP BA classifier to `ba-classifier` as the default DSCP map. You set:

- A best-effort forwarding class as `be-class`
- An expedited forwarding class as `ef-class`
- An assured forwarding class as `af-class`
- A network control forwarding class as `nc-class`

Finally, you apply the BA classifier to an interface called ge-0/0/0.

Table 10 on page 98 shows how the behavior aggregate classifier assigns loss priorities to incoming packets in the four forwarding classes.

Table 10: Sample ba-classifier Loss Priority Assignments

ba-classifier Forwarding Class	For CoS Traffic Type	ba-classifier Assignments
`be-class`	Best-effort traffic	High-priority code point: `000001`
`ef-class`	Expedited forwarding traffic	High-priority code point: `101111`
`af-class`	Assured forwarding traffic	High-priority code point: `001100`
`nc-class`	Network control traffic	High-priority code point: `110001`

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

```
set class-of-service classifiers dscp ba-classifier import default
set class-of-service classifiers dscp ba-classifier forwarding-class be-class loss-priority high
code-points 000001
set class-of-service classifiers dscp ba-classifier forwarding-class ef-class loss-priority high
code-points 101111
set class-of-service classifiers dscp ba-classifier forwarding-class af-class loss-priority high
code-points 001100
set class-of-service classifiers dscp ba-classifier forwarding-class nc-class loss-priority high
code-points 110001
set class-of-service interfaces ge-0/0/0 unit 0 classifiers dscp ba-classifier
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure behavior aggregate classifiers for a device:

1. Configure the class of service.

```
[edit]
user@host# edit class-of-service
```

2. Configure behavior aggregate classifiers for DiffServ CoS.

```
[edit class-of-service]
user@host# edit classifiers dscp ba-classifier
user@host# set import default
```

3. Configure a best-effort forwarding class classifier.

```
[edit class-of-service classifiers dscp ba-classifier]
user@host# set forwarding-class be-class loss-priority high code-points 000001
```

4. Configure an expedited forwarding class classifier.

```
[edit class-of-service classifiers dscp ba-classifier]
user@host# set forwarding-class ef-class loss-priority high code-points 101111
```

5. Configure an assured forwarding class classifier.

```
[edit class-of-service classifiers dscp ba-classifier]
user@host# set forwarding-class af-class loss-priority high code-points 001100
```

6. Configure a network control forwarding class classifier.

```
[edit class-of-service classifiers dscp ba-classifier]
user@host# set forwarding-class nc-class loss-priority high code-points 110001
```

7. Apply the behavior aggregate classifier to an interface.

```
[edit]
user@host# set class-of-service interfaces ge-0/0/0 unit 0 classifiers dscp ba-classifier
```

NOTE: You can use interface wildcards for `interface-name` and `logical-unit-number`.

Results

From configuration mode, confirm your configuration by entering the `show class-of-service` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host# show class-of-service
classifiers {
    dscp ba-classifier {
        import default;
        forwarding-class be-class {
            loss-priority high code-points 000001;
        }
        forwarding-class ef-class {
            loss-priority high code-points 101111;
        }
        forwarding-class af-class {
            loss-priority high code-points 001100;
        }
        forwarding-class nc-class {
            loss-priority high code-points 110001;
        }
    }
    interfaces {
        ge-0/0/0 {
            unit 0 {
                classifiers {
                    dscp ba-classifier;
                }
            }
        }
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verify the DSCP Classifier

Purpose

Make sure that the DSCP classifier is configured as expected.

Action

Run the `show class-of-service classifiers name ba-classifier` command.

```
user@host> show class-of-service classifiers name ba-classifier


Classifier: ba-classifier, Code point type: dscp, Index: 10755
  Code point         Forwarding class                   Loss priority
  000000             best-effort                        low
  000001             be-class                           high
  000010             best-effort                        low
  000011             best-effort                        low
  000100             best-effort                        low
  000101             best-effort                        low
  000110             best-effort                        low
  000111             best-effort                        low
  001000             best-effort                        low
  001001             best-effort                        low
  001010             assured-forwarding                 low
  001011             best-effort                        low
  001100             af-class                           high
  001101             best-effort                        low
  001110             assured-forwarding                 high
  001111             best-effort                        low
  010000             best-effort                        low
  010001             best-effort                        low
  010010             best-effort                        low
```

```
010011             best-effort                          low
010100             best-effort                          low
010101             best-effort                          low
010110             best-effort                          low
010111             best-effort                          low
011000             best-effort                          low
011001             best-effort                          low
011010             best-effort                          low
011011             best-effort                          low
011100             best-effort                          low
011101             best-effort                          low
011110             best-effort                          low
011111             best-effort                          low
100000             best-effort                          low
100001             best-effort                          low
100010             best-effort                          low
100011             best-effort                          low
100100             best-effort                          low
100101             best-effort                          low
100110             best-effort                          low
100111             best-effort                          low
101000             best-effort                          low
101001             best-effort                          low
101010             best-effort                          low
101011             best-effort                          low
101100             best-effort                          low
101101             best-effort                          low
101110             expedited-forwarding                 low
101111             ef-class                             high
110000             network-control                      low
110001             nc-class                             high
110010             best-effort                          low
110011             best-effort                          low
110100             best-effort                          low
110101             best-effort                          low
110110             best-effort                          low
110111             best-effort                          low
111000             network-control                      low
111001             best-effort                          low
111010             best-effort                          low
111011             best-effort                          low
111100             best-effort                          low
111101             best-effort                          low
```

```
  111110          best-effort                    low
  111111          best-effort                    low
```

Meaning

Notice that the default classifier is incorporated into the customer classifier. If you were to remove the `import default` statement from the custom classifier, the custom classifier would look like this:

```
user@host> show class-of-service classifier name ba-classifier
Classifier: ba-classifier, Code point type: dscp, Index: 10755
  Code point          Forwarding class               Loss priority
  000001              be-class                       high
  001100              af-class                       high
  101111              ef-class                       high
  110001              nc-class                       high
```

Verify That the Classifier Is Applied to the Interfaces

Purpose

Make sure that the classifier is applied to the correct interfaces.

Action

Run the `show class-of-service interface` command.

```
user@host> show class-of-service interface ge-0/0/0



Physical interface: ge-0/0/0, Index: 144
Queues supported: 8, Queues in use: 4
  Scheduler map: <default>, Index: 2
  Congestion-notification: Disabled

  Logical interface: ge-0/0/0.0, Index: 333
Object                  Name                  Type                    Index
Classifier              ba-classifier         dscp                    10755
```

Meaning

The interface is configured as expected.

RELATED DOCUMENTATION

Interfaces User Guide for Security Devices

Classification Overview

Sample Behavior Aggregate Classification

Understanding Packet Loss Priorities

DSCP Classification for VPLS

You can perform DSCP classification for IPv4 packets on Ethernet interfaces that are part of a VPLS routing instance on the ingress PE router.

To perform DSCP classification for IPv4 packets on Ethernet interfaces that are part of a VPLS routing instance on the ingress PE router, you must ensure the following:

- The correct encapsulation statement based on PIC type is configured for the interface
- The DSCP classifier is defined (default is allowed) at the `[edit class-of-service classifiers]` hierarchy level
- The defined DSCP classifier is applied to the interface
- The interface is included in the VPLS routing instance on the ingress of the PE router

A VPLS routing instance configured with the `no-tunnel-services` option has a default MPLS EXP classifier applied to the LSI for all VPLS packets coming from the remote VPLS PE. You can modify this default classifier.

On routing devices with eight queues, the default classification applied to `no-tunnel-services` VPLS packets are shown in Table 11 on page 105.

Table 11: Default VPLS Classifiers

MPLS Label EXP Bits	Forwarding Class/Queue
000	0

Table 11: Default VPLS Classifiers ***(Continued)***

MPLS Label EXP Bits	Forwarding Class/Queue
001	1
010	2
011	3
100	4
101	5
110	6
111	7

NOTE: Forwarding class to queue number mapping is not always one-to-one. Forwarding classes and queues are only the same when default forwarding-class-to-queue mapping is in effect. For more information about configuring forwarding class and queues, see "Configuring a Custom Forwarding Class for Each Queue" on page 291.

VPLS filters and policers act on an L2 frame that includes the MAC header (after the application of any VLAN rewrite or other rules), but does not include the cyclical redundancy check (CRC) field.

NOTE: On MX Series routers, if you apply a counter in a firewall for egress MPLS or VPLS packets with the EXP bits set to 0, the counter does not tally these packets.

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | **60**

Configure CoS for MPLS LSPs

IN THIS SECTION

The following sections provide an overview of MPLS CoS and describe how to configure the MPLS CoS value:

CoS for MPLS Overview

When IP traffic enters an LSP tunnel, the ingress router marks all packets with a CoS value, which is used to place the traffic into a transmission priority queue. Junos encodes the CoS value as part of the MPLS header and remains in the packets until the MPLS header is removed when the packets exit the egress router. The routers within the LSP use the CoS value set at the ingress. The CoS value is encoded by means of the CoS bits (also known as the EXP bits). For more information, see MPLS Label Allocation.

MPLS CoS works with the router’s general CoS functionality. If you do not configure any CoS features, the default general CoS settings are used. For MPLS CoS, you might want to configure WRR to prioritize how the transmit queues are serviced. Configure congestion avoidance using RED.

Configure the MPLS CoS Values

When traffic enters an LSP tunnel, the CoS value in the MPLS header is set in one of three ways:

- The output queue number into which the packet was buffered and the PLP are written into the MPLS header and are used as the packet’s CoS value. This is default behavior and requires no configuration. "Default MPLS EXP Classifier" on page 66 explains the default MPLS CoS values and how these values are treated.

- You set a fixed CoS value on all packets entering the LSP tunnel. A fixed CoS value means that all packets entering the LSP receive the same class of service.

- You set an MPLS EXP rewrite rule to override the default behavior.

To set a fixed CoS value on all packets entering the LSP, include the `class-of-service` statement:

```
class-of-service cos-value;
```

You can include this statement at the following hierarchy levels:

- `[edit protocols mpls]`
- `[edit protocols mpls label-switched-path path-name]`
- `[edit protocols mpls label-switched-path path-name primary path-name]`
- `[edit protocols mpls label-switched-path path-name secondary path-name]`
- `[edit protocols rsvp interface interface-name link-protection]`
- `[edit protocols rsvp interface interface-name link-protection bypass destination]`
- `[edit logical-systems logical-system-name protocols mpls]`
- `[edit logical-systems logical-system-name protocols mpls label-switched-path path-name]`
- `[edit logical-systems logical-system-name protocols mpls label-switched-path path-name primary path-name]`
- `[edit logical-systems logical-system-name protocols mpls label-switched-path path-name secondary path-name]`
- `[edit logical-systems logical-system-name protocols rsvp interface interface-name link-protection ]`
- `[edit logical-systems logical-system-name protocols rsvp interface interface-name link-protection bypass destination]`

The CoS value set using the `class-of-service` statement at the `[edit protocols mpls]` hierarchy level supersedes the CoS value set at the `[edit class-of-service]` hierarchy level for an interface. Effectively, the CoS value configured for an LSP overrides the CoS value set for an interface.

The `class-of-service` statement at the `[edit protocols mpls label-switched-path]` hierarchy level assigns an initial EXP value for the MPLS shim header of packets in the LSP. This value is initialized at the ingress routing device only and overrides the rewrite configuration established for that forwarding class. However, the CoS processing (weighted round robin [WRR] and RED) of packets entering the ingress routing device is not changed by the `class-of-service` statement on an MPLS LSP. Classification is still based on the behavior aggregate (BA) classifier at the `[edit class-of-service]` hierarchy level or the multifield classifier at the `[edit firewall]` hierarchy level.

BEST PRACTICE: We recommend configuring all routing devices along the LSP to have the same input classifier for EXP, and, if a rewrite rule is configured, all routing devices should have the same rewrite configuration. Otherwise, traffic at the next LSR might be classified into a different forwarding class, resulting in a different EXP value being written to the EXP header.

The CoS value can be a decimal number from 0 through 7. This number corresponds to a 3-bit binary number. The high-order 2 bits of the CoS value select which transmit queue to use on the outbound interface card.

The low-order bit of the CoS value is treated as the PLP bit and is used to select the RED drop profile to use on the output queue. If the low-order bit is 0, the non-PLP drop profile is used, and if the low-order bit is 1, the PLP drop profile is used. It is generally expected that RED will more aggressively drop packets that have the PLP bit set. For more information about RED and drop profiles, see "RED Drop Profiles for Congestion Management" on page 462.

NOTE: Configuring the PLP drop profile to drop packets more aggressively (for example, setting the CoS value from 6 to 7) decreases the likelihood of traffic getting through.

Table 12 on page 109 summarizes how MPLS CoS values correspond to the transmit queue and PLP bit. Note that in MPLS, the mapping between the CoS bit value and the output queue is hard-coded. You cannot configure the mapping for MPLS; you can configure it only for IPv4 traffic flows, as described in "Understanding How Forwarding Classes Assign Classes to Output Queues" on page 286.

Table 12: MPLS CoS Values

MPLS CoS Value	Bits	Transmit Queue	PLP Bit
0	000	0	Not set
1	001	0	Set
2	010	1	Not set
3	011	1	Set
4	100	2	Not set
5	101	2	Set

Table 12: MPLS CoS Values *(Continued)*

MPLS CoS Value	Bits	Transmit Queue	PLP Bit
6	110	3	Not set
7	111	3	Set

Because the CoS value is part of the MPLS header, the value is associated with the packets only as they travel through the LSP tunnel. The value is not copied back to the IP header when the packets exit from the LSP tunnel.

To configure class of service (CoS) for Multiprotocol Label Switching (MPLS) packets in a label-switched path (LSP):

1. Specify the CoS value

   If you do not specify a CoS value, the IP precedence bits from the packet's IP header are used as the packet's CoS value.

Rewrite IEEE 802.1p Packet Headers with the MPLS CoS Value

You can rewrite both MPLS and IEEE 802.1p CoS values to a configured value. Rewriting these values allows you to pass the configured value to the L2 VLAN path. To rewrite both the MPLS and IEEE 802.1p CoS values, you must include the EXP and IEEE 802.1p rewrite rules in the CoS interface configuration. The EXP rewrite table is applied when you configure the IEEE 802.1p and EXP rewrite rules.

For information about how to configure the EXP and IEEE 802.1p rewrite rules, see "Rewriting Packet Headers to Ensure Forwarding Behavior" on page 558.

Apply DSCP Classifiers to MPLS Traffic

IN THIS SECTION

You can configure custom DSCP BA classification for MPLS interfaces or VPLS/L3VPN routing instances (LSI interfaces).

NOTE: You cannot configure user-defined DSCP-based BA classification for MPLS interfaces on MX Series routers or EX Series switches when ingress queuing is used.

The following examples show how you can apply DSCP classifiers for MPLS traffic on core-facing interfaces and VPLS/L3VPN routing instances. These classifiers are applicable on egress PE routers for VPLS and L3VPN cases. For plain interfaces (not VPLS/L3VPN (LSI) interfaces), these classifiers are applicable on P and egress PE routers on core-facing interfaces.

Apply a DSCP Classifier to MPLS Packets on a Core-Facing Interface

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

The following example:

1. Configures core-facing interface ge-5/3/1.0 for protocol families IPv4, IPv6, and International Organization for Standardization Open Systems Interconnection (ISO OSI)

2. Configures the DSCP classifier `dscp11`.

3. Apply the DSCP classifier to the logical interface for the MPLS family.

To configure and apply a DSCP classifier to MPLS packets on a core-facing interface:

1. Configure the core-facing interface and associated logical interfaces.

```
[edit interfaces ge-5/3/1 unit 0]
user@host # set family inet address 10.1.1.1/24
user@host # set family iso
user@host # set family inet6 address 2001:db8::1/64
user@host # set family mpls
```

2. Configure the DSCP classifier.

```
[edit  class-of-service classifiers dscp dscp11]
user@host # set forwarding-class expedited-forwarding loss-priority low code-points [ef cs5]
user@host # set forwarding-class assured-forwarding loss-priority low code-points [af21 af31
af41 cs4]
user@host # set forwarding-class assured-forwarding loss-priority high code-points [af23 af33
af43 cs2 af22 af32 af42 cs3]
user@host # set forwarding-class best-effort loss-priority low code-points [af11 cs1 af12]
user@host # set forwarding-class best-effort loss-priority high code-points af13
user@host # set forwarding-class network-control loss-priority low code-points [cs6 cs7]
```

3. Apply the classifier to the logical interface for the MPLS family.

NOTE: You cannot configure more than one classifier per family.

```
[edit class-of-service interfaces ge-5/3/1 unit 0]
user@host # set classifiers dscp dscp11 family mpls
```

4. Confirm the configuration.

```
[edit interfaces ge-5/3/1 unit 0]
user@host# show
```

```
family inet {
    address 10.1.1.1/24;
}
family iso;
family inet6 {
    address 2001:db8::1/64;
}
family mpls;
```

```
[edit class-of-service classifiers dscp dscp11]
user@host# show
```

```
forwarding-class expedited-forwarding {
    loss-priority low code-points [ ef cs5 ];
}
forwarding-class assured-forwarding {
    loss-priority low code-points [ af21 af31 af41 cs4 ];
    loss-priority high code-points [ af23 af33 af43 cs2 af22 af32 af42 cs3 ];
}
forwarding-class best-effort {
    loss-priority low code-points [ af11 cs1 af12 ];
    loss-priority high code-points af13;
}
forwarding-class network-control {
    loss-priority low code-points [ cs6 cs7 ];
}
```

```
[edit class-of-service interfaces ge-5/3/1 unit 0]
user@host# show
```

```
classifiers {
    dscp dscp11 {
        family mpls;
    }
}
```

5. Save the configuration.

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

Apply a DSCP Classifier to MPLS Traffic for L3VPN/VPLS

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

The following example:

1. Configures routing instances of type vrf or vpls.

2. Configures the DSCP classifier.

3. Attaches the classifier to the routing instance.

To configure and apply a DSCP classifier to MPLS traffic for L3VPN/VPLS:

1. Configure routing instances of type vrf or vpls.

```
[edit routing-instances vpls1]
user@host# set instance-type vpls
user@host# set interface ge-2/2/2.0
user@host# set route-distinguisher 10.255.245.51:1
user@host# set vrf-target target:1234:1
user@host# set protocols vpls site-range 10
user@host# set protocols vpls no-tunnel-services
user@host# set protocols vpls site vpls-1-site-1 site-identifier 1
```

2. Configure the DSCP classifier.

```
[edit class-of-service classifiers dscp dscp11]
user@host # set forwarding-class expedited-forwarding loss-priority low code-points [ef cs5]
user@host # set forwarding-class assured-forwarding loss-priority low code-points [af21 af31
af41 cs4]
user@host # set forwarding-class assured-forwarding loss-priority high code-points [af23 af33
af43 cs2 af22 af32 af42 cs3]
user@host # set forwarding-class best-effort loss-priority low code-points [af11 cs1 af12]
user@host # set forwarding-class best-effort loss-priority high code-points af13
user@host # set forwarding-class network-control loss-priority low code-points [cs6 cs7]
```

3. Attach the classifier to the routing instance.

```
[edit class-of-service routing-instances vpls1]
user@host # set classifiers dscp dscp11
```

NOTE: You cannot configure more than one classifier per routing instance.

4. Confirm the configuration.

```
[edit routing-instances vpls1]
user@host# show
```

```
instance-type vpls;
interface ge-2/2/2.0; ## customer facing interface
route-distinguisher 10.255.245.51:1;
```

```
vrf-target target:1234:1;
protocols {
    vpls {
        site-range 10;
        no-tunnel-services;
        site vpls-1-site-1 {
            site-identifier 1;
        }
    }
}
```

```
[edit class-of-service]
user@host# show
```

```


classifiers {
    dscp dscp11 {
        forwarding-class expedited-forwarding {
            loss-priority low code-points [ ef cs5 ];
        }
        forwarding-class assured-forwarding {
            loss-priority low code-points [ af21 af31 af41 cs4 ];
            loss-priority high code-points [ af23 af33 af43 cs2 af22 af32 af42 cs3 ];
        }
        forwarding-class best-effort {
            loss-priority low code-points [ af11 cs1 af12 ];
            loss-priority high code-points af13;
        }
        forwarding-class network-control {
            loss-priority low code-points [ cs6 cs7 ];
        }
    }
}
routing-instances {
    vpls1 {
        classifiers {
            dscp dscp11;
        }
    }
}
```

5. Save the configuration.

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

RELATED DOCUMENTATION

Apply Behavior Aggregate Classifiers to Interfaces | 83

Apply MPLS EXP Classifiers to Routing Instances

IN THIS SECTION

This topic shows how to apply MPLS EXP classifiers to routing instances.

When you enable VRF table labels and you do not explicitly apply a classifier configuration to the routing instance, Junos applies the default MPLS EXP classifier to the routing instance. For detailed information about VRF table labels, see the Junos OS VPNs Library for Routing Devices.

Table 13 on page 117 show the default MPLS EXP classification table.

Table 13: Default MPLS EXP Classifier

MPLS EXP Bits	Forwarding Class	Loss Priority
`000`	`best-effort`	`low`
`001`	`best-effort`	`high`
`010`	`expedited-forwarding`	`low`
`011`	`expedited-forwarding`	`high`
`100`	`assured-forwarding`	`low`
`101`	`assured-forwarding`	`high`
`110`	`network-control`	`low`
`111`	`network-control`	`high`

NOTE: At times, you might need to maintain the original classifier—for example with bridge domains, where you neither want to configure a custom classifier for the routing instance nor accept the default classifier. You can maintain the original MPLS EXP classifier. To do so, apply the `no-default` option for the routing instance. For example:

```
[edit class-of-service]
routing-instances routing-instance-name {
    classifiers {
        no-default;
    }
}
```

Configure and Apply Custom MPLS EXP Classifiers to Routing Instances

For routing instances with VRF table labels enabled, you can override the default MPLS EXP classifier and apply a custom classifier to a routing instance.

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

To apply a custom classifier to a routing instance:

1. Filter traffic based on the IP header.

```
[edit]
user@host# edit routing-instances routing-instance-name
user@host# set vrf-table-label
```

2. Configure the custom MPLS EXP classifier.

```
[edit]
user@host# edit class-of-service
user@host# set classifiers exp classifier-name import classifier-name forwarding-class class-
name loss-priority level code-points (aliases | bit-patterns)
user@host# set forwarding-classes queue queue-number class-name priority (high | low)
```

3. Apply the custom MPLS EXP classifier to the routing instance.

```
[edit class-of-service routing-instances routing-instance-name classifiers]
user@host# set exp classifier-name;
```

4. Commit and confirm your configuration.

```
[edit]
user@host# show class-of-service routing-instances
```

Apply Global Classifiers and Wildcard Routing Instances

To apply a classifier to all routing instances:

- Specify that the MPLS EXP classifier is for all routing instances.

```
[edit class-of-service ]
user@host# set routing-instances all classifiers exp classifier-name
```

For routing instances you associate with specific classifiers, Junos ignores the global configuration.

To use a wildcard to apply a classifier to all routing instances:

- Include an asterisk (*) in the name of the routing instance.

```
[edit]]
user@host# edit class-of-service routing-instances routing-instance-name*
user@host# set classifiers exp classifier-name
```

The wildcard configuration follows the longest match. If you include a specific configuration, it has precedence over the wildcard configuration.

NOTE: Junos supports the wildcard * and the `all` keyword at the `[edit class-of-service routing-instances]` hierarchy level but not at the `[edit routing-instances]` hierarchy level.

If you configure a routing instance at the `[edit routing-instances]` hierarchy level with, for example, the name `vpn*`, Junos treats `vpn*` as a valid and distinct routing instance name. If you then try to apply a classifier to the `vpn*` routing instance at the `[edit class-of-service routing-instances]` hierarchy level, Junos treats the `vpn*` routing instance name as a wildcard. Thus all routing instances that start with `vpn` and do not have a specific classifier applied receive the classifier associated with `vpn*`.

This same behavior applies with the `all` keyword.

Note that the * wildcard *must* be appended to an instance name at these configuration levels. The * wildcard is not a stand-alone substitute for the `all` keyword.

Apply Global MPLS EXP Classifiers to Routing Instances

This example shows how to apply a global classifier to all routing instances and then override the global classifier for a specific routing instance. For this example, we have three routing instances: `vpn1`, `vpn2`, and `vpn3`, each with VRF table label enabled. The classifier `exp-classifier-global` is applied to `vpn1` and `vpn2` (that is, all but `vpn3`, which is listed separately). The classifier `exp-classifier-3` is applied to `vpn3`.

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

To configure a global classifier for all routing instances and override the global classifier for a specific routing instance:

1. Enable the VRF table label for all three routing instances.

```
[edit routing-instances]
user@host#  set vpn1 vrf-table-label
```

```
user@host#  set vpn2 vrf-table-label
user@host#  set vpn3 vrf-table-label
```

2. Apply the EXP classifier `exp-classifier-global` to all routing instances.

```
[edit class-of-service routing-instances]
user@host# set all classifiers exp exp-classifier-global
```

3. Apply the EXP classifier `exp-classifier-3` to only the routing-instance `vpn3`.

```
[edit class-of-service routing-instances]
user@host# set vpn3 classifiers exp exp-classifier-3
```

4. Confirm your configuration.

```
[edit routing-instances]
user@host# show
```

```
vpn1 {
    vrf-table-label;
}
vpn2 {
    vrf-table-label;
}
vpn3 {
    vrf-table-label;
}
```

```
[edit class-of-service routing-instances]
user@host# show
```

```
all {
    classifiers {
        exp exp-classifier-global;
    }
}
vpn3 {
    classifiers {
        exp exp-classifier-3;
    }
}
```

Apply Classifiers by Using Wildcard Routing Instances

Configure a wildcard routing instance and override the wildcard with a specific routing instance. For this example, we have three routing instances: `vpn-red`, `vpn-yellow`, and `vpn-green`, each with VRF table label enabled. The classifier `exp-class-wildcard` is applied to `vpn-yellow` and `vpn-green`. The classifier `exp-class-red` is applied to `vpn-red`.

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

To configure a wildcard routing instance and override the wildcard with a specific routing instance:

1. Enable the VRF table label for all three routing instances.

```
[edit routing-instances]
user@host#  set vpn-red vrf-table-label
user@host#  set vpn-yellow vrf-table-label
user@host#  set vpn-green vrf-table-label
```

2. Apply the EXP classifier `exp-class-wildcard` to all routing instances by using a wildcard.

```
[edit class-of-service routing-instances]
user@host# set vpn* classifiers exp exp-class-wildcard
```

3. Apply the EXP classifier `exp-class-red` to only the routing-instance `vpn-red`.

```
[edit class-of-service routing-instances]
user@host# set vpn-red classifiers exp exp-class-red
```

4. Commit and confirm your configuration.

```
[edit routing-instances]
user@host# show
```

```
vpn-red {
    vrf-table-label;
}
vpn-yellow {
    vrf-table-label;
}
vpn-green {
    vrf-table-label;
}
```

```
[edit class-of-service routing-instances]
user@host# show
```

```
vpn* {
    classifiers {
        exp exp-class-wildcard;
    }
}
vpn-red {
    classifiers {
        exp exp-class-red;
    }
}
```

Verify the Classifiers Associated with Routing Instances

IN THIS SECTION

Purpose

Display the MPLS EXP classifiers associated with two routing instances:

Action

To verify the MPLS EXP classifiers associated with two routing instances, enter the following Junos CLI operational mode command:

```
user@host> show class-of-service routing-instances
  Routing Instance : vpn1
    Object          Name                   Type                    Index
    Classifier      exp-default            exp                         8

  Routing Instance : vpn2
    Object          Name                   Type                    Index
    Classifier      class2                 exp                     57507
```

Platform-Specific Behavior

Use the following table to review platform-specific behaviors for your platforms.

Table 14: Platform-Specific Behavior

Platform	Difference
MX Series devices	• On MX Series devices only, you can maintain the original MPLS EXP classifier.

RELATED DOCUMENTATION

Configure Behavior Aggregate Classifiers | 80

Default MPLS EXP Classifier | 66

Apply MPLS EXP Classifiers for Explicit-Null Labels

When you configure MPLS explicit-null labels, label 0 is advertised to the egress router of an LSP. When label 0 is advertised, the egress router (instead of the penultimate router) removes the label. Ultimate-hop popping (UHP) ensures that any packets traversing an MPLS network include a label. For more information about explicit-null labels and UHP, see the MPLS Applications User Guide.

When you configure MPLS explicit-null labels with an MPLS EXP classifier, the MPLS EXP classifier can be different from an IPv4 or IPv6 classifier configured on the same logical interface. In other words, you can apply separate classifiers for MPLS EXP, IPv4, and IPv6 packets per logical interface.

Configure an MPLS EXP classifier for explicit-null labels:

1. Create the MPLS EXP classifier.

```
[edit]
user@host# edit class-of-service classifiers exp classifier-name
```

2. Specify the name of a predefined classifier to include in this configuration.

```
[edit class-of-service classifiers exp classifier-name]
user@host# set import classifier-name
```

3. Define a classification of code point aliases for the classifier.

```
[edit class-of-service classifiers exp classifier-name]
user@host# set forwarding-class class-name loss-priority level code-points value
```

Apply the MPLS EXP classifier to the logical interface:

1. Specify the physical and logical interface names on which you want to apply the classifier.

```
[edit]
user@host# edit class-of-service interfaces interface-name unit logical-unit-number
```

2. Specify the classifier type and name you want to apply to the interface.

```
[edit class-of-service classifiers interfaces interface-name unit logical-unit-number]
user@host# set classifiers exp classifier-name
```

NOTE: When a packet with a single label is received, if the label is an explicit-null label (0 or 2), the label is popped first, making the EXP information no longer available. The subsequent packet classification is based on the IPv4/IPv6 payload. To preserve the MPLS classification of the packet, set `[explicit-null-cos inet/inet6]` at the `[edit forwarding-options]` hierarchy level. This option makes the packet classification based on the MPLS EXP value rather than on the payload.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
18.1	Starting with Junos OS 18.1R1, PTX Series routers with third-generation FPCs (FPC3) support `[explicit-null-cos inet/inet6]` at the `[edit forwarding-options]` hierarchy level. This option makes the packet classification based on the MPLS EXP value rather than on the payload, thus preserving the MPLS classification of the packet.

RELATED DOCUMENTATION

Configure Behavior Aggregate Classifiers | 80

Default MPLS EXP Classifier | 66

Apply MPLS EXP Classifiers to Routing Instances | 116

Manage Ingress Oversubscription with Traffic Class Maps

SUMMARY

Ingress oversubscription is a state where the transmission rate of the incoming packets is much higher than the rate that the Packet Forwarding Engine and router can handle, causing important packets to be dropped. An oversubscribed link or service experiencing an excess of traffic can result in traffic loss or delay that can potentially affect other services and links.

IN THIS SECTION

Ingress Oversubscription at the Packet Forwarding Engine

The Packet Forwarding Engine uses fixed rules to decide the priority of incoming packets. Based on these fixed rules, the Packet Forwarding Engine categorizes incoming packets into *high-priority network control* packets and *low-priority best-effort* packets. Packets with protocols such as routing protocols are classified as *network control* packets. Packets with protocols such as Telnet, FTP, and SSH are classified as *best-effort* packets.

The limitation of these fixed rules is that even if the trusted and non-network-control packets marked by a CE router are forwarded to the transit router, the transit router might drop these packets. This packet drop happens because, according to the fixed rules, none of these packets are high-priority packets for the transit router.

To overcome this limitation, you can prioritize and classify the traffic entering a Packet Forwarding Engine by configuring a traffic class map based on CoS values and associating the values with a traffic class such as `real-time`, `network control`, or `best-effort`. You can associate the traffic class map with an interface on the transit router. During ingress oversubscription, the router interface uses this user-defined traffic class map to select the packet priority.

NOTE: Use Feature Explorer to confirm platform and release support for traffic class maps.

Configure Traffic Class Maps

To configure a traffic class map:

1. Configure the interface. You will associate this interface with the configured traffic class maps.

```
[edit]
user@host# set interfaces interface-name unit unit-number family family-name address address
```

2. Create a traffic class map based on CoS code points, and map the code points to a traffic class to decide the input packet priority.

   - To create a DSCP traffic class map and map the code points to a traffic class for IPv4 and IPv6 traffic, include the following statements at the `[edit class-of-service]` hierarchy level.

   ```
   [edit class-of-service]
   user@host# set traffic-class-map dscp traffic-class-map-name traffic-class best-effort
   code-points code-point-value

   user@host# set traffic-class-map dscp traffic-class-map-name traffic-class network-control
   code-points code-point-value

   user@host# set traffic-class-map dscp traffic-class-map-name traffic-class real-time code-
   points code-point-value
   ```

   - To create an IEEE 802.1 traffic class map and map the code points to a traffic class, include the following statements at the `[edit class-of-service]` hierarchy level.

   ```
   [edit class-of-service]
   user@host# set traffic-class-map ieee-802.1 traffic-class-map-name traffic-class best-
   effort code-points code-point-value

   user@host# set traffic-class-map ieee-802.1 traffic-class-map-name traffic-class network-
   control code-points code-point-value

   user@host# set traffic-class-map ieee-802.1 traffic-class-map-name traffic-class real-time
   code-points code-point-value
   ```

   - To create an MPLS EXP traffic class map and map the code points to a traffic class, include the following statements at the `[edit class-of-service]` hierarchy level.

   ```
   [edit class-of-service]
   user@host# set traffic-class-map exp traffic-class-map-name traffic-class best-effort code-
   ```

```
points code-point-value

user@host# set traffic-class-map exp traffic-class-map-name traffic-class network-control
code-points code-point-value

user@host# set traffic-class-map exp traffic-class-map-name traffic-class real-time code-
points code-point-value
```

- To create an IPv4 precedence traffic class map and map the code points to a traffic class, include the following statements at the `[edit class-of-service]` hierarchy level.

```
[edit class-of-service]
user@host# set traffic-class-map inet-precedence traffic-class-map-name traffic-class best-
effort code-points code-point-value

user@host# set traffic-class-map inet-precedence traffic-class-map-name traffic-class
network-control code-points code-point-value

user@host# set traffic-class-map inet-precedence traffic-class-map-name traffic-class real-
time code-points code-point-value
```

- To create an IEEE 802.1ad code point traffic class map and map the code points to a traffic class, include the following statements at the `[edit class-of-service]` hierarchy level.

```
[edit class-of-service]
user@host# set traffic-class-map ieee-802.1ad traffic-class-map-name traffic-class best-
effort code-points code-point-value
user@host# set traffic-class-map ieee-802.1ad traffic-class-map-name traffic-class network-
control code-points code-point-value

user@host# set traffic-class-map ieee-802.1ad traffic-class-map-name traffic-class real-
time code-points code-point-value
```

3. Associate the traffic class map with the interface that you configured in Step 1.

- Associate the DSCP traffic class map with the interface.

```
[edit class-of-service]
user@host# set interfaces interface-name traffic-class-map dscp traffic-class-map-name
```

- Associate the IEEE 802.1 traffic class map with the interface.

```
[edit class-of-service]
user@host# set interfaces interface-name traffic-class-map ieee-802.1 traffic-class-map-
name  <vlan-tag (inner | outer)>
```

- Associate the MPLS EXP traffic class map with the interface.

```
[edit class-of-service]
user@host# set interfaces interface-name traffic-class-map exp traffic-class-map-name
```

- Associate the IPv4 precedence traffic class map with the interface.

```
[edit class-of-service]
user@host# set interfaces interface-name traffic-class-map inet-precedence traffic-class-
map-name
```

- Associate the IEEE 802.1ad traffic class map with the interface.

```
[edit class-of-service]
user@host# set interfaces interface-name traffic-class-map ieee-802.1ad traffic-class-map-
name <vlan-tag (inner | outer)>
```

NOTE:

- If you do not associate the traffic class map with the configured interface, Junos treats all traffic through this interface with the existing fixed rule in the Packet Forwarding Engine. The existing fixed rule prioritizes network control traffic over best-effort traffic.
- As soon as you associate a traffic class map with an interface, any code points entering that interface and not included in the traffic class map are treated as BE.
- You can associate either an IPv4 precedence traffic class map or a DSCP traffic class map with an interface. You cannot associate both these traffic class maps with a single interface. The DSCP traffic class map applies to both IPv4 and IPv6 traffic.
- You can associate either an IEEE 802.1 traffic class map or an IEEE 802.1ad traffic class map with an interface. You cannot associate both these traffic class maps with a single interface.
- An aex bundle can have member links from both interfaces that support traffic class maps and interfaces that do not. A configured traffic class map is associated with an aggregated Ethernet bundle in following ways:
  - If an aggregated Ethernet bundle has child links only from interfaces that support traffic class maps, then the traffic class map is associated with all links of the aggregated Ethernet bundle.
  - If an aggregated Ethernet bundle has child links only from interfaces that do not support traffic class maps, then the traffic class map is not associated with the aggregated Ethernet bundle or its links.
  - If an aex has child links from interfaces that support traffic class maps and also interfaces that do not, the traffic class map associates only with the links from the interfaces that support traffic class maps.

Example: Configure Traffic Class Maps

IN THIS SECTION

This example shows the configuration of traffic class maps on an MX Series routers with MPCs.

NOTE: Use Feature Explorer to confirm platform and release support for traffic class maps.

Requirements

This example uses the following hardware and software components:

- One MX Series router
- One CE router

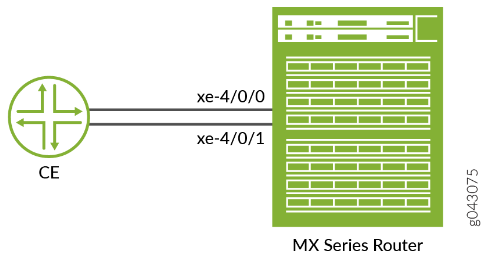
Before you configure the traffic class maps, be sure you have:

- Connected the CE router to the MX Series router.

Overview and Topology

This example shows the configuration of traffic class maps on an MX Series router that is connected to a CE router.

Figure 14: Configuring Traffic Class Maps on MX Series router



As shown in Figure 14 on page 131, the CE router forwards the traffic to interface xe-4/0/0 and xe-4/0/1 on the router.

The traffic class maps need to be configured on the router and associated with the interface xe-4/0/0 and the interface xe-4/0/1 so that the packets can be classified based on the user-defined configuration. When ingress oversubscription occurs, the router uses the user-defined traffic class map to process the packets.

This example shows how to create the following traffic class maps with CoS code points and associate these code points with the traffic class.

- IPv4 precedence traffic class map with code points 000 001, 010 011, and 100 101. Map these code points to the real-time, network-control, and best-effort traffic classes, respectively.
- MPLS EXP traffic class map with code points 000 001, 010 011, and 100 101. Map these code points to the real-time, network-control, and best-effort traffic classes, respectively.
- IEEE 802.1 traffic class map with code points 000 001, 010 011, and 100 101. Map these code points to the real-time, network-control, and best-effort traffic classes, respectively.
- DSCP traffic class map with code points 100001 100010 100011, 010011 010100 010101, and 101001 101010 101011. Map these code points to the real-time, network-control, and best-effort traffic classes, respectively.
- IEEE 802.1ad traffic class map with code points 0000 0001 1000 1001, 0010 0011 1010 1011, and 0100 0101 1100 1101. Map these code points to the real-time, network-control, and best-effort traffic classes, respectively.

The traffic class maps IPv4 precedence, MPLS EXP, and IEEE 802.1 are associated with the interface xe-4/0/0. The traffic class maps DSCP and IEEE 802.1ad are associated with the interface xe-4/0/1.

Configuration

IN THIS SECTION

To configure the traffic class map, perform the following tasks:

CLI Quick Configuration

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

```
[edit]
set interfaces xe-4/0/0 unit 0 family inet address 198.51.100.0/24
set interfaces xe-4/0/1 vlan-tagging
set interfaces xe-4/0/1 unit 0 vlan-id 111
set interfaces xe-4/0/1 unit 0 family inet address 198.51.100.1/24
set class-of-service traffic-class-map inet-precedence inetp traffic-class real-time code-points
[ 000 001 ]
set class-of-service traffic-class-map inet-precedence inetp traffic-class network-control code-
points [ 010 011 ]
set class-of-service traffic-class-map inet-precedence inetp traffic-class best-effort code-
points [ 100 101 ]
set class-of-service traffic-class-map exp mpls_exp traffic-class real-time code-points [ 000
001 ]
set class-of-service traffic-class-map exp mpls_exp traffic-class network-control code-points
[ 010 011 ]
set class-of-service traffic-class-map exp mpls_exp traffic-class best-effort code-points [ 100
101 ]
set class-of-service traffic-class-map ieee-802.1 802.1p traffic-class real-time code-points
[ 000 001 ]
set class-of-service traffic-class-map ieee-802.1 802.1p traffic-class network-control code-
points [ 010 011 ]
set class-of-service traffic-class-map ieee-802.1 802.1p traffic-class best-effort code-points
[ 100 101 ]
set class-of-service traffic-class-map dscp dscp_v4 traffic-class real-time code-points [ 100001
100010 100011 ]
set class-of-service traffic-class-map dscp dscp_v4 traffic-class network-control code-points
[ 010011 010100 010101 ]
set class-of-service traffic-class-map dscp dscp_v4 traffic-class best-effort code-points
[ 101001 101010 101011 ]
set class-of-service traffic-class-map ieee-802.1ad 802.1ad traffic-class real-time code-points
[ 0000 0001 1000 1001 ]
set class-of-service traffic-class-map ieee-802.1ad 802.1ad traffic-class network-control code-
points [ 0010 0011 1010 1011 ]
set class-of-service traffic-class-map ieee-802.1ad 802.1ad traffic-class best-effort code-
points [ 0100 0101 1100 1101 ]
set interfaces xe-4/0/0 traffic-class-map inet-precedence inetp
set interfaces xe-4/0/0 traffic-class-map exp mpls_exp
set interfaces xe-4/0/0 traffic-class-map  ieee-802.1 802.1p vlan-tag inner
set interfaces xe-4/0/1 traffic-class-map dscp dscp_v4
set interfaces xe-4/0/1 traffic-class-map ieee-802.1ad 802.1ad vlan-tag inner
```

Configuring Interfaces

Step-by-Step Procedure

Configure the interfaces. These interfaces need to be associated with traffic class maps.

1. Configure the interface xe-4/0/0 with unit 0 as its logical interface, inet as protocol family, and 198.51.100.0/24 as the IP address.

   ```
   [edit]
   user@host#set interfaces xe-4/0/0 unit 0  family inet address 198.51.100.0/24
   ```

2. Configure the interface xe-4/0/1 with unit 0 as its logical interface, inet as protocol family, and 198.51.100.1/24 as the IP address. Also, enable the VLAN tagging and configure a VLAN ID (for example, 111) to receive and transmit VLAN-tagged frames on the interface.

   ```
   [edit]
   user@host#set interfaces xe-4/0/1 vlan-tagging
   user@host#set interfaces xe-4/0/1 unit 0 vlan-id 111
   user@host#set interfaces xe-4/0/1 unit 0 family inet address 198.51.100.1/24
   ```

Configuring Traffic Class Maps for the Code Points and Mapping the Code Points to a Traffic Class

Step-by-Step Procedure

You can prioritize and classify the traffic entering a Packet Forwarding Engine by configuring a traffic class map based on the code points and associating the map with the traffic class.

1. Create an IPv4 precedence traffic class map inetp and map its code points 000 001, 010 011, and 100 101 to the real-time, network control, and best-effort traffic classes, respectively.

   ```
   [edit class-of-service]
   user@host# set traffic-class-map inet-precedence inetp traffic-class real-time code-points
   [ 000 001 ]
   ```

```
user@host# set traffic-class-map inet-precedence inetp traffic-class network-control code-
points [ 010 011 ]
user@host# set traffic-class-map inet-precedence inetp traffic-class best-effort code-points
[ 100 101 ]
```

2. Create an MPLS EXP traffic class map mpls_exp and map the code points 000 001, 010 011, and 100 101 to the real-time, network control, and best-effort traffic classes, respectively.

```
[edit class-of-service]
user@host# set traffic-class-map exp mpls_exp traffic-class real-time code-points [ 000 001 ]
user@host# set traffic-class-map exp mpls_exp traffic-class network-control code-points [ 010
011 ]
user@host# set traffic-class-map exp mpls_exp traffic-class best-effort code-points [ 100
101 ]
```

3. Create an IEEE 802.1 traffic class map 802.1p and map the code points 000 001, 010 011, and 100 101 to the real-time, network control, and best-effort traffic classes, respectively.

```
[edit class-of-service]
user@host# set traffic-class-map ieee-802.1 802.1p traffic-class real-time code-points [ 000
001 ]
user@host# set traffic-class-map ieee-802.1 802.1p traffic-class network-control code-points
[ 010 011 ]
user@host# set traffic-class-map ieee-802.1 802.1p traffic-class best-effort code-points
[ 100 101 ]
```

4. Create a DSCP traffic class map dscp_v4 and map the code points 100001 100010 100011, 010011 010100 010101, and 101001 101010 101011 to the real-time, network control, and best-effort traffic classes, respectively.

```
[edit class-of-service]
user@host# set traffic-class-map dscp dscp_v4 traffic-class real-time code-points [ 100001
100010 100011 ]
user@host# set traffic-class-map dscp dscp_v4 traffic-class network-control code-points
[ 010011 010100 010101 ]
user@host# set traffic-class-map dscp dscp_v4 traffic-class best-effort code-points [ 101001
101010 101011 ]
```

5. Create an IEEE802.1ad traffic class map 802.1ad and map the code points 0000 0001 1000 1001,0010 0011 1010 1011, and 0100 0101 1100 1101 to the real-time, network control, and best-effort traffic classes, respectively.

```
[edit class-of-service]
user@host# set traffic-class-map ieee-802.1ad 802.1ad traffic-class real-time code-points
[ 0000 0001 1000 1001 ]
user@host# set traffic-class-map ieee-802.1ad 802.1ad traffic-class network-control code-
points [ 0010 0011 1010 1011 ]
user@host# set traffic-class-map ieee-802.1ad 802.1ad traffic-class best-effort code-points
[ 0100 0101 1100 1101 ]
```

Associating Interfaces with Traffic Class Maps

Step-by-Step Procedure

You need to associate the configured traffic class maps with the interfaces on which you want to prioritize and classify the input traffic.

1. Associate the traffic class maps inetp, mpls_exp, and 802.1p with the interface xe-4/0/0.

```
[edit class-of-service]
user@host# set interfaces xe-4/0/0 traffic-class-map inet-precedence inetp
user@host# set interfaces xe-4/0/0 traffic-class-map exp mpls_exp
user@host# set interfaces xe-4/0/0 traffic-class-map  ieee-802.1 802.1p vlan-tag inner
```

2. Associate the traffic class map dscp_v4 and 802.1ad with the interface xe-4/0/1.

```
[edit class-of-service]
user@host# set interfaces xe-4/0/1 traffic-class-map dscp dscp_v4
user@host# set interfaces xe-4/0/1 traffic-class-map ieee-802.1ad 802.1ad vlan-tag inner
```

Results

```
interfaces {
    xe-4/0/0 {
        unit 0 {
            family inet {
                address 198.51.100.0/24;
            }
        }
    }
    xe-4/0/1 {
        vlan-tagging;
        unit 0 {
            vlan-id 111;
            family inet {
                address 198.51.100.1/24;
            }
        }
    }
}
class-of-service {
    traffic-class-map {
        inet-precedence inetp {
            traffic-class real-time code-points [ 000 001 ];
            traffic-class network-control code-points [ 010 011 ];
            traffic-class best-effort code-points [ 100 101 ];
        }
        dscp dscp_v4 {
            traffic-class real-time code-points [ 100001 100010 100011 ];
            traffic-class network-control code-points [ 010011 010100 010101 ];
            traffic-class best-effort code-points [ 101001 101010 101011 ];
        }
        exp mpls_exp {
            traffic-class real-time code-points [ 000 001 ];
            traffic-class network-control code-points [ 010 011 ];
            traffic-class best-effort code-points [ 100 101 ];
        }
        ieee-802.1 802.1p {
            traffic-class real-time code-points [ 000 001 ];
            traffic-class network-control code-points [ 010 011 ];
            traffic-class best-effort code-points [ 100 101 ];
        }
        ieee-802.1ad 802.1ad {
            traffic-class real-time code-points [ 0000 0001 1000 1001 ];
            traffic-class network-control code-points [ 0010 0011 1010 1011 ];
            traffic-class best-effort code-points [ 0100 0101 1100 1101 ];
        }
    }
    interfaces {
        xe-4/0/0 {
            traffic-class-map {
                inet-precedence inetp;
                exp mpls_exp;
                ieee-802.1 802.1p vlan-tag inner;
            }
        }
        xe-4/0/1 {
            traffic-class-map {
                dscp dscp_v4;
                ieee-802.1ad 802.1ad vlan-tag inner;
            }
        }
    }
}
```

Verification

IN THIS SECTION

Verify Mapping of Code Points to Input Traffic Classes

Purpose

Verify that the code points of traffic class maps are mapped to the corresponding traffic classes.

Action

In operational mode, enter the `show class-of-service traffic-class-map` command.

```
user@host> show class-of-service traffic-class-map

Traffic-class-map: inetp, Code-point type: inet-precedence, Index: 43854
```

```
  Code point         Traffic class
  000                real-time
  001                real-time
  010                network-control
  011                network-control
  100                best-effort
  101                best-effort

Traffic-class-map: dscp_v4, Code-point type: dscp, Index: 37469
  Code point         Traffic class
  010011             network-control
  010100             network-control
  010101             network-control
  100001             real-time
  100010             real-time
  100011             real-time
  101001             best-effort
  101010             best-effort
  101011             best-effort

Traffic-class-map: mpls_exp, Code-point type: exp, Index: 39622
  Code point         Traffic class
  000                real-time
  001                real-time
  010                network-control
  011                network-control
  100                best-effort
  101                best-effort

Traffic-class-map: 802.1p, Code-point type: ieee-802.1, Index: 13605
  Code point         Traffic class
  000                real-time
  001                real-time
  010                network-control
  011                network-control
  100                best-effort
  101                best-effort

Traffic-class-map: 802.1ad, Code-point type: ieee-802.1ad, Index: 13677
  Code point         Traffic class
  0000               real-time
  0001               real-time
  0010               network-control
  0011                  network-control
  0100                  best-effort
  0101                  best-effort
  1000                  real-time
  1001                  real-time
  1010                  network-control
  1011                  network-control
  1100                  best-effort
  1101                  best-effort
```

Meaning

The display output fields `Traffic-class-map` and `Code-point type` indicate the configured traffic class map and the type of code point information, respectively.

The fields `Code point` and `Traffic class` show the mapping between the code points and the traffic class.

Verifying Mapping of Interfaces to Traffic Class Maps

Purpose

Verify that the configured interfaces are mapped to the corresponding traffic class maps.

Action

In operational mode, enter the `show class-of-service forwarding-table traffic-class-map mapping` command.

```
user@host> show class-of-service forwarding-table traffic-class-map mapping

Interface      Index    Table Index   Table type
xe-4/0/0         162        43854     INET-Precedence
                            39622     MPLS EXP
                            13605     IEEE-802.1
xe-4/0/1         163        37469     DSCP
                            13677     IEEE-802.1AD
```

Meaning

The output shows that:

- Interface `xe-4/0/0` is associated with the traffic class maps `INET-Precedence`, `MPLS EXP`, and `IEEE-802.1`.

- Interface `xe-4/0/1` is associated with the traffic class maps `DSCP` and `IEEE-802.1AD`.

Verifying Traffic Class Information on the Interface

Purpose

Verify the packet information based on the configured traffic class map.

Action

In operational mode, enter the `show interfaces xe-4/0/0 extensive` and `show interfaces xe-4/0/1 extensive` commands.

```
user@host> show interfaces xe-4/0/0 extensive

Physical interface: xe-4/0/0, Enabled, Physical link is Up
  Interface index: 162, SNMP ifIndex: 541, Generation: 165
  Link-level type: Ethernet, MTU: 1518, MRU: 0, LAN-PHY mode, Speed: 10Gbps, BPDU Error: None,
MAC-REWRITE Error: None, Loopback: None, Source filtering: Disabled,
  Flow control: Enabled
  ...
  Preclassifier statistics:
    Traffic Class        Received Packets   Transmitted  Packets      Dropped Packets
    real-time                      3000              3000                0
    network-control                2000              2000                0
    best-effort                    2000              1000                1000
  Interface transmit statistics: Enabled

  ...
user@host> show interfaces xe-4/0/1 extensive

Physical interface: xe-4/0/1, Enabled, Physical link is Up
  Interface index: 163, SNMP ifIndex: 525, Generation: 166
  Link-level type: Ethernet, MTU: 1518, MRU: 0, LAN-PHY mode, Speed: 10Gbps, BPDU Error: None,
MAC-REWRITE Error: None, Loopback: None, Source filtering: Disabled,
  Flow control: Enabled
  ...
  Preclassifier statistics:
    Traffic Class        Received Packets   Transmitted  Packets      Dropped Packets
    real-time                      2000              2000                0
    network-control                1000              1000                0
    best-effort                    1000              600                400
```

```
Interface transmit statistics: Enabled

  ...
```

Meaning

The `Preclassifier statistics` field shows the information for received, transmitted, and dropped packets for each of the configured traffic class map.

RELATED DOCUMENTATION

show class-of-service forwarding-table traffic-class-map

show class-of-service traffic-class-map

Configure ToS Translation Tables

On some PICs, the behavior aggregate (BA) translation tables are included for every *logical interface* (unit) protocol family configured on the logical interface. The proper default translation table is active even if you do not include any explicit translation tables. You can display the current translation table values with the `show class-of-service classifiers` command.

You can configure a physical interface (port) or logical interface (unit) with up to three translation tables. For example, you can configure a port or unit with BA classification for IPv4 DSCP, IPv6 DSCP, and MPLS EXP. The number of frame relay data-link connection identifiers (DLCIs) (units) that you can configure on each PIC varies based on the number and type of BA classification tables configured on the interfaces.

You can replace the ToS bit value on the incoming packet header on a logical interface with a user-defined value. The new ToS value is used for all class-of-service processing and is applied before any other class-of-service or firewall treatment of the packet. The values you configure with the `translation-table` statement at the `[edit class-of-service]` hierarchy level determines the new ToS bit values.

Four types of translation tables are supported: IP precedence, IPv4 DSCP, IPv6 DSCP, and MPLS EXP. You can configure a maximum of eight tables for each supported type. If a translation table is enabled for a particular type of traffic, then behavior aggregate (BA) classification of the same type must be configured for that logical interface. In other words, if you configure an IPv4 translation table, you must configure IPv4 BA classification on the same logical interface.

The `from-code-points` statement establishes the values to match on the incoming packets. The `default` option is used to match all values not explicitly listed, and, as a single entry in the translation table, to mark all incoming packets on an interface the same way. The `to-code-point` statement establishes the target values for the translation. If an incoming packet header ToS bit configuration is not covered by the translation table list and a `*` option is not specified, the ToS bits in the incoming packet header are left unchanged.

You can define many translation tables, as long as they have distinct names. You apply a translation table to a logical interface at the `[edit class-of-service interfaces]` hierarchy level. Translation tables always translate "like to like." For example, a translation table applied to MPLS traffic can only translate from received EXP bit values to new EXP bit values. That is, translation tables cannot translate (for instance) from DSCP bits to INET precedence code points.

Incoming ToS bit translation is subject to the following rules:

- Locally generated traffic is not subject to translation.
- The `to-dscp-from-dscp` translation table type is not supported if an Internet precedence classifier is configured.
- The `to-inet-precedence-from-inet-precedence` translation table type is not supported if a DSCP classifier is configured.
- The `to-dscp-from-dscp` and `to-inet-precedence-from-inet-precedence` translation table types cannot be configured on the same unit.
- The `to-dscp-from-dscp` and `to-inet-precedence-from-inet-precedence` translation table types are supported for IPv4 packets.
- Only the `to-dscp-ipv6-from-dscp-ipv6` translation table type is supported for IPv6 packets.
- Only the `to-exp-from-exp` translation table type is supported for MPLS packets.

NOTE: Translation tables are not supported if fixed classification is configured on the logical interface.

A maximum of 32 distinct translation tables are supported on each PIC. However, this maximum is limited by the number of classifiers configured along with translation tables because on the PIC the hardware tables are not always merged. For example, if a translation table and a classifier are both configured on the same logical interface (such as `unit 0`), there is only one hardware table and only one table added to the 32 translation table limit. However, if the translation table is configured on `unit 0` and the classifier on `unit 1` on the same physical interface, then two hardware tables are used and these two tables count toward the 32 maximum.

If you try to configure mutually exclusive translation tables on the same interface unit, you will get a warning message when you display or commit the configuration:

```
ge-0/1/1 {
    unit 0 {
        translation-table {
            ##
            ## Warning: to-dscp-from-dscp and to-inet-precedence-from-inet-precedence not
allowed on same unit
            ##
            to-inet-precedence-from-inet-precedence inet-trans-table;
            to-dscp-from-dscp dscp-trans-table;
        }
    }
}
```

ToS translation is a form of behavior aggregate (BA) classification.

To configure ToS translation , include the `translation-table` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
translation-table {
    (to-dscp-from-dscp | to-dscp-ipv6-from-dscp-ipv6 | to-exp-from-exp | to-inet-precedence-from-
inet-precedence) table-name {
        to-code-point value from-code-points (* | [ values ]);
    }
}
```

The following example procedure translates incoming DSCP values to the new values listed in the table. All incoming DSCP values other than `111111`, `111110`, `000111`, and `100111` are translated to `000111`:

1. Create and configure the translation table.

   ```
   [edit class-of-service]
   user@host# set translation-table to-dscp-from-dscp dscp-trans-table to-code-point 000000 from-
   code-points 111111
   user@host# set translation-table to-dscp-from-dscp dscp-trans-table to-code-point 000001 from-
   code-points 111110
   user@host# set translation-table to-dscp-from-dscp dscp-trans-table to-code-point 111000 from-
   code-points [ 000111 100111 ]
   ```

```
user@host# set translation-table to-dscp-from-dscp dscp-trans-table to-code-point 000111 from-
code-points *
```

2. Apply the translation table to the logical interface input.

```
[edit class-of-service]
user@host# set interfaces so-1/0/0 unit 0 translation-table to-dscp-from-dscp dscp-trans
```

3. Verify the configuration.

   - To verify that the correct values are configured, use the `show class-of-service translation-table` command. The `show class-of-service translation-table` command displays the code points of all translation tables configured. All values are displayed, not just those configured:

```
user@host> show class-of-service translation-table
Translation Table: dscp-trans-table, Translation table type: dscp-to-dscp, Index: 6761
  From Code point    To Code Point
  000000                      000111
  000001                      000111
  000010                      000111
  000011                      000111
  000100                      000111
  000101                      000111
  000110                      000111
  000111                      111000
  001000                      000111
  001001                      000111
  001010                      000111
  001011                      000111
  001100                      000111
  001101                      000111
  001110                      000111
  001111                      000111
  010000                      000111
  010001                      000111
  010010                      000111
  010011                      000111
  010100                      000111
  010101                      000111
  010110                      000111
  010111                      000111
  011000                      000111
```

```
011001                      000111
011010                      000111
011011                      000111
011100                      000111
011101                      000111
011110                      000111
011111                      000111
100000                      000111
100001                      000111
100010                      000111
100011                      000111
100100                      000111
100101                      000111
100110                      000111
100111                      111000
101000                      000111
101001                      000111
101010                      000111
101011                      000111
101100                      000111
101101                      000111
101110                      000111
101111                      000111
110000                      000111
110001                      000111
110010                      000111
110011                      000111
110100                      000111
110101                      000111
110110                      000111
110111                      000111
111000                      000111
111001                      000111
111010                      000111
111011                      000111
111100                      000111
111101                      000111
111110                      000001
111111                      000000
```

- To verify that the configured translation table is applied to the correct interface, use the `show class-of-service interface` *`interface-name`* command. The `show class-of-service interface` *`interface-name`* command displays the translation tables applied to the interface:

```
user@host> show class-of-service interface ge-0/1/1
Physical interface: ge-0/1/1, Index: 156  From Code point    To Code Point
  Queues supported: 4, Queues in use: 4
    Scheduler map: <default>, Index: 2
    Chassis scheduler map: <default-chassis>, Index: 4

  Logical interface: so-2/3/0.0, Index: 68
    Object                  Name                  Type                    Index
    Rewrite                 exp-default           exp (mpls-any)             29
    Classifier              dscp-default          dscp                        7
    Classifier              exp-default           exp                        10
    Translation Table       exp-trans-table       EXP_TO_EXP              61925
```

4. Save the configuration.

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

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 60

Assign CoS Levels to Packets Based on Multiple Packet Header Fields | 148

CHAPTER 3

Assign Service Levels with Multifield Classifiers

IN THIS CHAPTER

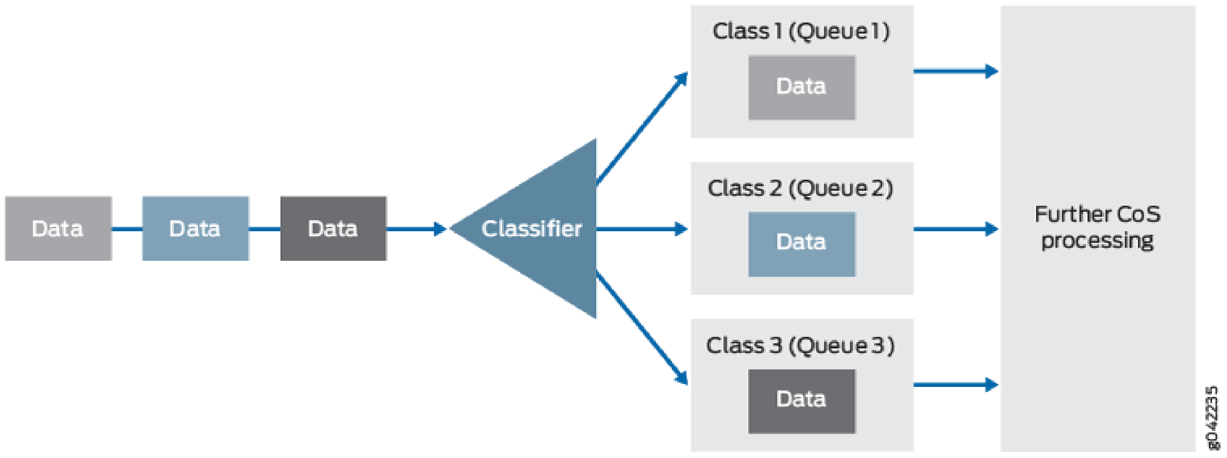
Assign CoS Levels to Packets Based on Multiple Packet Header Fields

Behavior aggregate (BA) classification (see "Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic" on page 60), classifies packets based on their QoS markings. BA classification is the most common way to assign service levels, because it is straightforward and based on a well-established, fixed-length header fields, which makes BA classifiers computationally more efficient. However, sometimes BA classification does not provide sufficient granularity, or the QoS markings in the packet headers cannot be trusted. In such situations, multifield classifiers can be used. A multifield classifier is a method of classifying traffic flows based on multiple packet header fields. Devices that sit at the edge of a network usually classify packets based on multiple packet header fields. Multifield classification is normally performed at the network edge because of the general lack of 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 header fields to determine the forwarding class for a packet. Typically, a classifier performs matching operations on the selected fields against a configured value. A multifield classifier can examine multiple fields in the packet header: destination address, source address, IP protocol, source port, destination port, and DSCP value. Multifield classifiers are used when a simple BA classifier is insufficient to classify a packet.

Figure 15 on page 149 provides a high-level illustration of how a classifier works.

Figure 15: How a Classifier Works



In Junos, you configure a multifield classifier with a *firewall filter* and its associated match conditions. Multifield classification 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.

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | **60**

Configure Multifield Classifiers | **150**

Configure Multifield Classifiers

This topic describes how to configure multifield classifiers.

Multifield classifiers classify packets to a forwarding class and loss priority based on firewall filter match criteria. You usually use multifield classification at the edge of the network for packets that do not have valid or trusted BA code points.

If you configure both a BA classifier and a multifield classifier, BA classification is performed first; then multifield classification is performed. If both a BA classifier and a multifield classifier conflict, the multifield classifier overrides the BA classifier.

NOTE: For a specified interface, you can configure both a multifield classifier and a BA classifier without conflicts. Because Junos applies the classifiers in sequential order, the BA classifier followed by the multifield classifier, the multifield classifier overrides the BA classifier if the two classifiers conflict.

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

You configure multifield classifiers by:

1. **Defining the filter**—Configure *either* a firewall filter or a simple filter. Simple filters filter only IPv4 traffic (family inet). Firewall filters enable you to filter additional protocol families and more complex filters. The following sections describe both procedures.

2. **Applying the filter**—Activate the filter by configuring on a logical interface as an *input* filter.

To configure a firewall filter:

1. Under the `firewall` statement, specify the protocol family for which you want to filter traffic and specify a name for the filter.

```
edit
user@host# edit firewall family family-name filter filter-name
```

2. Specify the term name and match criteria you want to look for in incoming packets.

```
[edit firewall family family-name filter filter-name]
user@host# set term term-name from match-conditions
```

3. Specify the action you want to take when a packet matches the conditions.

```
[edit firewall family family-name filter filter-name]
user@host# set term term-name then actions
```

For multifield classifiers, you can perform the following actions:

- Set the value of the DSCP field of incoming packets.

```
user@host# set term term-name then dscp code-point
```

- Set the forwarding class of incoming packets. The forwarding class determines the output queue.

```
user@host# set term term-name then forwarding-class class-name
```

- Set the loss priority of incoming packets. The loss priority is used by schedulers in conjunction with the random early discard (RED) algorithm to control packet discard during periods of congestion.

```
user@host# set term term-name then loss-priority (high | low | medium-high | medium-low)
```

To configure a simple filter:

1. Specify a name for the simple filter.

```
[edit firewall family family-name]
user@host# edit simple-filter filter-name
```

2. Specify the term name and match criteria you want to look for in incoming packets.

```
[edit firewall family family-name simple-filter filter-name]
user@host# set term term-name from match-conditions
```

3. Specify the action you want to take when a packet matches the conditions.

```
[edit firewall family family-name simple-filter filter-name]
user@host# set term term-name then actions
```

For multifield classifiers, you can perform the following actions for a simple filter:

- Set the *forwarding-class* of incoming packets.
- Set the*loss-priority* of incoming packets.

To apply the firewall filter to the appropriate logical interfaces as an input filter.

1. Specify the physical and logical interface on which you want to apply the firewall filter.

```
edit
user@host# edit interfaces interface-name unit unit-number
```

2. Specify the protocol family for the firewall filter.

```
[edit interfaces interface-name unit unit-number]
user@host# set family family-name
```

3. Specify the names of the firewall filters to apply to received packets.

```
[edit interfaces interface-name unit unit-number]
user@host# set filter input filter-name
```

   Repeat this step for the family protocol filter and the simple filter.

4. Save your configuration.

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

Platform-Specific Multifield Classifier Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behaviors for your platform:

Platform	Difference
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.
MX Series	• 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.

RELATED DOCUMENTATION

Assign CoS Levels to Packets Based on Multiple Packet Header Fields | **148**

Use Multifield Classifiers to Set Packet Loss Priority | **153**

Use Multifield Classifiers to Set Packet Loss Priority

This topic describes how to configure multifield classifiers to set the packet loss priority (PLP) of incoming or outgoing packets.

Multifield classifiers take action on incoming or outgoing packets, depending on whether the firewall rule is applied as an input filter or an output filter.

To configure the PLP for a multifield classifier, include the `loss-priority` statement in a policer or firewall filter that you configure at the at the `[edit firewall]` hierarchy level:

The inputs (match conditions) for a multifield classifier are one or more of the six packet header fields: destination address, source address, IP protocol, source port, destination port, and DSCP. The outputs for a multifield classifier are the forwarding class and the PLP. A multifield classifier sets the forwarding class and the PLP for each packet entering or exiting the interface with a specific destination address, source address, IP protocol, source port, destination port, or DSCP.

In the following sample procedure, the forwarding class `expedited-forwarding` and PLP `medium-high` are assigned to all IPv4 packets with the `10.1.1.0/24` or `10.1.2.0/24` source address.

To use the classifier in this sample procedure, you must configure the settings for the `expedited-forwarding` forwarding class at the `[edit class-of-service forwarding-classes queue` *`queue-number`* `expedited-forwarding]` hierarchy level. For more information, see "Understanding How Forwarding Classes Assign Classes to Output Queues" on page 286.

1. Under the `firewall` statement, specify the protocol family as IPv4 (inet) and specify a name for the filter.

```
edit
user@host# edit firewall family inet filter classify-customers
```

2. Specify the term name and match criteria you want to look for in incoming packets.

```
[edit firewall family inet filter classify-customers]
user@host# set term isp1-customers from source-address 10.1.1.0/24
user@host# set term isp1-customers from source-address 10.1.2.0/24
```

3. Specify the action you want to take when a packet matches the conditions.

```
[edit firewall family inet filter classify-customers]
user@host# set term isp1-customers then loss-priority medium-high
user@host# set term isp1-customers then forwarding-class medium-high
```

4. Verify your configuration.

```
[edit firewall]
user@host# show
```

```
 filter classify-customers {
        term isp1-customers {
            from {
                source-address {
                    10.1.1.0/24;
                    10.1.2.0/24;
                }
            }
            then {
                loss-priority medium-low;
                forwarding-class assured-forwarding;
            }
        }
    }
```

5. Save your configuration.

```
[edit firewall]
user@host# commit
```

RELATED DOCUMENTATION

Configure Multifield Classifiers | 150

Assign CoS Levels to Packets Based on Multiple Packet Header Fields | 148

Example: Configure and Apply a Firewall Filter for a Multifield Classifier

IN THIS SECTION

This example shows how to configure a firewall filter to classify traffic using a multifield classifier. The classifier detects packets of interest to CoS as the packets arrive on an interface. Use multifield classifiers when a simple BA classifier is insufficient to classify a packet, when peering routers do not have CoS bits marked, or when the peering router's marking is untrusted.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based, or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos. The example shown here was tested and verified on MX Series routers running Junos OS Release 10.4.

Overview

IN THIS SECTION

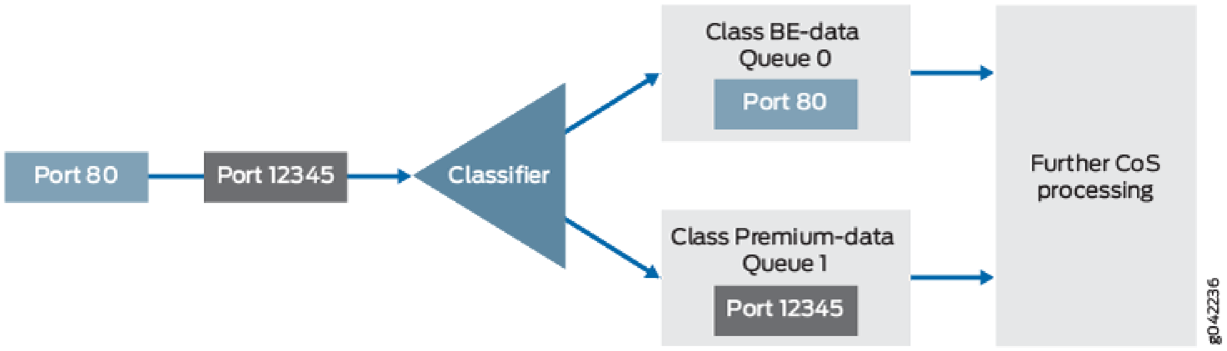
A classifier is a software operation that inspects a packet as it enters the router or switch. The packet header contents are examined, and this examination determines how the packet is treated when the network becomes too busy to handle all of the packets and you want your devices to drop packets intelligently, instead of dropping packets indiscriminately. One common way to detect packets of interest is by source port number. The TCP port numbers 80 and 12345 are used in this example, but many other matching criteria for packet detection are available to multifield classifiers, using firewall filter match conditions. The configuration in this example specifies that TCP packets with source port 80 are classified into the BE-data forwarding class and queue number 0. TCP packets with source port 12345 are classified into the Premium-data forwarding class and queue number 1.

You typically use multifield classifiers at the network edge as packets enter an autonomous system (AS).

In this example, you configure the firewall filter mf-classifier and specify some custom forwarding classes on Device R1. In specifying the custom forwarding classes, you also associate each class with a queue.

The classifier operation is shown in Figure 16 on page 157.

Figure 16: Multifield Classifier Based on TCP Source Ports

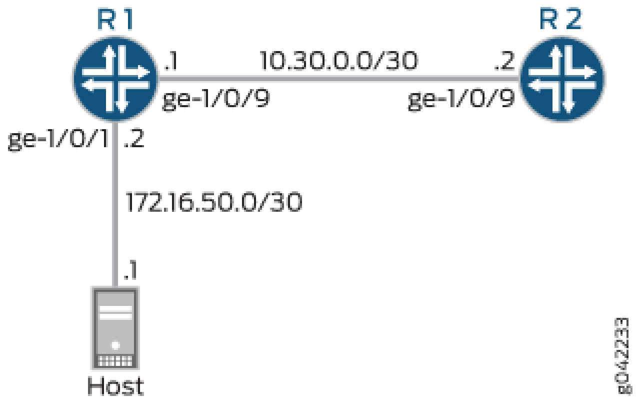


You apply the multifield classifier's firewall filter as an input filter on each customer-facing or host-facing interface that needs the filter. The incoming interface is ge-1/0/1 on Device R1. The classification and queue assignment is verified on the outgoing interface. The outgoing interface is Device R1's ge-1/0/9 interface.

Topology

Figure 17 on page 157 shows the sample network.

Figure 17: Multifield Classifier Scenario



"CLI Quick Configuration" on page 158 shows the configuration for all of the Juniper Networks devices in Figure 17 on page 157.

"Step-by-Step Procedure" on page 159 describes the steps on Device R1.

Classifiers are described in more detail in the following Juniper Networks Learning Byte video.

Video: Class of Service Basics, Part 2: Classification Learning Byte

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces ge-1/0/1 description to-host
set interfaces ge-1/0/1 unit 0 family inet filter input mf-classifier
set interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
set interfaces ge-1/0/9 description to-R2
set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.1/30
set class-of-service forwarding-classes class BE-data queue-num 0
set class-of-service forwarding-classes class Premium-data queue-num 1
set class-of-service forwarding-classes class Voice queue-num 2
set class-of-service forwarding-classes class NC queue-num 3
set firewall family inet filter mf-classifier term BE-data from protocol tcp
set firewall family inet filter mf-classifier term BE-data from port 80
set firewall family inet filter mf-classifier term BE-data then forwarding-class BE-data
set firewall family inet filter mf-classifier term Premium-data from protocol tcp
set firewall family inet filter mf-classifier term Premium-data from port 12345
set firewall family inet filter mf-classifier term Premium-data then forwarding-class Premium-
data
set firewall family inet filter mf-classifier term accept-all-else then accept
```

Device R2

```
set interfaces ge-1/0/9 description to-R1
set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.2/30
```

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set ge-1/0/1 description to-host
user@R1# set ge-1/0/1 unit 0 family inet address 172.16.50.2/30
user@R1# set ge-1/0/9 description to-R2
user@R1# set ge-1/0/9 unit 0 family inet address 10.30.0.1/30
```

2. Configure the custom forwarding classes and associated queue numbers.

```
[edit class-of-service forwarding-classes]
user@R1# set BE-data queue-num 0
user@R1# set Premium-data queue-num 1
user@R1# set Voice queue-num 2
user@R1# set NC queue-num 3
```

3. Configure the firewall filter term that places TCP traffic with a source port of 80 (HTTP traffic) into the BE-data forwarding class, associated with queue 0.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term BE-data from protocol tcp
user@R1# set term BE-data from port 80
user@R1# set term BE-data then forwarding-class BE-data
```

4. Configure the firewall filter term that places TCP traffic with a source port of 12345 into the Premium-data forwarding class, associated with queue 1.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term Premium-data from protocol tcp
user@R1# set term Premium-data from port 12345
user@R1# set term Premium-data then forwarding-class Premium-data
```

5. At the end of your firewall filter, configure a default term that accepts all other traffic.

   Otherwise, all traffic that arrives on the interface and is not explicitly accepted by the firewall filter is discarded.

   ```
   [edit firewall family inet filter mf-classifier]
   user@R1# set term accept-all-else then accept
   ```

6. Apply the firewall filter to the ge-1/0/1 interface as an input filter.

   ```
   [edit interfaces]
   user@R1# set ge-1/0/1 unit 0 family inet filter input mf-classifier
   ```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show class-of-service`, `show firewall` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
ge-1/0/1 {
    description to-host;
    unit 0 {
        family inet {
            filter {
                input mf-classifier;
            }
            address 172.16.50.2/30;
        }
    }
}
ge-1/0/9 {
    description to-R2;
    unit 0 {
        family inet {
            address 10.30.0.1/30;
        }
```

```
    }
}
```

```
user@R1# show class-of-service
forwarding-classes {
    class BE-data queue-num 0;
    class Premium-data queue-num 1;
    class Voice queue-num 2;
    class NC queue-num 3;
}
```

```
user@R1# show firewall
family inet {
    filter mf-classifier {
        term BE-data {
            from {
                protocol tcp;
                port 80;
            }
            then forwarding-class BE-data;
        }
        term Premium-data {
            from {
                protocol tcp;
                port 12345;
            }
            then forwarding-class Premium-data;
        }
        term accept-all-else {
            then accept;
        }
    }
}
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking the CoS Settings

Purpose

Confirm that the forwarding classes are configured correctly.

Action

From Device R1, run the `show class-of-service forwardng-classes` command.

```
user@R1> show class-of-service forwarding-class

Forwarding class                       ID      Queue  Restricted queue  Fabric priority
Policing priority   SPU priority
  BE-data                               0       0          0             low
normal            low
  Premium-data                          1       1          1             low
normal            low
  Voice                                 2       2          2             low
normal            low
  NC                                    3       3          3             low
normal            low
```

Meaning

The output shows the configured custom classifier settings.

Sending TCP Traffic into the Network and Monitoring the Queue Placement

Purpose

Make sure that the traffic of interest is sent out the expected queue.

Action

1. Clear the interface statistics on Device R1's outgoing interface.

```
user@R1> clear interfaces statistics ge-1/0/9
```

2. Use a traffic generator to send 50 TCP port 80 packets to Device R2 or to some other downstream device.

3. On Device R1, check the queue counters.

   Notice that you check the queue counters on the downstream output interface, not on the incoming interface.

```
user@R1> show interfaces extensive ge-1/0/9 | find "Queue counters"

  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0                               50                   50                    0
    1                                0                   57                    0
    2                                0                    0                    0
    3                                0                    0                    0
```

4. Use a traffic generator to send 50 TCP port 12345 packets to Device R2 or to some other downstream device.

```
[root@host]#  hping 172.16.60.1  -c 50 -s 12345  -k
```

5. On Device R1, check the queue counters.

```
user@R1> show interfaces extensive ge-1/0/9 | find "Queue counters"

  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0                               50                   50                    0
    1                          50                 57                 0
    2                           0                  0                 0
    3                           0                  0                 0
```

Meaning

The output shows that the packets are classified correctly. When port 80 is used in the TCP packets, queue 0 is incremented. When port 12345 is used, queue 1 is incremented.

RELATED DOCUMENTATION

Example: Configuring a Two-Rate Three-Color Policer

Example: Classify Packets Based on Their Destination Address

IN THIS SECTION

This example shows how to classify packets based on their destination address by using a multifield classifier.

Requirements

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

Overview

In this example you configure a multifield classifier (firewall filter) 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.

You then apply this filter to the inbound interface `xe-1/2/2.0` and verify your configuration is attached to the correct interface, issue the `show interfaces filters` command.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure the multifield classifer (firewall filter), copy the following commands to a text file, remove any line breaks, and then paste the commands into the CLI.

```
set firewall family inet filter set-FC-to-platinum term match-a-single-route from destination-
address 10.10.10.0/24
set firewall family inet filter set-FC-to-platinum term match-a-single-route then forwarding-
class platinum
set firewall family inet filter set-FC-to-platinum term match-a-single-route then accept
set firewall family inet filter set-FC-to-platinum term accept-all then accept
set interfaces xe-1/2/2 unit 0 family inet filter input set-FC-to-platinum
```

Configuring Firewall Filter

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see Using the CLI Editor in Configuration Mode. To configure the multifield classifier (firewall filter):

1. Create and configure the multifield classifier (firewall filter).

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

2. Apply the classifier to the interface.

```
[edit interfaces]
set interfaces xe-1/2/2 unit 0 family inet filter input set-FC-to-platinum
```

Results

Confirm your configuration by entering the `show firewall` and `show interfaces` commands from configuration mode. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@host# show firewall
```

```
filter set-FC-to-platinum {
        term match-a-single-route {
            from {
                destination-address {
                    10.10.10.0/24;
                }
            }
            then {
                forwarding-class platinum;
                accept;
            }
        }
```

```
user@host# show interfaces
```

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

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

RELATED DOCUMENTATION

Assign CoS Levels to Packets Based on Multiple Packet Header Fields | 148

Configure Multifield Classifiers | 150

Example: Use Multifield Classifiers to Classify VoIP and SIP Traffic

IN THIS SECTION

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

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 router 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 Junos.

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.

CHAPTER 4

Control Network Access with Traffic Policing

IN THIS CHAPTER

Control Network Access Using Traffic Policing Overview

IN THIS SECTION

Congestion Management for IP Traffic Flows

Traffic policing, also known as *rate limiting*, is an essential component of network access security that is designed to thwart denial-of-service (DoS) attacks. Traffic policing enables you to control the maximum rate of IP traffic sent or received on an interface and also to partition network traffic into multiple priority levels, also known as *classes of service*. A policer defines a set of traffic rate limits and sets consequences for traffic that does not conform to the configured limits. Packets in a traffic flow that do not conform to traffic limits are either discarded or marked with a different forwarding class or packet loss priority (PLP) level.

With the exception of policers configured to rate-limit aggregate traffic (all protocol families and logical interfaces configured on a physical interface), you can apply a policer to all IP packets in a Layer 2 or Layer 3 traffic flow at a *logical interface*.

With the exception of policers configured to rate-limit based on physical interface media rate, you can apply a policer to specific IP packets in a Layer 3 traffic flow at a logical interface by using a stateless *firewall filter*.

You can apply a policer to inbound or outbound interface traffic. Policers applied to inbound traffic help to conserve resources by dropping traffic that does not need to be routed through a network. Dropping inbound traffic also helps to thwart denial-of-service (DoS) attacks. Policers applied to outbound traffic control the bandwidth used.

NOTE: Traffic policers are instantiated on a per-PIC basis. Traffic policing does not work when the traffic for one local policy decision function (L-PDF) subscriber is distributed over multiple Multiservices PICs in an AMS group.

Traffic Limits

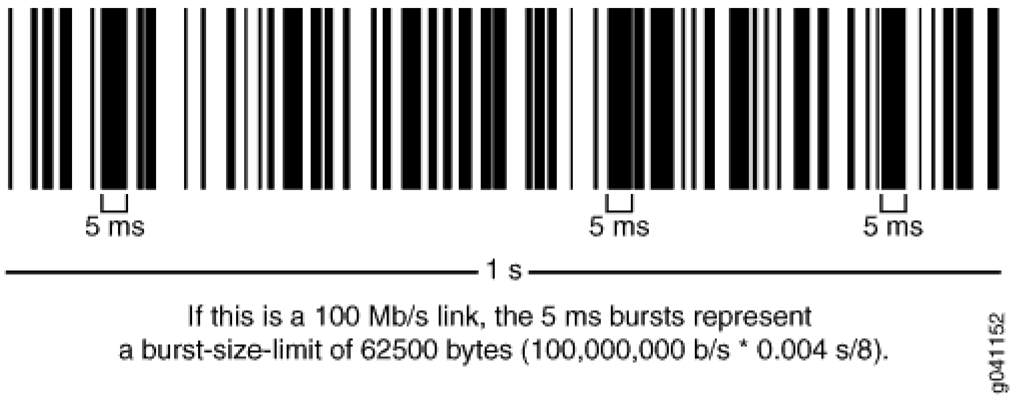
Junos policers use a *token bucket algorithm* to enforce a limit on an average transmit or receive rate of traffic at an interface while allowing bursts of traffic up to a maximum value based on the configured bandwidth limit and configured burst size. The token bucket algorithm offers more flexibility than a *leaky bucket algorithm* in that you can allow a specified traffic burst before starting to discard packets. The token bucket algorithm also enables you to apply a penalty such as packet output-queuing priority or packet-drop priority.

In the token-bucket model, the bucket represents the rate-limiting function of the policer. Tokens are added to the bucket at a fixed rate, but once the specified depth of the bucket is reached, tokens allocated after cannot be stored and used. Each token represents a “credit” for some number of bits, and tokens in the bucket are “cashed in” for the ability to transmit or receive traffic at the interface. When sufficient tokens are present in the bucket, a traffic flow continues unrestricted. Otherwise, packets might be dropped or else re-marked with a lower forwarding class, a higher packet loss priority (PLP) level, or both.

- The rate at which tokens are added to the bucket represents the highest average transmit or receive rate in bits per second allowed for a given service level. You specify this highest average traffic rate as the *bandwidth limit* of the policer. If the traffic arrival rate (or fixed bits-per-second) is so high that at some point insufficient tokens are present in the bucket, then the traffic flow is no longer conforming to the traffic limit. During periods of relatively low traffic (traffic that arrives at or departs from the interface at average rates below the token arrival rate), unused tokens accumulate in the bucket.

- The depth of the bucket in bytes controls the amount of back-to-back bursting allowed. You specify this factor as the *burst-size limit* of the policer. This second limit affects the average transmit or receive rate by limiting the number of bytes permitted in a transmission burst for a given interval of time. Bursts exceeding the current burst-size limit are dropped until there are sufficient tokens available to permit the burst to proceed.

Figure 18: Network Traffic and Burst Rates


5 ms
5 ms
5 ms
1 s
If this is a 100 Mb/s link, the 5 ms bursts represent
a burst-size-limit of 62500 bytes (100,000,000 b/s * 0.004 s/8).
g041152

As shown in the figure above, a UPC bar code is a good facsimile of what traffic looks like on the line; an interface is either transmitting (bursting at full rate) or it is not. The black lines represent periods of data transmission and the white space represents periods of silence when the token bucket can replenish.

Depending on the type of policer used, packets in a policed traffic flow that surpasses the defined limits might be implicitly set to a higher PLP level, assigned to a configured forwarding class or set to a configured PLP level (or both), or simply discarded. If packets encounter downstream congestion, packets with a `low` PLP level are less likely to be discarded than those with a `medium-low`, `medium-high`, or `high` PLP level.

Traffic Color Marking

Based on the particular set of traffic limits configured, a policer identifies a traffic flow as belonging to one of either two or three categories that are similar to the colors of a traffic light used to control automobile traffic.

- *Single-rate two-color*—A two-color marking policer (or "policer" when used without qualification) meters the traffic stream and classifies packets into two categories of packet loss priority (PLP) according to a configured bandwidth and burst-size limit. You can mark packets that exceed the bandwidth and burst-size limit in some way, or simply discard them.

  A policer is most useful for metering traffic at the port (physical interface) level.

- *Single-rate three-color*—This type of policer is defined in RFC 2697, *A Single Rate Three Color Marker*, as part of an assured forwarding (AF) per-hop-behavior (PHB) classification system for a Differentiated Services (DiffServ) environment. This type of policer meters traffic based on the configured committed information rate (CIR), committed burst size (CBS), and the excess burst size (EBS). Traffic is marked as belonging to one of three categories (green, yellow, or red) based on whether the packets arriving are below the CBS (green), exceed the CBS (yellow) but not the EBS, or exceed the EBS (red).

  A single-rate three-color policer is most useful when a service is structured according to packet length and not peak arrival rate.

- *Two-rate three-color*—This type of policer is defined in RFC 2698, *A Two Rate Three Color Marker*, as part of an assured forwarding (AF) per-hop-behavior (PHB) classification system for a Differentiated Services (DiffServ) environment. This type of policer meters traffic based on the configured CIR and peak information rate (PIR), along with their associated burst sizes, the CBS and *peak burst size* (PBS). Traffic is marked as belonging to one of three categories (green, yellow, or red) based on whether the packets arriving are below the CIR (green), exceed the CIR (yellow) but not the PIR, or exceed the PIR (red).

  A two-rate three-color policer is most useful when a service is structured according to arrival rates and not necessarily packet length.

Policer actions are implicit or explicit and vary by policer type. The term *Implicit* means that Junos assigns the loss-priority automatically. Table 15 on page 175 describes the policer actions.

Table 15: Policer Actions

Policer	Marking	Implicit Action	Configurable Action
Single-rate two-color	Green (Conforming)	Assign low loss priority	None

Table 15: Policer Actions *(Continued)*

Policer	Marking	Implicit Action	Configurable Action
	Red (Nonconforming)	None	Assign low or high loss priority, assign a forwarding class, or discard On some platforms, you can assign medium-low or medium-high loss priority
Single-rate three-color	Green (Conforming)	Assign low loss priority	None
	Yellow (Above the CIR and CBS)	Assign medium-high loss priority	None
	Red (Above the EBS)	Assign high loss priority	Discard
Two-rate three-color	Green (Conforming)	Assign low loss priority	None
	Yellow (Above the CIR and CBS)	Assign medium-high loss priority	None
	Red (Above the PIR and PBS)	Assign high loss priority	Discard

Forwarding Classes and PLP Levels

A packet's forwarding class assignment and PLP level are used by the Junos CoS features. The Junos CoS features include a set of mechanisms that you can use to provide DiffServ when best-effort traffic delivery is insufficient. For interfaces that carry IPv4, IPv6, and MPLS traffic, you can configure CoS features to take in a single flow of traffic entering at the edge of your network and provide different levels of service across the network. These services include internal forwarding and scheduling (queuing) for output based on the forwarding class assignments and PLP levels of the individual packets.

NOTE: Forwarding-class and loss-priority assignments performed by a policer or a stateless firewall filter override any such assignments from the default or configured BA classifier.

Based on CoS configurations, packets of a given forwarding class are transmitted through a specific output queue, and each output queue is associated with a transmission service level defined in a *scheduler*.

Based on other CoS configurations, when packets in an output queue encounter congestion, packets with higher loss-priority values are more likely to be dropped by the random early detection (RED) algorithm. Packet loss priority values affect the scheduling of a packet without affecting the packet's relative ordering within the traffic flow.

Policer Application to Traffic

After you have defined and named a policer, it is stored as a template. You can later use the same policer name to provide the same policer configuration each time you want to use it. This eliminates the need to define the same policer values more than once.

You can apply a policer to a traffic flow in either of two ways:

- You can configure a standard stateless firewall filter that specifies the `policer` *`policer-name`* nonterminating action or the `three-color-policer (single-rate | two-rate)` *`policer-name`* nonterminating action. When you apply the standard filter to the input or output at a logical interface, the policer is applied to all packets of the filter-specific protocol family that match the conditions specified in the filter configuration.

  With this method of applying a policer, you can define specific classes of traffic on an interface and apply traffic rate-limiting to each class.

- You can apply a policer directly to an interface so that traffic rate-limiting applies to all traffic on that interface, regardless of protocol family or any match conditions.

You can configure policers at the queue, logical interface, or Layer 2 (MAC) level. Only a single policer is applied to a packet at the egress queue, and the search for policers occurs in this order:

- Queue level
- Logical interface level
- Layer 2 (MAC) level

RELATED DOCUMENTATION

Stateless Firewall Filter Overview

Traffic Policer Types

Order of Policer and Firewall Filter Operations

Packet Flow Through the Junos CoS Process | **16**

Effect of Two-Color Policers on 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 you configure a logical interface bandwidth policer, 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 you configure a bandwidth percentage policer, 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.

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

```
[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;
            }
        }
    }
}
```

RELATED DOCUMENTATION

Configure Policers Based on Logical Interface Bandwidth | 179

Configure Policers Based on Logical Interface Bandwidth

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, set the `logical-bandwidth-policer` option at the `[edit firewall]` hierarchy level. Next, specify the `shaping-rate` for the logical interfaces under the `[edit class-of-service]` hierarchy level and apply the policer to the logical interfaces.
- 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 CoS statements.

The following example configures and 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).

To configure and apply this policer:

1. Create and configure the policer.
   a. Create the policer.

   ```
   [edit]
   user@host# edit firewall policer Logical_Policer
   ```

   b. Specify that the policer is based on the shaping rate of the logical interface.

   ```
   [edit firewall policer Logical_Policer]
   user@host# set logical-bandwidth-policer
   ```

   c. Configure the rate limits for the policer.

   ```
   [edit firewall policer Logical_Policer]
   user@host# set if-exceeding bandwidth-limit 50
   user@host# set burst-size-limit 125k
   ```

   d. Configure the policer to discard packets that exceed the specified rate limits.

   ```
   [edit firewall policer Logical_Policer]
   user@host# set then discard
   ```

2. Specify the shaping-rate for each logical interface.

```
{edit}
user@host# edit class-of-service interfaces ge-0/2/7
user@host# set unit 0 shaping-rate 4m
user@host# set unit 1 shaping-rate 2m
```

3. Apply the policer to the logical interfaces.

   - Enable scheduling on logical interfaces.

   ```
   [edit]
   user@host# edit interfaces ge-0/2/7
   user@host# set per-unit-scheduler
   ```

   - Enable the reception and transmission of 802.1Q VLAN-tagged frames on the interface.

   ```
   [edit interfaces ge-0/2/7]
   user@host# set vlan-tagging
   ```

   - Apply the policer to the first logical interface.

   ```
   [edit interfaces ge-0/2/7]
   user@host# set unit 0 vlan-id 100 family inet policer input Logical_Policer
   user@host# set unit 0 vlan-id 100 family inet policer output Logical_Policer
   user@host# set unit 0 vlan-id 100 family inet address 172.16.1.1/30
   ```

   - Apply the policer to the second logical interface.

   ```
   [edit interfaces ge-0/2/7]
   user@host# set unit 1 vlan-id 200 family inet policer input Logical_Policer
   user@host# set unit 1 vlan-id 200 family inet policer output Logical_Policer
   user@host# set unit 1 vlan-id 200 family inet address 172.26.1.1/30
   ```

4. Confirm your configuration.

```
[edit]
user@host# show firewall
```

```
policer Logical_Policer {
    logical-bandwidth-policer;
    if-exceeding {
        bandwidth-percent 50;
        burst-size-limit 125k;
    }
    then discard;
}
```

```
[edit]
user@host# show class-of-service interfaces ge-0/2/7
```

```
unit 0 {
    shaping-rate 4m;
}
unit 1 {
    shaping-rate 2m;
}
```

```
[edit]
user@host# show 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 172.16.1.1/30;
    }
}
unit 1 {
    vlan-id 200;
    family inet {
        policer {
            input Logical_Policer;
            output Logical_Policer;
        }
        address 172.26.1.1/30;
    }
}
```

5. Save the configuration.

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

RELATED DOCUMENTATION

Controlling Network Access Using Traffic Policing Overview

logical-bandwidth-policer

shaping-rate (Applying to an Interface)

Example: Configure an Ingress Single-Rate Two-Color Policer

IN THIS SECTION

This example shows you how to configure an ingress single-rate two-color policer to filter incoming traffic. The policer enforces the CoS strategy for in-contract and out-of-contract traffic. You can apply a single-rate two-color policer to incoming packets, outgoing packets, or both. This example applies the policer as an input (ingress) policer. The goal of this topic is to provide you with an introduction to policing by using an example that shows traffic policing in action.

Policers use a concept known as a token bucket to allocate system resources based on the parameters defined for the policer. A thorough explanation of the token bucket concept and its underlying algorithms is beyond the scope of this document. For more information about traffic policing, and CoS in general, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at www.juniper.net/books.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos. The example shown here was tested and verified on MX Series routers running Junos OS Release 10.4.

Overview

IN THIS SECTION

Single-rate two-color policing enforces a configured rate of traffic flow for a particular service level by applying implicit or configured actions to traffic that does not conform to the limits. When you apply a single-rate two-color policer to the input or output traffic at an interface, the policer meters the traffic flow to the rate limit defined by the following components:

- Bandwidth limit—The average number of bits per second permitted for packets received or transmitted at the interface. You can specify the bandwidth limit as an absolute number of bits per second or as a percentage value from 1 through 100. If a percentage value is specified, the effective bandwidth limit is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.

- Burst-size limit—The maximum size permitted for bursts of data. Burst sizes are measured in bytes. We recommend two formulas for calculating burst size:

  Burst size = bandwidth x allowable time for burst traffic / 8

  Or

Burst size = interface mtu x 10

For information about configuring the burst size, see *Determining Proper Burst Size for Traffic Policers*.

NOTE: There is a finite buffer space for an interface. In general, the estimated total buffer depth for an interface is about 125 ms.

For a traffic flow that conforms to the configured limits (categorized as green traffic), packets are implicitly marked with a packet loss priority (PLP) level of low and are allowed to pass through the interface unrestricted.

For a traffic flow that exceeds the configured limits (categorized as red traffic), packets are handled according to the traffic-policing actions configured for the policer. This example discards packets that burst over the 15 KBps limit.

To rate-limit Layer 3 traffic, you can apply a two-color policer in the following ways:

- Directly to a logical interface, at a specific protocol level.
- As the action of a standard stateless firewall filter that is applied to a logical interface, at a specific protocol level. This is the technique used in this example.

To rate-limit Layer 2 traffic, you can apply a two-color policer as a logical interface policer only. You cannot apply a two-color policer to Layer 2 traffic through a firewall filter.

CAUTION: You can choose either bandwidth-limit or bandwidth percent within the policer, as they are mutually exclusive. You cannot configure a policer to use bandwidth percent for aggregate, tunnel, and software interfaces.

In this example, the host is a traffic generator emulating a webserver. Devices R1 and R2 are owned by a service provider. The webserver is accessed by users on Device Host2. Device Host1 will be sending traffic with a source TCP HTTP port of 80 to the users. A single-rate two-color policer is configured and applied to the interface on Device R1 that connects to Device Host1. The policer enforces the contractual bandwidth availability made between the owner of the webserver and the service provider that owns Device R1 for the web traffic that flows over the link that connects Device Host1 to Device R1.

In accordance with the contractual bandwidth availability made between the owner of the webserver and the service provider that owns Devices R1 and R2, the policer will limit the HTTP port 80 traffic originating from Device Host1 to using 700 Mbps (70 percent) of the available bandwidth with an allowable burst rate of 10 x the MTU size of the gigabit Ethernet interface between the host Device Host1 and Device R1.

NOTE: In a real-world scenario you would probably also rate limit traffic for a variety of other ports such as FTP, SFTP, SSH, TELNET, SMTP, IMAP, and POP3 because they are often included as additional services with web hosting services.

NOTE: You need to leave some additional bandwidth available that is not rate limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

Topology

This example uses the topology in Figure 19 on page 186.

Figure 19: Single-Rate Two-Color Policer Scenario

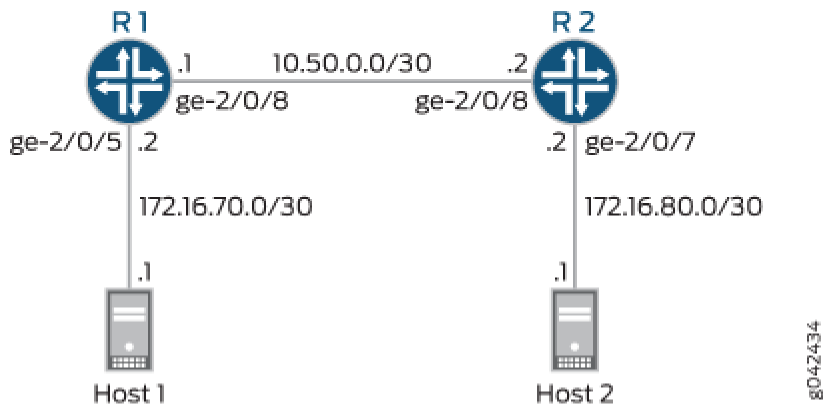
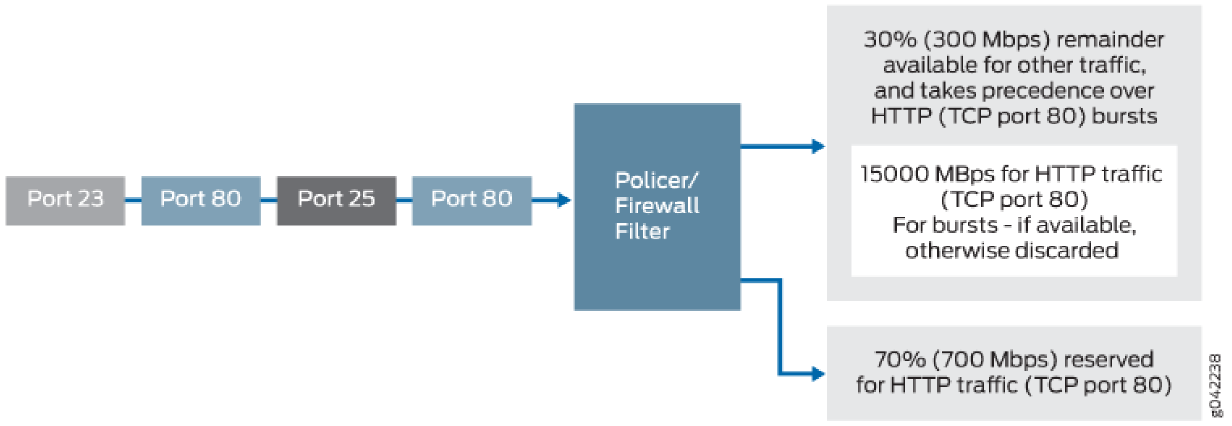


Figure 20 on page 187 shows the policing behavior.

Figure 20: Traffic Limiting in a Single-Rate Two-Color Policer Scenario


30% (300 Mbps) remainder available for other traffic, and takes precedence over HTTP (TCP port 80) bursts
15000 MBps for HTTP traffic (TCP port 80) For bursts - if available, otherwise discarded
Port 23
Port 80
Port 25
Port 80
Policer/ Firewall Filter
70% (700 Mbps) reserved for HTTP traffic (TCP port 80)
g042238

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set firewall policer discard if-exceeding bandwidth-limit 700m
set firewall policer discard if-exceeding burst-size-limit 15k
set firewall policer discard then discard
set firewall family inet filter mf-classifier term t1 from protocol tcp
set firewall family inet filter mf-classifier term t1 from port 80
set firewall family inet filter mf-classifier term t1 then policer discard
set firewall family inet filter mf-classifier term t2 then accept
set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Device R2

```
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces ge-2/0/7 description to-Host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.14.1/32
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

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

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set ge-2/0/5 description to-Host
user@R1# set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set ge-2/0/8 description to-R2
user@R1# set ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set lo0 unit 0 description looback-interface
user@R1# set lo0 unit 0 family inet address 192.168.13.1/32
```

2. Apply the firewall filter to interface ge-2/0/5 as an input filter.

```
[edit interfaces ge-2/0/5 unit 0 family inet]
user@R1# set filter input mf-classifier
```

3. Configure the policer to rate-limit to a bandwidth of 700 Mbps and a burst size of 15000 KBps for HTTP traffic (TCP port 80).

```
[edit firewall policer discard]
user@R1# set if-exceeding bandwidth-limit 700m
user@R1# set if-exceeding burst-size-limit 15k
```

4. Configure the policer to discard packets in the red traffic flow.

```
[edit firewall policer discard]
user@R1# set then discard
```

5. Configure the two conditions of the firewall to accept all TCP traffic to port HTTP (port 80).

```
[edit firewall  family inet filter mf-classifier]
user@R1# set term t1 from protocol tcp
user@R1# set term t1 from port 80
```

6. Configure the firewall action to rate-limit HTTP TCP traffic using the policer.

```
[edit firewall  family inet filter mf-classifier]
user@R1# set term t1 then policer discard
```

7. At the end of the firewall filter, configure a default action that accepts all other traffic.

   Otherwise, all traffic that arrives on the interface and is not explicitly accepted by the firewall is discarded.

```
[edit firewall  family inet filter mf-classifier]
user@R1# set term t2 then accept
```

8. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set ge-2/0/8 description to-R1
user@R1# set ge-2/0/7 description to-Host
user@R1# set lo0 unit 0 description looback-interface
user@R1# set ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@R1# set ge-2/0/7 unit 0 family inet address 172.16.80.2/30
user@R1# set lo0 unit 0 family inet address 192.168.14.1/32
```

2. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/7.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces` , `show firewall`, and `show protocols ospf` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
ge-2/0/5 {
    description to-Host;
    unit 0 {
        family inet {
            filter {
                input mf-classifier;
            }
            address 172.16.70.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R2;
    unit 0 {
        family inet {
            address 10.50.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
```

```
user@R1# show firewall
family inet {
    filter mf-classifier {
        term t1 {
            from {
                protocol tcp;
                port 80;
            }
            then policer discard;
        }
        term t2 {
            then accept;
        }
    }
}
policer discard {
    if-exceeding {
        bandwidth-limit 700m;
        burst-size-limit 15k;
    }
    then discard;
}
```

```
user@R1# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/5.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R1, enter `commit` from configuration mode.

```
user@R2# show interfaces
ge-2/0/7 {
    description to-Host;
    unit 0 {
        family inet {
            address 172.16.80.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.50.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.14.1/32;
        }
    }
}
```

```
user@R2# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/7.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R2, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Clearing the Counters

Purpose

Confirm that the firewall counters are cleared.

Action

On Device R1, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@R1> clear firewall all
```

Sending TCP Traffic into the Network and Monitoring the Discards

Purpose

Make sure that the traffic of interest that is sent is rate-limited on the input interface (ge-2/0/5).

Action

1. Use a traffic generator to send 10 TCP packets with a source port of 80.

   The -s flag sets the source port. The -k flag causes the source port to remain steady at 80 instead of incrementing. The -c flag sets the number of packets to 10. The -d flag sets the packet size.

   The destination IP address of 172.16.80.1 belongs to Device Host 2 that is connected to Device R2. The user on Device Host 2 has requested a webpage from Device Host 1 (the webserver emulated by the traffic generator on Device Host 1). The packets that being rate-limited are sent from Device Host 1 in response to the request from Device Host 2.

   NOTE: In this example the policer numbers are reduced to a bandwidth limit of 8 Kbps and a burst size limit of 1500 KBps to ensure that some packets are dropped during this test.

   ```
   [root@host]#  hping 172.16.80.1  -c 10 -s 80  -k -d 300



   [User@Host]#  hping 172.16.80.1 -c 10 -s 80 -k -d 350
   HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 350 data bytes
   len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.5 ms
   .
   .
   .
   --- 172.16.80.1 hping statistic ---
   ```

```
10 packets transmitted, 6 packets received, 40% packet loss
round-trip min/avg/max = 0.5/3000.8/7001.3 ms
```

2. On Device R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall



User@R1# run show firewall

Filter: __default_bpdu_filter__

Filter: mf-classifier
Policers:
Name                                            Bytes             Packets
discard-t1                                      1560              4
```

Meaning

In Steps 1 and 2 the output from both devices shows that 4 packets were discarded This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 KBps burst option for red out-of-contract HTTP port 80 traffic was exceeded.

Example: Configure an Egress Single-Rate Two-Color Policer

IN THIS SECTION

This example shows how to configure an egress single-rate two-color policer. Policers use a concept known as a token bucket. The policer enforces the CoS strategy for in-contract and out-of-contract traffic. You can apply a single-rate two-color policer to incoming packets, outgoing packets, or both. This example applies the policer as an output (egress) policer. This example is an introduction to policing by using an example that shows traffic policing in action.

A thorough explanation of the token bucket concept and its underlying algorithms is beyond the scope of this document. For more information about traffic policing, and CoS in general, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at www.juniper.net/books.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos. The example shown here was tested and verified on MX Series routers running a supported Junos OS release.

Overview

IN THIS SECTION

Single-rate two-color policing enforces a configured rate of traffic flow for a particular service level by applying implicit or configured actions to traffic that does not conform to the limits. When you apply a single-rate two-color policer to the input or output traffic at an interface, the policer meters the traffic flow to the rate limit defined by the following components:

- Bandwidth limit—The average number of bits per second permitted for packets received or transmitted at the interface. You can specify the bandwidth limit as an absolute number of bits per second or as a percentage value from 1 through 100. If a percentage value is specified, the effective bandwidth limit is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.

- Burst-size limit—The maximum size permitted for bursts of data. Burst sizes are measured in bytes. We recommend two formulas for calculating burst size:

  Burst size = bandwidth x allowable time for burst traffic / 8

  Or

  Burst size = interface mtu x 10

For information about configuring the burst size, see *Determining Proper Burst Size for Traffic Policers*.

NOTE: There is a finite buffer space for an interface. In general, the estimated total buffer depth for an interface is about 125 ms.

For a traffic flow that conforms to the configured limits (categorized as green traffic), packets are implicitly marked with a packet loss priority (PLP) level of low and are allowed to pass through the interface unrestricted.

For a traffic flow that exceeds the configured limits (categorized as red traffic), packets are handled according to the traffic-policing actions configured for the policer. This example discards packets that burst over the 15 KBps limit.

To rate-limit Layer 3 traffic, you can apply a two-color policer in the following ways:

- Directly to a logical interface, at a specific protocol level.
- As the action of a standard stateless firewall filter that is applied to a logical interface, at a specific protocol level. This is the technique used in this example.

To rate-limit Layer 2 traffic, you can apply a two-color policer as a logical interface policer only. You cannot apply a two-color policer to Layer 2 traffic through a firewall filter.

CAUTION: You can choose either bandwidth-limit or bandwidth percent within the policer, as they are mutually exclusive. You cannot configure a policer to use bandwidth percent for aggregate, tunnel, or software interfaces.

In this example, the host is a traffic generator emulating a webserver. Devices R1 and R2 are owned by a service provider. The webserver is accessed by users behind Device R2. The host will be sending traffic with a source TCP HTTP port of 80 to the users. A single-rate two-color policer is configured and applied to the interface on Device R1 that connects to Device R2. The policer enforces the contractual bandwidth availability made between the owner of the webserver (in this case emulated by the host) and the service provider that owns Devices R1 and R2 for the web traffic that flows over the link that connects Devices R1 and R2.

In accordance with the contractual bandwidth availability made between the owner of the webserver and the service provider that owns Devices R1 and R2, the policer will limit the HTTP port 80 traffic originating from the host to using 700 Mbps (70 percent) of the available bandwidth with an allowable burst rate of 10 x the MTU size of the gigabit Ethernet interface between Devices R1 and R2.

NOTE: In a real-world scenario you would probably also rate-limit traffic for a variety of other ports such as FTP, SFTP, SSH, TELNET, SMTP, IMAP, and POP3 because they are often included as additional services with web hosting services.

NOTE: You need to leave some additional bandwidth available that is not rate-limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

Topology

This example uses the topology in Figure 21 on page 198.

Figure 21: Single-Rate Two-Color Policer Scenario

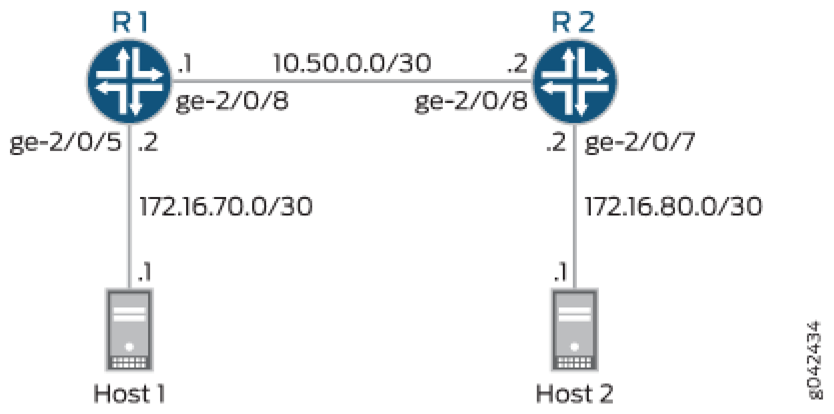
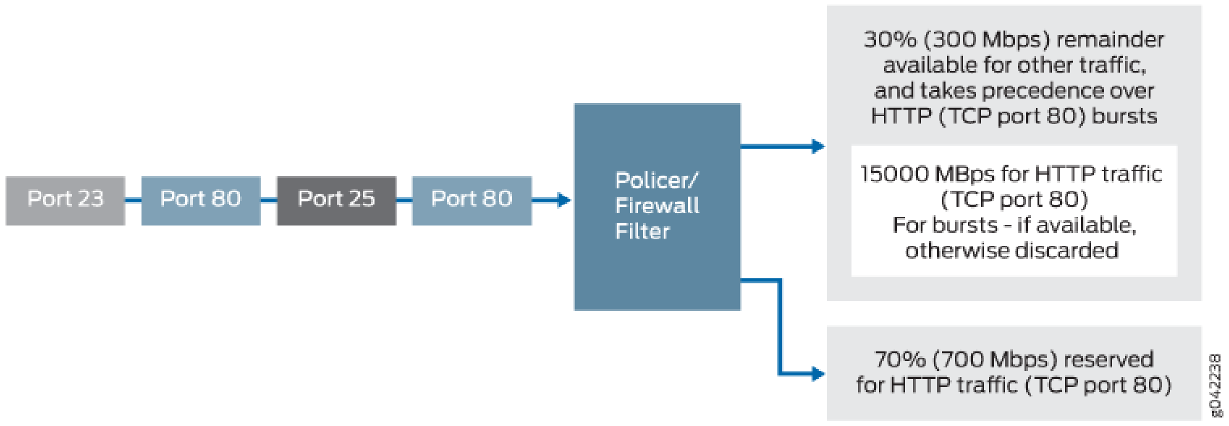


Figure 22 on page 199 shows the policing behavior.

Figure 22: Traffic Limiting in a Single-Rate Two-Color Policer Scenario


Port 23
Port 80
Port 25
Port 80
Policer/ Firewall Filter
30% (300 Mbps) remainder available for other traffic, and takes precedence over HTTP (TCP port 80) bursts
15000 MBps for HTTP traffic (TCP port 80) For bursts - if available, otherwise discarded
70% (700 Mbps) reserved for HTTP traffic (TCP port 80)
g042238

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces ge-2/0/8 unit 0 family inet filter output mf-classifier
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set firewall policer discard if-exceeding bandwidth-limit 700m
set firewall policer discard if-exceeding burst-size-limit 15k
set firewall policer discard then discard
set firewall family inet filter mf-classifier term t1 from protocol tcp
set firewall family inet filter mf-classifier term t1 from port 80
set firewall family inet filter mf-classifier term t1 then policer discard
set firewall family inet filter mf-classifier term t2 then accept
set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Device R2

```
set interfaces ge-2/0/7 description to-Host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.14.1/32
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1#set ge-2/0/5 description to-Host
user@R1#set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1#set ge-2/0/8 description to-R2
user@R1#set ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set lo0 unit 0 description looback-interface
user@R1#set lo0 unit 0 family inet address 192.168.13.1/32
```

2. Configure the policer to rate-limit to a bandwidth of 700 Mbps and a burst size of 15 KBps for HTTP traffic (TCP port 80).

```
[edit firewall policer discard]
user@R1# set if-exceeding bandwidth-limit 700m
user@R1# set if-exceeding burst-size-limit 15k
```

3. Configure the policer to discard packets in the red traffic flow.

```
[edit firewall policer discard]
user@R1# set then discard
```

4. Configure the two conditions of the firewall to accept all TCP traffic to port HTTP (port 80).

```
[edit firewall  family inet filter mf-classifier]
user@R1# set term t1 from protocol tcp
user@R1# set term t1 from port 80
```

5. Configure the firewall action to rate-limit HTTP TCP traffic using the policer.

```
[edit firewall  family inet filter mf-classifier]
user@R1# set term t1 then policer discard
```

6. At the end of the firewall filter, configure a default action that accepts all other traffic.

   Otherwise, all traffic that arrives on the interface and is not explicitly accepted by the firewall is discarded.

```
[edit firewall  family inet filter mf-classifier]
user@R1# set term t2 then accept
```

7. Apply the firewall filter to interface ge-2/0/8 as an output filter.

```
[edit interfaces ge-2/0/8 unit 0 family inet]
user@R1# set filter output mf-classifier
```

8. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
set ge-2/0/7 description to-Host
set ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set ge-2/0/8 description to-R1
set ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set lo0 unit 0 description looback-interface
set lo0 unit 0 family inet address 192.168.14.1/32
```

2. Configure OSPF.

```
[edit protocols ospf]
set area 0.0.0.0 interface ge-2/0/7.0 passive
set area 0.0.0.0 interface lo0.0 passive
set area 0.0.0.0 interface ge-2/0/8.0
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show firewall`, and `show protocols OSPF` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
ge-2/0/5 {
    description to-Host;
    unit 0 {
        family inet {
            address 172.16.70.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R2;
    unit 0 {
        family inet {
            filter {
                output mf-classifier;
            }
            address 10.50.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
```

```
user@R1# show firewall
family inet {
    filter mf-classifier {
        term t1 {
            from {
                protocol tcp;
                port 80;
            }
            then policer discard;
        }
        term t2 {
            then accept;
        }
    }
}
policer discard {
    if-exceeding {
        bandwidth-limit 700m;
        burst-size-limit 15k;
    }
    then discard;
}
```

```
policer discard {
    if-exceeding {
        bandwidth-limit 700m;
        burst-size-limit 15k;
    }
    then discard;
}
```

```
user@R1# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/5.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R1, enter `commit` from configuration mode.

```
user@R2# show interfaces
ge-2/0/7 {
    description to-Host;
    unit 0 {
        family inet {
            address 172.16.80.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.50.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.14.1/32;
        }
    }
}
```

```
user@R2# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/7.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R2, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Clearing the Counters

Purpose

Confirm that the firewall counters are cleared.

Action

On Device R1, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@R1> clear firewall all
```

Sending TCP Traffic into the Network and Monitoring the Discards

Purpose

Make sure that the traffic of interest that is sent is rate-limited on the output interface (ge-2/0/8).

Action

1. Use a traffic generator to send 20 TCP packets with a source port of 80.

   The -s flag sets the source port. The -k flag causes the source port to remain steady at 80 instead of incrementing. The -c flag sets the number of packets to 10. The -d flag sets the packet size.

   The destination IP address of 172.16.80.1 represents a user that is downstream of Device R2. The user has requested a webpage from the host (the webserver emulated by the traffic generator), and the packets are sent in response to the request.

   NOTE: In this example the policer numbers are reduced to a bandwidth limit of 8 Kbps and a burst size limit of 1500 KBps to ensure that some packets are dropped.

```
[root@host]#  hping 172.16.80.1 -s 80 -k -d 375 -c 20


[root@tp-lnx03 rtwright]# hping 172.16.80.1 -s 80 -k -d 375 -c 20
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 375 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=4000.8 ms
.
```

```
.
.
--- 172.16.80.1 hping statistic ---
20 packets transmitted, 12 packets received, 40% packet loss
```

2. On Device R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall



user@sugar# run show firewall

Filter: mf-classifier
Policers:
Name                                                Bytes              Packets
discard-t1                                3320                8
```

Meaning

In Steps 1 and 2 the output from both devices shows that 8 packets were discarded. This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 KBps burst option for red out-of-contract HTTP port 80 traffic was exceeded.

RELATED DOCUMENTATION

Routing Policies, Firewall Filters, and Traffic Policers User Guide

Example: Configuring a Two-Rate Three-Color Policer

Example: Configure an Ingress Single-Rate Two-Color Policer and Multifield Classifiers

IN THIS SECTION

This example shows how to limit customer traffic within your network using a single-rate two-color policer. Policers use a concept known as a token bucket to identify which traffic to drop. The policer enforces the CoS strategy of in-contract and out-of-contract traffic at the interface level. You can apply a single-rate two-color policer to incoming packets, outgoing packets, or both. This example applies the policer as an input (ingress) policer for incoming traffic. The multifield classifier CoS queuing option places the traffic into the assigned queues which will help you manage resource utilization at the output interface level by applying scheduling and shaping at a later date.

A thorough explanation of the token bucket concept and its underlying algorithms is beyond the scope of this document. For more information about traffic policing, and CoS in general, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at www.juniper.net/books .

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos. The example shown here was tested and verified on MX Series routers running a supported Junos OS release.

Overview

IN THIS SECTION

Policing

Single-rate two-color policing enforces a configured rate of traffic flow for a particular service level by applying implicit or configured actions to traffic that does not conform to the limits. When you apply a single-rate two-color policer to the input or output traffic at an interface, the policer meters the traffic flow to the rate limit defined by the following components:

- Bandwidth limit—The average number of bits per second permitted for packets received or transmitted at the interface. You can specify the bandwidth limit as an absolute number of bits per second or as a percentage value from 1 through 100. If a percentage value is specified, the effective bandwidth limit is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.

- Burst-size limit—The maximum size permitted for bursts of data. Burst sizes are measured in bytes. We recommend two formulas for calculating burst size:

  Burst size = bandwidth x allowable time for burst traffic / 8

  Or

  Burst size = interface mtu x 10

  For information about configuring the burst size, see *Determining Proper Burst Size for Traffic Policers*.

  NOTE: There is a finite buffer space for an interface. In general, the estimated total buffer depth for an interface is about 125 ms.

For a traffic flow that conforms to the configured limits (categorized as green traffic), packets are implicitly marked with a packet loss priority (PLP) level of low and are allowed to pass through the interface unrestricted.

For a traffic flow that exceeds the configured limits (categorized as red traffic), packets are handled according to the traffic-policing actions configured for the policer. This example discards packets that burst over the 15 KBps limit.

To rate-limit Layer 3 traffic, you can apply a two-color policer in the following ways:

- Directly to a logical interface, at a specific protocol level.

- As the action of a standard stateless firewall filter that is applied to a logical interface, at a specific protocol level. This is the technique used in this example.

To rate-limit Layer 2 traffic, you can apply a two-color policer as a logical interface policer only. You cannot apply a two-color policer to Layer 2 traffic through a firewall filter.

CAUTION: You can choose either bandwidth-limit or bandwidth percent within the policer, as they are mutually exclusive. You cannot configure a policer to use bandwidth percent for aggregate, tunnel, or software interfaces.

In this example, the host is a traffic generator emulating a webserver. Devices R1 and R2 are owned by a service provider. The webserver is accessed by users behind Device R2. The host will be sending traffic with a source port TCP HTTP port 80 and a source port 12345 to the users. A single-rate two-color policer is configured and applied to the interface on Device R1 that connects the host to Device R1. The policer enforces the contractual bandwidth availability made between the owner of the webserver (in this case emulated by the host) and the service provider that owns Device R1 for the web traffic that flows over the link that connects the host to Device R1.

In accordance with the contractual bandwidth availability made between the owner of the webserver and the service provider that owns Devices R1 and R2, the policer will limit the HTTP port 80 traffic and the port 12345 traffic originating from the host to using 700 Mbps (70 percent) of the available bandwidth with an allowable burst rate of 10 x the MTU size of the gigabit Ethernet interface between the host and Device R1.

NOTE: In a real-world scenario you would probably also rate-limit traffic for a variety of other ports such as FTP, SFTP, SSH, TELNET, SMTP, IMAP, and POP3 because they are often included as additional services with web hosting services.

NOTE: You need to leave some additional bandwidth available that is not rate-limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

Topology

This example uses the topology in Figure 23 on page 210.

Figure 23: Single-Rate Two-Color Policer Scenario

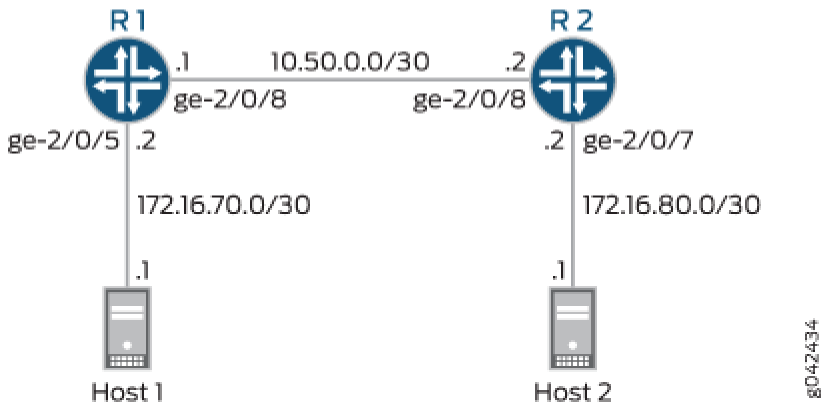
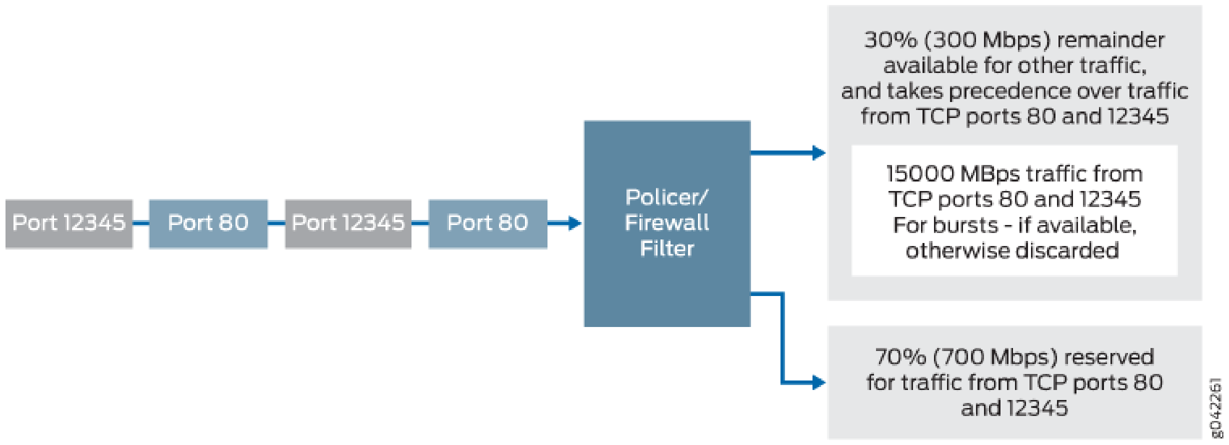


Figure 24 on page 211 shows the policing behavior.

Figure 24: Traffic Limiting in a Single-Rate Two-Color Policer Scenario


Port 12345
Port 80
Port 12345
Port 80
Policer/ Firewall Filter
30% (300 Mbps) remainder available for other traffic, and takes precedence over traffic from TCP ports 80 and 12345
15000 MBps traffic from TCP ports 80 and 12345 For bursts - if available, otherwise discarded
70% (700 Mbps) reserved for traffic from TCP ports 80 and 12345
g042261

Multifield Classifying

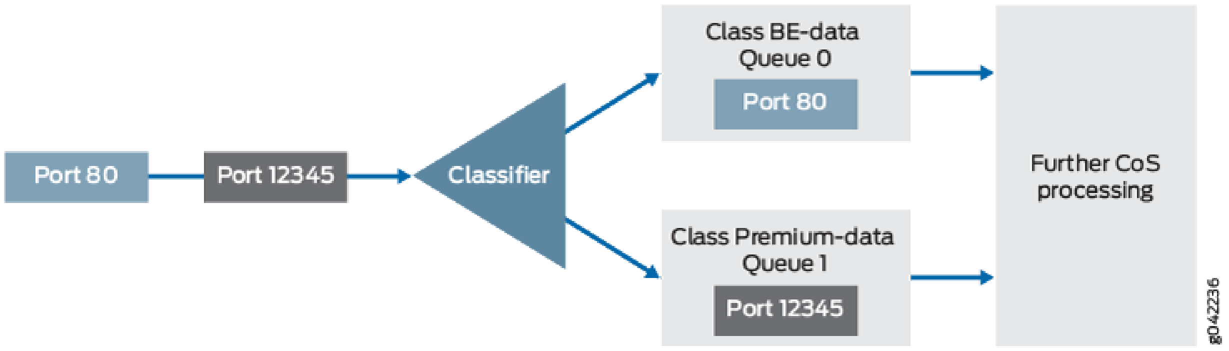
A classifier is a software operation that a router or switch uses to inspect and classify a packet after it has made it through any policing, if policing is configured. During classification, the packet header contents are examined, and this examination determines how the packet is treated when the outbound interface becomes too busy to handle all of the packets and you want your device to drop packets intelligently, instead of dropping packets indiscriminately. One common way to detect packets of interest is by source port number. The TCP source port numbers 80 and 12345 are used in this example, but many other matching criteria for packet detection are available to multifield classifiers, using firewall filter match conditions. The configuration in this example specifies that TCP packets with a source port 80 are classified into the BE-data forwarding class and queue number 0, and TCP packets with a source port 12345 are classified into the Premium-data forwarding class and queue number 1. Traffic from both port numbers is monitored by the policer first. If the traffic makes it through the policer, it is handed off to the outbound interface in the assigned queue for transmission.

Multifield classifiers are typically used at the network edge as packets enter an autonomous system (AS).

In this example, you configure the firewall filter mf-classifier and specify some custom forwarding classes on Device R1. In specifying the custom forwarding classes, you also associate each class with a queue.

The classifier operation is shown in Figure 25 on page 212.

Figure 25: Multifield Classifier Based on TCP Source Ports



You apply the multifield classifier's firewall filter as an input filter on each customer-facing or host-facing interface that needs the filter. In this example, the incoming interface ge-2/0/5 on Device R1 is used. You monitor the behavior of the queues on the interfaces that the traffic is transmitted over. In this example, to determine how the queues are being serviced, you examine the traffic statistics on interface ge-2/0/8 by using the `extensive` option in the `show interfaces` command.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set firewall policer discard if-exceeding bandwidth-limit 700m
set firewall policer discard if-exceeding burst-size-limit 15k
set firewall policer discard then discard
set class-of-service forwarding-classes class BE-data queue-num 0
set class-of-service forwarding-classes class Premium-data queue-num 1
set class-of-service forwarding-classes class Voice queue-num 2
set class-of-service forwarding-classes class NC queue-num 3
set firewall family inet filter mf-classifier term BE-data from protocol tcp
set firewall family inet filter mf-classifier term BE-data from port http
set firewall family inet filter mf-classifier term BE-data then forwarding-class BE-data
set firewall family inet filter mf-classifier term BE-data then policer discard
set firewall family inet filter mf-classifier term Premium-data from protocol tcp
set firewall family inet filter mf-classifier term Premium-data from port 12345
set firewall family inet filter mf-classifier term Premium-data then forwarding-class Premium-
data
set firewall family inet filter mf-classifier term Premium-data then policer discard
set firewall family inet filter mf-classifier term accept then accept
set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Device R2

```
set interfaces ge-2/0/7 description to-Host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.14.1/32
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1#set ge-2/0/5 description to-Host
user@R1#set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1#set ge-2/0/8 description to-R2
user@R1#set ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set lo0 unit 0 description looback-interface
user@R1#set lo0 unit 0 family inet address 192.168.13.1/32
```

2. Configure the policer to rate-limit to a bandwidth of 700 Mbps and a burst size of 15 KBps.

```
[edit firewall policer discard]
user@R1# set if-exceeding bandwidth-limit 700m
user@R1# set if-exceeding burst-size-limit 15k
```

3. Configure the policer to discard packets in the red traffic flow.

```
[edit firewall policer discard]
user@R1# set then discard
```

4. Configure the custom forwarding classes and associated queue numbers.

```
[edit class-of-service forwarding-classes]
user@R1# set class BE-data queue-num 0
user@R1# set class Premium-data queue-num 1
user@R1# set class Voice queue-num 2
user@R1# set class NC queue-num 3
```

5. Configure the firewall filter term that places TCP traffic with a source port of 80 (HTTP traffic) into the BE-data forwarding class, associated with queue 0.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term BE-data from protocol tcp
user@R1# set term BE-data from port http
user@R1# set term BE-data then forwarding-class BE-data
user@R1# set term BE-data then policer discard
```

6. Configure the firewall filter term that places TCP traffic with a source port of 12345 into the Premium-data forwarding class, associated with queue 1.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term Premium-data from protocol tcp
user@R1# set term Premium-data from port 12345
user@R1# set term Premium-data then forwarding-class Premium-data
user@R1# set term Premium-data then policer discard
```

7. At the end of your firewall filter, configure a default term that accepts all other traffic.

   Otherwise, all traffic that arrives on the interface and is not explicitly accepted by the firewall filter is discarded.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term accept then accept
```

8. Apply the firewall filter to the ge-2/0/5 interface as an input filter.

```
[edit interfaces]
user@R1# set ge-2/0/5 unit 0 family inet filter input mf-classifier
```

9. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set ge-2/0/7 description to-Host
user@R2# set ge-2/0/7 unit 0 family inet address 172.16.80.2/30
```

```
user@R2# set ge-2/0/8 description to-R1
user@R2# set ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@R2# set lo0 unit 0 description looback-interface
user@R2# set lo0 unit 0 family inet address 192.168.14.1/32
```

2. Configure OSPF.

```
[edit protocols ospf]
user@R2#  set area 0.0.0.0 interface ge-2/0/7.0 passive
user@R2#  set area 0.0.0.0 interface lo0.0 passive
user@R2#  set area 0.0.0.0 interface ge-2/0/8.0
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show class-of-service`, `show firewall`, and `show protocols ospf` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
ge-2/0/5 {
    description to-Host;
    unit 0 {
        family inet {
            filter {
                input mf-classifier;
            }
            address 172.16.70.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R2;
    unit 0 {
        family inet {
            address 10.50.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
```

```
user@R1# show class-of-service
forwarding-classes {
    class BE-data queue-num 0;
    class Premium-data queue-num 1;
    class Voice queue-num 2;
    class NC queue-num 3;
}
```

```
user@R1# show firewall
family inet {
    filter mf-classifier {
        term BE-data {
            from {
                protocol tcp;
                port http;
            }
            then {
                policer discard;
                forwarding-class BE-data;
            }
        }
        term Premium-data {
            from {
                protocol tcp;
                port 12345;
            }
            then {
                policer discard;
                forwarding-class Premium-data;
            }
        }
        term accept {
            then accept;
        }
    }
}
policer discard {
    if-exceeding {
        bandwidth-limit 700m;
        burst-size-limit 15k;
    }
    then discard;
}
```

```
user@R1# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/5.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R1, enter `commit` from configuration mode.

```
user@R2# show interfaces
ge-2/0/7 {
    description to-Host;
    unit 0 {
        family inet {
            address 172.16.80.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.50.0.2/30;
        }
    }
}
```

```
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.14.1/32;
        }
    }
}
```

```
user@R2# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/7.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R2, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking the CoS Settings

Purpose

Confirm that the forwarding classes are configured correctly.

Action

From Device R1, run the `show class-of-service forwarding-class` command.

```
user@R1> show class-of-service forwarding-class

Forwarding class                      ID      Queue  Restricted queue  Fabric priority
Policing priority   SPU priority
  BE-data                              0       0          0              low
normal            low
  Premium-data                         1       1          1              low
normal            low
  Voice                                2       2          2              low
normal            low
  NC                                   3       3          3              low
normal            low
```

Meaning

The output shows the configured custom classifier settings.

Clearing the Counters

Purpose

Confirm that the firewall and interface counters are cleared.

Action

- On Device R1, run the `clear firewall all` command to reset the firewall counters to 0.

  ```
  user@R1> clear firewall all
  ```

- On Device R1, run the `clear interface statistics ge-2/0/5` command to reset the interface counters to 0.

```
user@R1> clear interface statistics ge-2/0/8
```

Sending Traffic into the Network from TCP HTTP Port 80 and Monitoring the Results

Purpose

Send traffic that can monitored at the policer and custom queue level.

Action

1. Use a traffic generator to send 20 TCP packets with a source port of 80 into the network.

   The -s flag sets the source port. The -k flag causes the source port to remain steady at 80 instead of incrementing. The -c flag sets the number of packets to 20. The -d flag sets the packet size.

   NOTE: In this example the policer numbers are reduced to a bandwidth limit of 8 Kbps and a burst size limit of 1500 KBps to ensure that some packets are dropped.

```
[User@host]#  hping 172.16.80.1  -c 20 -s 80  -k -d 300



[root@host]# hping 172.16.80.1 -s 80 -k -c 20 -d 300
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 300 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=1.4 ms
.
.
.
--- 172.16.80.1 hping statistic ---
20 packets transmitted, 16 packets received, 20% packet loss
round-trip min/avg/max = 1.4/8688.9/17002.3 ms
```

2. On Device R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall
```

```
Filter: mf-classifier
Policers:
Name                                                Bytes              Packets
discard-BE-data                                      1360                4
discard-Premium-data                                    0                0
```

Notice that in the hping output that there was 20% packet loss (4 packets out of 20) and the same number of packets were dropped by the policer as shown in the output of the `show firewall` command. Also notice that the drops are associated with the queue BE-data as specified in the mf-classifier in the firewall configuration.

3. On Device R1, check the queue counters by using the `show interfaces extensive ge-2/0/8| find "Queue counters"` command.

```
user@R1> show interfaces extensive ge-2/0/8| find "Queue counters"

  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0                               16                   16                    0
    1                                0                    0                    0
    2                                0                    0                    0
    3                                4                    4                    0
  Queue number:         Mapped forwarding classes
    0                   BE-data
    1                   Premium-data
    2                   Voice
    3                   NC
```

Notice that 16 packets were transmitted out interface 2/0/8 using the queue BE-data as specified in the mf-classifier in the firewall configuration. The remaining 4 packets, were dropped by the policer, as shown above. The 4 packets sent to queue 3 are network control traffic. They are possibly routing protocol updates.

Meaning

The output from both devices shows that 4 packets were discarded This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 KBps burst option for red out-of-contract HTTP port 80 traffic was exceeded. In Steps 2 and 3, you can see that the correct queues were used to transmit the remaining traffic out interface 2/0/8.

Sending Traffic into the Network from TCP Port 12345 and Monitoring the Results

Purpose

Send traffic that can monitored at the policer and custom queue level.

Action

1. Clear the counters again as shown in section "Clearing the Counters" on page 220.
2. Use a traffic generator to send 20 TCP packets with a source port of 12345 into the network.

   The -s flag sets the source port. The -k flag causes the source port to remain steady at 12345 instead of incrementing. The -c flag sets the number of packets to 20. The -d flag sets the packet size.

   ```
   [User@host]#  hping 172.16.80.1  -c 20 -s 12345  -k -d 300


   [root@tp-host]# hping 172.16.80.1 -s 12345 -k -c 20 -d 300
   HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 300 data bytes
   len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.4 ms
   .
   .
   .
   --- 172.16.80.1 hping statistic ---
   20 packets transmitted, 16 packets received, 20% packet loss
   round-trip min/avg/max = 0.4/9126.3/18002.4 ms
   ```

3. On Device R1, check the firewall counters by using the `show firewall` command.

   ```
   user@R1> show firewall

   Filter: mf-classifier
   Policers:
   Name                                              Bytes            Packets
   discard-BE-data                                       0                  0
   discard-Premium-data                               1360                  4
   ```

   Notice that in the hping output that there was 20% packet loss (4 packets out of 20) and the same number of packets were dropped by the policer as shown in the output of the `show firewall` command.

Also notice that the drops are associated with the queue Premium-data as specified in the mf-classifier in the firewall configuration.

4. On Device R1, check the queue counters by using the `show interfaces extensive ge-2/0/8| find "Queue counters"` command.

```
user@R1> show interfaces extensive ge-2/0/8| find "Queue counters"

  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0                                0                    0                 0
    1                               16                   16                 0
    2                                0                    0                 0
    3                               19                   19                 0
  Queue number:         Mapped forwarding classes
    0                   BE-data
    1                   Premium-data
    2                   Voice
    3                   NC
```

Notice that 16 packets were transmitted out interface 2/0/8 using the Premium-data queues as specified in the mf-classifier firewall configuration. The remaining 4 packets were dropped by the policer, as shown above. The 19 packets sent to queue 3 are network control traffic. They are possibly routing protocol updates.

Meaning

The output from both devices shows that 4 packets were discarded. This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 KBps burst option for red out-of-contract HTTP port 80 traffic was exceeded. In Steps 3 and 4, you can see that the correct queues were used to transmit the remaining traffic out interface 2/0/8.

RELATED DOCUMENTATION

Routing Policies, Firewall Filters, and Traffic Policers User Guide

Example: Configuring a Two-Rate Three-Color Policer

Example: Configure an Egress Single-Rate Two-Color Policer and Multifield Classifiers

IN THIS SECTION

This example shows how to limit customer traffic within your network using a single-rate two-color policer. Policers use a concept known as a token bucket to identify which traffic to drop. The policer enforces the CoS strategy of in-contract and out-of-contract traffic at the interface level. You can apply a single-rate two-color policer to incoming packets, outgoing packets, or both. This example applies the policer as an output (egress) policer for outgoing traffic. The multifield classifier CoS queueing option places the traffic into the assigned queues which will help you manage resource utilization at the output interface level by applying scheduling and shaping later.

A thorough explanation of the token bucket concept and its underlying algorithms is beyond the scope of this document. For more information about traffic policing, and CoS in general, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at www.juniper.net/books.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos. The example shown here was tested and verified on MX Series routers running a supported Junos OS release.

Overview

IN THIS SECTION

Policing

Single-rate two-color policing enforces a configured rate of traffic flow for a particular service level by applying implicit or configured actions to traffic that does not conform to the limits. When you apply a single-rate two-color policer to the input or output traffic at an interface, the policer meters the traffic flow to the rate limit defined by the following components:

- Bandwidth limit—The average number of bits per second permitted for packets received or transmitted at the interface. You can specify the bandwidth limit as an absolute number of bits per second or as a percentage value from 1 through 100. If a percentage value is specified, the effective bandwidth limit is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.

- Burst-size limit—The maximum size permitted for bursts of data. Burst sizes are measured in bytes. We recommend two formulas for calculating burst size:

  Burst size = bandwidth x allowable time for burst traffic / 8

  Or

  Burst size = interface mtu x 10

  For information about configuring the burst size, see *Determining Proper Burst Size for Traffic Policers*.

  NOTE: There is a finite buffer space for an interface. In general, the estimated total buffer depth for an interface is about 125 ms.

For a traffic flow that conforms to the configured limits (categorized as green traffic), packets are implicitly marked with a packet loss priority (PLP) level of `low` and are allowed to pass through the interface unrestricted.

For a traffic flow that exceeds the configured limits (categorized as red traffic), packets are handled according to the traffic-policing actions configured for the policer. This example discards packets that burst over the 15 KBps limit.

To rate-limit Layer 3 traffic, you can apply a two-color policer in the following ways:

- Directly to a logical interface, at a specific protocol level.

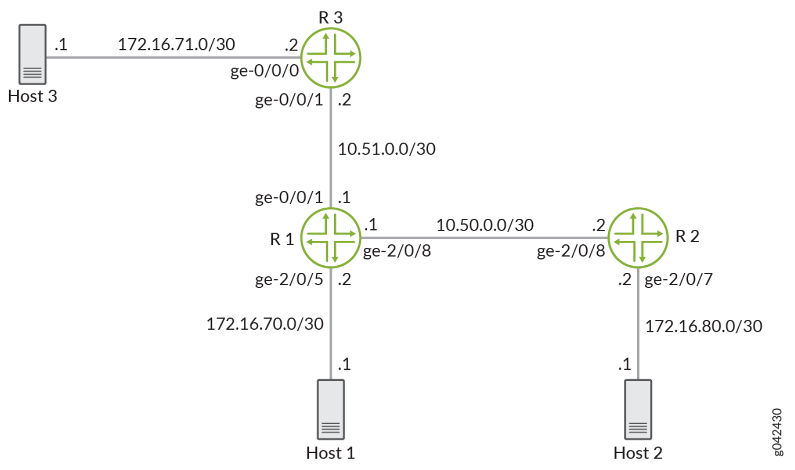
- As the action of a standard stateless firewall filter that is applied to a logical interface, at a specific protocol level. This is the technique used in this example.

To rate-limit Layer 2 traffic, you can apply a two-color policer as a logical interface policer only. You cannot apply a two-color policer to Layer 2 traffic through a firewall filter.

CAUTION: You can choose either bandwidth-limit or bandwidth percent within the policer, as they are mutually exclusive. You cannot configure a policer to use bandwidth percent for aggregate, tunnel, and software interfaces.

In this example, as illustrated in Figure 26 on page 227, Host1 connected to device R1 and Host3 connected to device R3 are traffic generators emulating webservers. Both Host1 and Host3 are sending traffic to Host2 behind device R2. Devices R1, R2, and R3 are owned by a service provider. Host1 is accessed by users on Host2 behind R2. Host1 and Host2 are owned by the same customer and their traffic must be managed. Host1 will be sending traffic with a source TCP HTTP port of 80 to the users. A single-rate two-color policer is configured and applied to the interface on R1 that connects to R2. The policer enforces the contract agreed upon by the webserver owner and the service provider for the bandwidth available to the web traffic flowing between R1 and R2.

Figure 26: Single-Rate Two-Color Policer Scenario



This example applies an egress policer between R1 and R2 because this is the point where the traffic from both customer sites shares the same link. This makes it easier to enforce the required policing parameters. Trying to rate-limit the combined customer traffic on the link between R1 and R2 by applying the policers as ingress policers on interfaces ge-0/0/0 on R3 and ge-2/0/5 on R1 would be complicated because using the contracted rate of 700 Mbps (70 percent) of the available bandwidth with an allowable burst rate of 10 x the MTU size of the gigabit Ethernet interface between Host3 and R3 and the Host1 and R1 would result in allowing a maximum throughput of 1400 Mbps over the link between R1 and R2.

Therefore, the rate-limiting applied to the host connections between the hosts and R3 and R1 would have to be reduced below 700 Mbps. The calculation of what to reduce the rate-limit number to would be a problem because just reducing each host to 350 Mbps would mean that if one host was transmitting traffic while the other host was not transmitting, the maximum throughput on the link between R1 and R2 would be only one half of the contracted rate (350 Mbps instead of 700 Mbps). This is why this example is useful to show the amount of thought that must go into applying CoS in a network to achieve the desired goals.

According to the contractual bandwidth availability, the egress policer on R1 will limit the HTTP port 80 traffic originating from Host1 to using 700 Mbps (70 percent) of the available bandwidth with an allowable burst rate of 10 x the MTU size of the gigabit Ethernet interface between R1 and R2.

Additional traffic from TCP source port 12345 is used in this example to further illustrate how traffic is allocated to the outbound queues.

NOTE: In a real-world scenario you would probably also rate-limit traffic for a variety of other ports such as FTP, SFTP, SSH, TELNET, SMTP, IMAP, and POP3 because they are often included as additional services with web hosting services.

NOTE: You must leave some additional bandwidth available that is not rate-limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

Topology

This example uses the topology in Figure 27 on page 228.

Figure 27: Single-Rate Two-Color Policer Scenario

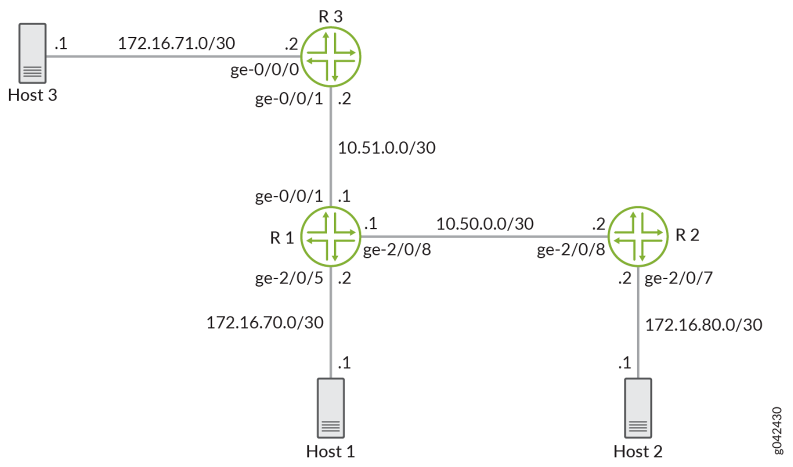
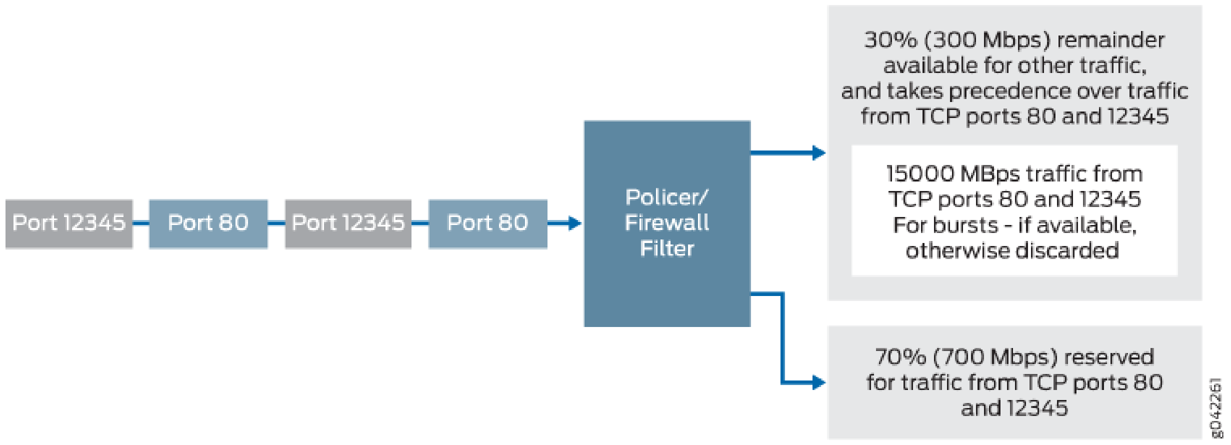


Figure 28 on page 229 shows the policing behavior.

Figure 28: Traffic Limiting in a Single-Rate Two-Color Policer Scenario


Port 12345
Port 80
Port 12345
Port 80
Policer/ Firewall Filter
30% (300 Mbps) remainder available for other traffic, and takes precedence over traffic from TCP ports 80 and 12345
15000 MBps traffic from TCP ports 80 and 12345 For bursts - if available, otherwise discarded
70% (700 Mbps) reserved for traffic from TCP ports 80 and 12345
g042261

Multifield Classifying

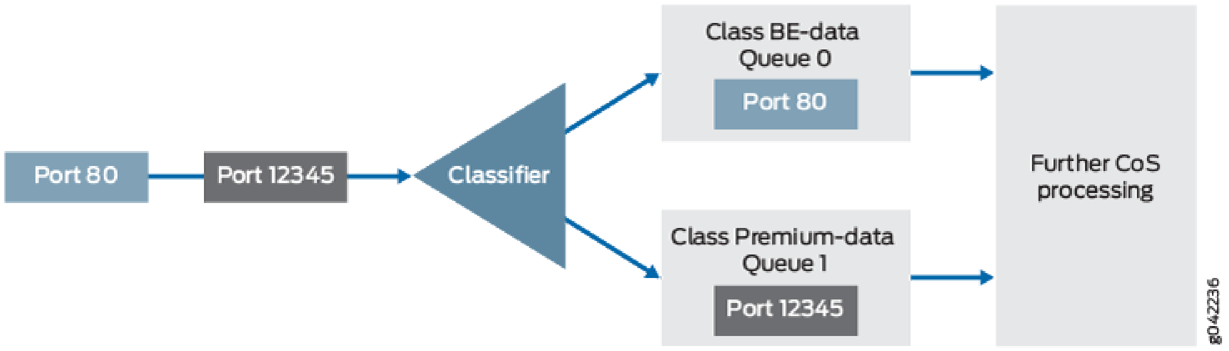
A classifier is a software operation that a router or switch uses to inspect and classify a packet after it has made it through any policing, if policing is configured. During classification, the packet header contents are examined, and this examination determines how the packet is treated when the outbound interface becomes too busy to handle all of the packets and you want your device to drop packets intelligently, instead of dropping packets indiscriminately. One common way to detect packets of interest is by source port number. The TCP source port numbers 80 and 12345 are used in this example, but many other matching criteria for packet detection are available to multifield classifiers, using firewall filter match conditions. The configuration in this example specifies that TCP packets with a source port 80 are classified into the BE-data forwarding class and queue number 0, and TCP packets with a source port 12345 are classified into the Premium-data forwarding class and queue number 1. Traffic from both port numbers is monitored by the policer first. If the traffic makes it through the policer, it is handed off to the outbound interface in the assigned queue for transmission.

Multifield classifiers are typically used at the network edge as packets enter an autonomous system (AS). However, as explained previously in the policing section, in this example the multifield classifier is configured within the AS of the service provider.

In this example, you configure the firewall filter **mf-classifier** and specify some custom forwarding classes on R1. In specifying the custom forwarding classes, you also associate each class with a queue.

The classifier operation is shown in Figure 29 on page 230.

Figure 29: Multifield Classifier Based on TCP Source Ports



You monitor the behavior of the queues on the interfaces that the traffic is transmitted over. In this example, to determine how the queues are being serviced, you examine the traffic statistics on interface ge-2/0/8 on R1 by using the `extensive` option in the `show interfaces` command.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces ge-0/0/1 description to-R3
set interfaces ge-0/0/1 unit 0 family inet address 10.51.0.1/30
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces ge-2/0/8 unit 0 family inet filter output mf-classifier
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set firewall policer discard if-exceeding bandwidth-limit 700m
set firewall policer discard if-exceeding burst-size-limit 15k
set firewall policer discard then discard
set class-of-service forwarding-classes class BE-data queue-num 0
set class-of-service forwarding-classes class Premium-data queue-num 1
set class-of-service forwarding-classes class Voice queue-num 2
set class-of-service forwarding-classes class NC queue-num 3
set firewall family inet filter mf-classifier term BE-data from protocol tcp
set firewall family inet filter mf-classifier term BE-data from port http
set firewall family inet filter mf-classifier term BE-data then forwarding-class BE-data
set firewall family inet filter mf-classifier term BE-data then policer discard
set firewall family inet filter mf-classifier term Premium-data from protocol tcp
set firewall family inet filter mf-classifier term Premium-data from port 12345
set firewall family inet filter mf-classifier term Premium-data then forwarding-class Premium-
data
set firewall family inet filter mf-classifier term Premium-data then policer discard
set firewall family inet filter mf-classifier term accept then accept
set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Device R2

```
set interfaces ge-2/0/7 description to-Host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.14.1/32
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Device R3

```
set interfaces ge-0/0/0 description to-Host
set interfaces ge-0/0/0 unit 0 family inet address 172.16.71.1/30
set interfaces ge-0/0/1 description to-R1
set interfaces ge-0/0/1 unit 0 family inet address 10.51.0.2/30
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.15.1/32
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set ge-0/0/1 description to-R3
user@R1# set ge-0/0/1 unit 0 family inet address 10.51.0.1/30
user@R1# set ge-2/0/5 description to-Host
user@R1# set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set ge-2/0/8 description to-R2
user@R1# set ge-2/0/8 unit 0 family inet address 10.50.0.1/30
```

2. Configure the policer to rate-limit to a bandwidth of 700 Mbps and a burst size of 15 KBps.

```
[edit firewall policer discard]
user@R1# set if-exceeding bandwidth-limit 700m
user@R1# set if-exceeding burst-size-limit 15k
```

3. Configure the policer to discard packets in the red traffic flow.

```
[edit firewall policer discard]
user@R1# set then discard
```

4. Configure the custom forwarding classes and associated queue numbers.

```
[edit class-of-service forwarding-classes]
user@R1# set class BE-data queue-num 0
user@R1# set class Premium-data queue-num 1
user@R1# set class Voice queue-num 2
user@R1# set class NC queue-num 3
```

5. Configure the firewall filter term that places TCP traffic with a source port of 80 (HTTP traffic) into the BE-data forwarding class, associated with queue 0.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term BE-data from protocol tcp
user@R1# set term BE-data from port http
user@R1# set term BE-data then forwarding-class BE-data
user@R1# set term BE-data then policer discard
```

6. Configure the firewall filter term that places TCP traffic with a source port of 12345 into the Premium-data forwarding class, associated with queue 1.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term Premium-data from protocol tcp
user@R1# set term Premium-data from port 12345
user@R1# set term Premium-data then forwarding-class Premium-data
user@R1# set term Premium-data then policer discard
```

7. At the end of your firewall filter, configure a default term that accepts all other traffic.

   Otherwise, all traffic that arrives on the interface that is not explicitly accepted by the firewall filter is discarded.

```
[edit firewall family inet filter mf-classifier]
user@R1# set term accept then accept
```

8. Apply the firewall filter to interface ge-2/0/8 as an output filter.

```
[edit interfaces]
user@R1# set ge-2/0/8 unit 0 family inet filter output mf-classifier
```

9. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-0/0/1.0
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

To configure R2:

1. Configure the device interfaces.

```
[edit]
user@R2# set interfaces ge-2/0/7 description to-Host
user@R2# set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
user@R2# set interfaces ge-2/0/8 description to-R1
user@R2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@R2# set interfaces lo0 unit 0 description loopback-interface
user@R2# set interfaces lo0 unit 0 family inet address 192.168.14.1/32
```

Configure OSPF.

```
[edit protocols ospf]
user@R2# set area 0.0.0.0 interface ge-2/0/7.0 passive
user@R2# set area 0.0.0.0 interface lo0.0 passive
user@R2# set area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

To configure R3:

1. Configure the interfaces.

```
[edit]
user@R3# set interfaces ge-0/0/0 description to-Host
user@R3# set interfaces ge-0/0/0 unit 0 family inet address 172.16.71.1/30
user@R3# set interfaces ge-0/0/1 description to-R1
```

```
user@R3# set interfaces ge-0/0/1 unit 0 family inet address 10.51.0.2/30
user@R3# set interfaces lo0 unit 0 description loopback-interface
user@R3# set interfaces lo0 unit 0 family inet address 192.168.15.1/32
```

2. Configure OSPF

```
[edit protocols ospf]
user@R3# set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 passive
user@R3# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@R3# set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show class-of-service`, `show firewall`, and `show protocols ospf` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1# show interfaces
ge-0/0/1 {
    description to-R3;
    unit 0 {
        family inet {
            address 10.51.0.1/30;
        }
    }
}
}
    ge-2/0/5 {
        description to-Host;
        unit 0 {
            family inet {
                address 172.16.70.2/30;
            }
        }
    }
    ge-2/0/8 {
        description to-R2;
        unit 0 {
            family inet {
                filter {
                    output mf-classifier;
                }
                address 10.50.0.1/30;
            }
        }
    }
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
```

```
user@R1# **show class-of-service**
forwarding-classes {
    class BE-data queue-num 0;
    class Premium-data queue-num 1;
    class Voice queue-num 2;
    class NC queue-num 3;
}
```

```
user@R1# **show firewall**
family inet {
    filter mf-classifier {
        term BE-data {
            from {
                protocol tcp;
                port http;
            }
            then {
                policer discard;
                forwarding-class BE-data;
            }
        }
        term Premium-data {
            from {
                protocol tcp;
                port 12345;
            }
            then {
                policer discard;
                forwarding-class Premium-data;
            }
        }
        term accept {
            then accept;
        }
    }
}
policer discard {
    if-exceeding {
        bandwidth-limit 700m;
        burst-size-limit 15k;
    }
    then discard;
}
```

```
user@R1# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/5.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-0/0/1.0;
    interface ge-2/0/8.0;
}
```

If you are done configuring R1, enter `commit` from configuration mode.

```
user@R2# show interfaces
ge-2/0/7 {
    description to-Host;
    unit 0 {
        family inet {
            address 172.16.80.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.50.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.14.1/32;
        }
    }
}
```

```
user@R2# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/7.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring R2, enter `commit` from configuration mode.

```
user@R3# show interfaces
ge-0/0/0 {
    description to-Host;
    unit 0 {
        family inet {
            address 172.16.71.2/30;
        }
    }
}
ge-0/0/1 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.51.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.15.1/32;
        }
    }
}
```

```
user@R3# show protocols ospf
area 0.0.0.0 {
    interface ge-0/0/0.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-0/0/1.0;
}
```

If you are done configuring R3, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Checking the CoS Settings

Purpose

Confirm that the forwarding classes are configured correctly.

Action

From R1, run the `show class-of-service forwarding-class` command.

```
user@R1> show class-of-service forwarding-class

Forwarding class                       ID      Queue  Restricted queue  Fabric priority
Policing priority   SPU priority
  BE-data                               0       0          0              low
normal            low
  Premium-data                          1       1          1              low
normal            low
  Voice                                 2       2          2              low
normal            low
  NC                                    3       3          3              low
normal            low
```

Meaning

The output shows the configured custom classifier settings.

Clearing the Counters

Purpose

Confirm that the firewall and interface counters are cleared.

Action

- On R1, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@R1> clear firewall all
```

- On R1, run the `clear interface statistics ge-2/0/5` command to reset the interface counters to 0.

```
user@R1> clear interface statistics ge-2/0/8
```

Sending Traffic into the Network from TCP HTTP Port 80 and Monitoring the Results

Purpose

Send traffic that can monitored at the policer and custom queue level.

Action

1. Use a traffic generator to send 20 TCP packets with a source port of 80 into the network.

   The `-s` flag sets the source port. The `-k` flag causes the source port to remain steady at 80 instead of incrementing. The `-c` flag sets the number of packets to 20. The `-d` flag sets the packet size.

   NOTE: In this example the policer numbers are reduced to a bandwidth limit of 8 Kbps and a burst size limit of 1500 KBps to ensure that some packets are dropped.

```
[User@host]#  hping 172.16.80.1  -c 20 -s 80  -k -d 300

[User@Host]# hping 172.16.80.1 -s 80 -k -c 20 -d 375
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 375 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=1001.0 ms
.
.
.
--- 172.16.80.1 hping statistic ---
20 packets transmitted, 14 packets received, 30% packet loss
round-trip min/avg/max = 1001.0/10287.1/19002.1 ms
```

2. On R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall

Filter: mf-classifier
Policers:
Name                                                Bytes              Packets
discard-BE-data                                      2490                6
discard-Premium-data                                    0                0
```

   Notice that in the `hping` output that there was 30% packet loss (6 packets out of 20) and the same number of packets was dropped by the policer as shown in the output of the `show firewall` command. Also notice that the drops are associated with the queue `BE-data` as specified in the `mf-classifier` in the firewall configuration.

3. On R1, check the queue counters by using the `show interfaces extensive ge-2/0/8| find "Queue counters"` command.

```
user@R1> show interfaces extensive ge-2/0/8| find "Queue counters"
 Queue counters:       Queued packets  Transmitted packets      Dropped packets
 0                               14                   14                    0
 1                                0                    0                    0
 2                                0                    0                    0
 3                               16                   16                    0
 Queue number:       Mapped forwarding classes
 0                   BE-data
 1                   Premium-data
 2                   Voice
 3                   NC
```

   Notice that 14 packets were transmitted out interface 2/0/8 using the queue `BE-data` as specified in the `mf-classifier` in the firewall configuration. The remaining 6 packets were dropped by the policer, as shown above. The 16 packets sent to queue 3 are network control traffic. They are possibly routing protocol updates.

Meaning

The output from both devices shows that 6 packets were discarded This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 KBps burst option for red (out-of-contract HTTP port 80) traffic was exceeded. In Steps 2 and 3, you can see that the correct queues were used to transmit the remaining traffic out interface 2/0/8.

Sending Traffic into the Network from TCP Port 12345 and Monitoring the Results

Purpose

Send traffic that can monitored at the policer and custom queue level.

Action

1. Clear the counters again as shown in section "Clearing the Counters" on page 240.

2. Use a traffic generator to send 20 TCP packets with a source port of 12345 into the network.

   The `-s` flag sets the source port. The `-k` flag causes the source port to remain steady at 12345 instead of incrementing. The `-c` flag sets the number of packets to 20. The `-d` flag sets the packet size.

```
[User@host]#  hping 172.16.80.1  -c 20 -s 12345  -k -d 300
[Host@User]# hping 172.16.80.1 -s 12345 -k -c 20 -d 375
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 375 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=1000.4 ms
.
.
.

--- 172.16.80.1 hping statistic ---
20 packets transmitted, 13 packets received, 35% packet loss
round-trip min/avg/max = 1000.4/10924.5/19002.2 ms
```

3. On R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall
Filter: mf-classifier
Policers:
Name                                              Bytes              Packets
discard-BE-data                                       0                 0
discard-Premium-data                               2905                 7
```

   Notice that in the `hping` output that there was 35% packet loss (7 packets out of 20) and the same number of packets were dropped by the policer as shown in the output of the `show firewall` command.

Also notice that the drops are associated with the queue `Premium-data` as specified in the `mf-classifier` in the firewall configuration.

4. On R1, check the queue counters by using the `show interfaces extensive ge-2/0/8| find "Queue counters"` command.

```
user@R1> show interfaces extensive ge-2/0/8| find "Queue counters"
Queue counters:       Queued packets  Transmitted packets      Dropped packets
 0                                0                  0                    0
 1                               13                 13                    0
 2                                0                   0                   0
 3                               16                 16                    0
 Queue number:         Mapped forwarding classes
 0                     BE-data
 1                     Premium-data
 2                     Voice
 3                     NC
```

Notice that 13 packets were transmitted out interface 2/0/8 using the Premium-data queues specified in the `mf-classifier` in the firewall configuration. The remaining 7 packets were dropped by the policer, as shown above. The 16 packets sent to queue 3 are network control traffic. They are possibly routing protocol updates.

Meaning

The output from both devices shows that 7 packets were discarded. This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 KBps burst option for red (out-of-contract HTTP port 80) traffic was exceeded. In Steps 3 and 4, you can see that the correct queues were used to transmit the remaining traffic out interface 2/0/8.

RELATED DOCUMENTATION

Routing Policies, Firewall Filters, and Traffic Policers User Guide

Example: Configuring a Two-Rate Three-Color Policer

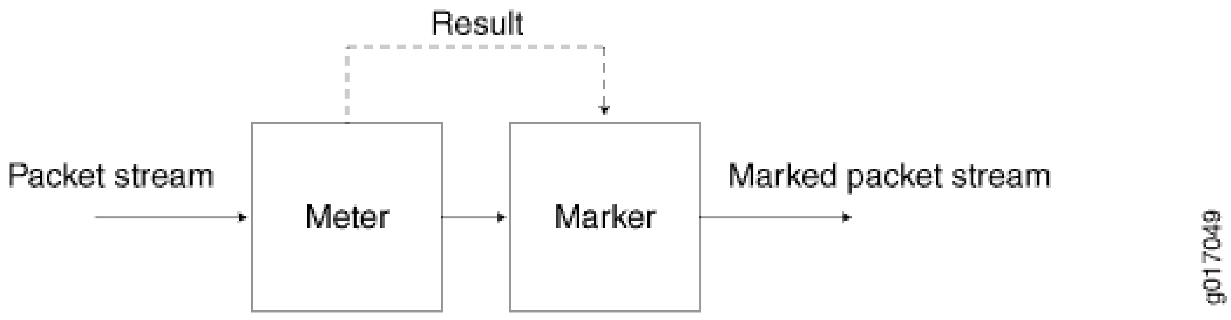
Overview of Tricolor Marking (TCM) Architecture

IN THIS SECTION

Policers provide two functions: metering and marking.

The policer meters each packet and passes the packet and the metering result to the marker, as shown in Figure 30 on page 246.

Figure 30: Flow of Tricolor Marking Policer Operation



The meter operates in two modes. In the color-blind mode, the meter treats the packet stream as uncolored. Any preset loss priorities are ignored. In the color-aware mode, the meter inspects the *PLP* field, which has been set by an upstream device as high, medium-high, medium-low, or low. In other words, the PLP field has already been set by a BA or multifield classifier. The marker changes the PLP of each incoming IP packet according to the results of the meter. For more information, see "Configuring Two-Rate Tricolor Marking" on page 259.

Single-rate TCM is so called because traffic is policed according to one rate—the committed information rate (CIR)—and two burst sizes: the committed burst size (CBS) and excess burst size (EBS). The CIR specifies the average rate at which bits are admitted to the network. The CBS specifies the usual burst size in bytes admitted to the network. The EBS specifies the maximum burst size in bytes for packets admitted to the network. The EBS is greater than or equal to the CBS, and neither can be 0. As each packet enters the network, its bytes are counted. Packets that do not exceed the CBS are marked low PLP. Packets that exceed the CBS but are below the EBS are marked medium-high PLP. Packets that exceed the EBS are marked high PLP.

Two-rate TCM is so called because traffic is policed according to two rates: the CIR and the peak information rate (PIR). The PIR is greater than or equal to the CIR. The PIR specifies the maximum rate at which bits are admitted to the network. As each packet enters the network, its bits are counted. Bits in packets that do not exceed the CIR have their packets marked low PLP. Bits in packets that exceed the CIR but are below the PIR have their packets marked medium-high PLP. Bits in packets that exceed the PIR have their packets marked high PLP.

For information about how to use marking policers with BA and multifield classifiers, see "Configuring Behavior Aggregate Classifiers" on page 80 and "Using Multifield Classifiers to Set Packet Loss Priority" on page 153.

Platform-Specific TCM Policer Behavior

Use Feature Explorer to confirm platform and release support.

Use the following table to review platform-specific behaviors for your platform:

Platform	Difference
MX Series	• On MX Series, you can apply three-color policers to aggregated interfaces.

RELATED DOCUMENTATION

Configure and Apply TCM Policers | 248

Configure and Apply TCM Policers

IN THIS SECTION

A tricolor marking (TCM) policer polices traffic on the basis of metering rates, including the CIR, the PIR, their associated burst sizes, and any policing actions configured for the traffic.

This topic describes how to configure and apply TCM policers.

Define a Tricolor Marking Policer

To configure a TCM policer, first enable tricolor marking if not already enabled:

```
[edit]
user@host# edit class-of-service
user@host#  set tri-color
```

You can configure a tricolor policer to discard high loss priority traffic on a logical interface in the ingress or egress direction.

You can specify the values for bps and bytes either as complete decimal numbers or as decimal numbers followed by the abbreviation `k` (1000), `m` (1,000,000), or `g` (1,000,000,000).

The color-blind policer implicitly marks packets into three loss priority categories:

- Low
- Medium-high
- High

NOTE: In a single *firewall filter* term, you cannot configure both the `loss-priority` action modifier and the `three-color-policer` action modifier. These statements are mutually exclusive.

Table 16 on page 249 describes all the configurable TCM statements.

Table 16: TCM Policer Statements

Statement	Meaning	Configurable Values
`single-rate`	Marking is based on the CIR, CBS, and EBS.	–
`two-rate`	Marking is based on the CIR, PIR, and rated burst sizes.	–
`color-aware`	Metering depends on the packet's preclassification. Metering can increase a packet's assigned PLP, but cannot decrease it.	–
`color-blind`	All packets are evaluated by the CIR or CBS. If a packet exceeds the CIR or CBS, it is evaluated by the PIR or EBS.	–
`committed-information-rate`	Guaranteed bandwidth under normal line conditions and the average rate up to which packets are marked green.	1500 through 100,000,000,000 bps
`committed-burst-size`	Maximum number of bytes allowed for incoming packets to burst above the CIR, but still be marked green.	1500 through 100,000,000,000 bytes
`excess-burst-size`	Maximum number of bytes allowed for incoming packets to burst above the CIR, but still be marked yellow.	1500 through 100,000,000,000 bytes
`peak-information-rate`	Maximum achievable rate. Packets that exceed the CIR but are below the PIR are marked yellow. Packets that exceed the PIR are marked red.	1500 through 100,000,000,000 bps

Table 16: TCM Policer Statements *(Continued)*

Statement	Meaning	Configurable Values
`peak-burst-size`	Maximum number of bytes allowed for incoming packets to burst above the PIR, but still be marked yellow.	1500 through 100,000,000,000 bytes

Define the TCM policer at the `[edit firewall]` hierarchy level:

1. Create the TCM policer by defining a name for the policer.

```
[edit]
user@host# edit firewall three-color-policer three-color-policer-name
```

2. Discard traffic on a logical interface using tricolor marking policing.

```
[edit firewall three-color-policer name]
user@host# set action loss-priority high then discard
```

3. Define the filter as a logical interface policer.

```
[edit firewall three-color-policer name]
user@host# set logical-interface-policer
```

4. Configure a single-rate three-color policer in which marking is based on the committed information rate (CIR), committed burst size (CBS), and excess burst size (EBS).

```
[edit firewall three-color-policer name]
user@host# set single-rate (color-aware | color-blind)
user@host# set single-rate committed-information-rate bps
user@host# set single-rate committed-burst-size bytes
user@host# set single-rate excess-burst-size bytes
```

5. Configure a two-rate three-color policer in which marking is based on the committed information rate (CIR), committed burst size (CBS), peak information rate (PIR), and peak burst size (PBS).

```
[edit firewall three-color-policer name]
user@host# set two-rate (color-aware | color-blind)
```

```
user@host# set two-rate committed-information-rate bps
user@host# set two-rate committed-burst-size bytes
user@host# set two-rate peak-information-rate bps
user@host# set two-rate peak-burst-size bytes
```

6. Confirm the configuration.

```
[edit firewall]
user@host# show
```

```
three-color-policer name {
    action {
        loss-priority high then discard; # Only for IQ2 PICs
    }
    logical-interface-policer;
    single-rate {
        (color-aware | color-blind);
        committed-information-rate bps;
        committed-burst-size bytes;
        excess-burst-size bytes;
    }
    two-rate {
        (color-aware | color-blind);
        committed-information-rate bps;
        committed-burst-size bytes;
        peak-information-rate bps;
        peak-burst-size bytes;
    }
}
```

7. Save the configuration.

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

Apply TCM Policers to Firewall Filters

To rate-limit traffic by applying a TCM policer to a firewall filter:

- Set the three-color-policer statement at the edit firewall hierarchy level:

  ```
  [edit]
  user@host# edit firewall
  user@host# set three-color-policer  three-color-policer-name
  ```

You can include this statement at the following hierarchy levels:

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

In the family statement, the protocol family can be any, ccc, inet, inet6, mpls, or vpls.

You must identify the referenced policer as a single-rate or two-rate policer, and this statement must match the configured TCM policer. Otherwise, an error message appears in the configuration listing.

For example, if you configure srTCM as a single-rate TCM policer and try to apply it as a two-rate policer, the following message appears:

```
[edit firewall]
user@host# show three-color-policer srTCM
single-rate {
    color-aware;
    . . .
}
user@host# show filter TESTER
term A {
    then {
        three-color-policer {
            ##
            ## Warning: Referenced two-rate policer does not exist
            ##
            two-rate srTCM;
        }
    }
}
```

Apply Firewall Filter TCM Policers to Interfaces

To apply a TCM policer to an interface, you must reference the filter name in the interface configuration.

- Set the `filter` statement:

```
[edit]
user@host# edit interfaces interface-name unit logical-unit-number family family
user@host# set filter input filter-name
user@host# set filter output filter-name
```

> **NOTE**: The filter name that you reference must have an attached tricolor marking policer.

You can include these statements at the following hierarchy levels:

- `[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]`

Example: Configure and Apply a Single-Rate TCM Policer

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

This example describes how to configure and apply a color-blind, single-rate, TCM policer.

1. Configure the single-rate, color-blind, TCM policer.

```
[edit]
user@host# edit firewall three-color-policer srtcm1-cb single-rate
user@host# set color-blind
user@host# set committed-information-rate 1048576
user@host# set committed-burst-size 65536
user@host# excess-burst-size 131072
```

2. Apply the policer to the `fil` firewall filter.

```
[edit firewall]
user@host# set filter fil term default then three-color-policer single-rate srtc1-cb
```

3. Apply the `fil` firewall filter to the logical interface:

```
[edit]
user@host# edit interfaces xe-1/0/0 unit 0
user@host# set family inet filter input fil
```

4. Verify the configuration.

```
[edit firewall]
user@host# show
```

```
three-color-policer srtcm1-cb {
    single-rate {
        color-blind;
        committed-information-rate 1048576;
        committed-burst-size 65536;
        excess-burst-size 131072;
    }
}
filter fil {
    term default {
        then {
            three-color-policer {
                single-rate srtcm1-cb;
            }
        }
    }
}
```

```
[edit interfaces]
user@host# show
```

```
xe-1/0/0 {
    unit 0 {
        family inet {
            filter {
                input fil;
            }
        }
    }
}
```

5. Save the configuration.

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

RELATED DOCUMENTATION

Controlling Network Access Using Traffic Policing Overview

Overview of Tricolor Marking (TCM) Architecture | 245

Configuring Single-Rate TCM Policers

IN THIS SECTION

With TCM, you can configure traffic policing according to two separate modes—color-blind and color-aware. In color-blind mode, the current PLP value is ignored. In color-aware mode, the current PLP values are considered by the policer and can only be increased.

This topic describes how to configure each mode for single-rate TCM and includes the following sections:

Configuring Color-Blind Mode for Single-Rate TCM

All packets are evaluated by the CBS. If a packet exceeds the CBS, it is evaluated by the EBS. In color-blind mode, the policer supports three loss priorities only: low, medium-high, and high.

In color-blind mode, packets that exceed the CBS but are below the EBS are marked yellow (medium-high). Packets that exceed the EBS are marked red (high), as shown in Table 17 on page 256.

Table 17: Color-Blind Mode TCM Color-to-PLP Mapping

Color	PLP	Meaning
Green	`low`	Packet does not exceed the CBS.
Yellow	`medium-high`	Packet exceeds the CBS but does not exceed the EBS.
Red	`high`	Packet exceeds the EBS.

If you are using color-blind mode and you wish to configure an output policer that marks packets to have medium-low loss priority, you must configure a policer at the `[edit firewall policer` *`policer-name`*`]` hierarchy level. For example:

```
firewall {
    policer 4PLP {
        if-exceeding {
            bandwidth-limit 40k;
            burst-size-limit 4k;
        }
        then loss-priority medium-low;
    }
}
```

Apply this policer at one or both of the following hierarchy levels:

- `[edit firewall family` *`family`* `filter` *`filter-name`* `term` *`rule-name`* `then policer` *`policer-name`*`]`
- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `family` *`family`* `filter` *`filter-name`*`]`

Configuring Color-Aware Mode for Single-Rate TCM

In color-aware mode, the metering treatment the packet receives depends on its classification. Metering can increase a packet's preassigned PLP, but cannot decrease it, as shown in Table 18 on page 257.

Table 18: Color-Aware Mode TCM PLP Mapping

Incoming PLP	Packet Metered Against	Possible Cases	Outgoing PLP
`low`	CBS and EBS	Packet does not exceed the CBS.	`low`
		Packet exceeds the CBS but not the EBS.	`medium-high`
		Packet exceeds the EBS.	`high`
`medium-low`	EBS only	Packet does not exceed the CBS.	`medium-low`
		Packet does not exceed the EBS.	`medium-low`
		Packet exceeds the EBS.	`high`
`medium-high`	EBS only	Packet does not exceed the CBS.	`medium-high`
		Packet does not exceed the EBS.	`medium-high`
		Packet exceeds the EBS.	`high`
`high`	Not metered by the policer.	All cases.	`high`

The following sections describe single-rate color-aware PLP mapping in more detail.

Effect on Low PLP of Single-Rate Policer

Packets belonging to the green class have already been marked by a classifier with low PLP. The marking policer can leave the packet's PLP unchanged or increase the PLP to medium-high or high. Therefore, these packets are metered against both the CBS and the EBS.

For example, if a BA or multifield classifier marks a packet with low PLP, and the two-rate TCM policer is in color-aware mode, the output loss priority is as follows:

- If the rate of traffic flow is less than the CBS, packets remain marked as low PLP.

- If the rate of traffic flow is greater than the CBS but less than the EBS, some of the packets are marked as medium-high PLP, and some of the packets remain marked as low PLP.
- If the rate of traffic flow is greater than the EBS, some of the packets are marked as high PLP, and some of the packets remain marked as low PLP.

Effect on Medium-Low PLP of Single-Rate Policer

Packets belonging to the yellow class have already been marked by a classifier with medium-low or medium-high PLP. The marking policer can leave the packet's PLP unchanged or increase the PLP to high. Therefore, these packets are metered against the EBS only.

For example, if a BA or multifield classifier marks a packet with medium-low PLP, and the two-rate TCM policer is in color-aware mode, the output loss priority is as follows:

- If the rate of traffic flow is less than the CBS, packets remain marked as medium-low PLP.
- If the rate of traffic flow is greater than the CBS but less than the EBS, packets remain marked as medium-low PLP.
- If the rate of traffic flow is greater than the EBS, some of the packets are marked as high PLP, and some of the packets remain marked as medium-low PLP.

Effect on Medium-High PLP of Single-Rate Policer

Packets belonging to the yellow class have already been marked by a classifier with medium-low or medium-high PLP. The marking policer can leave the packet's PLP unchanged or increase the PLP to high. Therefore, these packets are metered against the EBS only.

For example, if a BA or multifield classifier marks a packet with medium-high PLP, and the two-rate TCM policer is in color-aware mode, the output loss priority is as follows:

- If the rate of traffic flow is less than the CBS, packets remain marked as medium-high PLP.
- If the rate of traffic flow is greater than the CBS but less than the EBS, packets remain marked as medium-high PLP.
- If the rate of traffic flow is greater than the EBS, some of the packets are marked as high PLP, and some of the packets remain marked as medium-high PLP.

Effect on High PLP of Single-Rate Policer

Packets belonging to the red class have already been marked by a classifier with high PLP. The marking policer can only leave the packet's PLP unchanged. Therefore, these packets are not metered against the CBS or the EBS and all the packets remain marked as high PLP.

RELATED DOCUMENTATION

Configuring Two-Rate TCM Policers

IN THIS SECTION

With TCM, you can configure traffic policing according to two separate modes—color-blind and color-aware. In color-blind mode, the current PLP value is ignored. In color-aware mode, the current PLP values are considered by the policer and can only be increased.

This topic describes how to configure each mode for two-rate TCM and includes the following sections:

Configuring Color-Blind Mode for Two-Rate TCM

All packets are evaluated by the CIR. If a packet exceeds the CIR, it is evaluated by the PIR. In color-blind mode, the policer supports three loss priorities only: low, medium-high, and high.

In color-blind mode, packets that exceed the CIR but are below the PIR are marked yellow (medium-high). Packets that exceed the PIR are marked red (high), as shown in Table 19 on page 259.

Table 19: Color-Blind Mode TCM Color-to-PLP Mapping

Color	PLP	Meaning
Green	`low`	Packet does not exceed the CIR.
Yellow	`medium-high`	Packet exceeds the CIR but does not exceed the PIR.
Red	`high`	Packet exceeds the PIR.

If you are using color-blind mode and you want to configure an output policer that marks packets to have medium-low loss priority, you must configure a policer at the `[edit firewall policer` *`policer-name`*`]` hierarchy level. For example:

```
firewall {
    policer 4PLP {
        if-exceeding {
            bandwidth-limit 40k;
            burst-size-limit 4k;
        }
        then loss-priority medium-low;
    }
}
```

Apply this policer at one or both of the following hierarchy levels:

- `[edit firewall family` *`family`* `filer` *`filter-name`* `term` *`rule-name`* `then policer` *`policer-name`*`]`
- `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `family` *`family`* `filter` *`filter-name`*`]`

Configuring Color-Aware Mode for Two-Rate TCM

In color-aware mode, the metering treatment the packet receives depends on its classification. Metering can increase a packet's preassigned PLP, but cannot decrease it, as shown in Table 20 on page 260.

Table 20: Color-Aware Mode TCM Mapping

Incoming PLP	Packet Metered Against	Possible Cases	Outgoing PLP	Outgoing PLP (MPCs Only)
`low`	CIR and PIR	Packet does not exceed the CIR.	`low`	`low`
		Packet exceeds the CIR but not the PIR.	`medium-high`	`medium-high`
		Packet exceeds the PIR.	`high`	`high`
`medium-low`	PIR only	Packet does not exceed the CIR.	`medium-low`	`medium-high`

Table 20: Color-Aware Mode TCM Mapping *(Continued)*

Incoming PLP	Packet Metered Against	Possible Cases	Outgoing PLP	Outgoing PLP (MPCs Only)
		Packet does not exceed the PIR.	`medium-low`	`medium-high`
		Packet exceeds the PIR.	`high`	`high`
`medium-high`	PIR only	Packet does not exceed the CIR.	`medium-high`	`medium-high`
		Packet does not exceed the PIR.	`medium-high`	`medium-high`
		Packet exceeds the PIR.	`high`	`high`
`high`	Not metered by the policer.	All cases.	`high`	`high`

The following sections describe color-aware two-rate PLP mapping in more detail.

Effect on Low PLP of Two-Rate Policer

Packets belonging to the green class have already been marked by a classifier with low PLP. The marking policer can leave the packet’s PLP unchanged or increase the PLP to medium-high or high. Therefore, these packets are metered against both the CIR and the PIR.

For example, if a BA or multifield classifier marks a packet with low PLP, and the two-rate TCM policer is in color-aware mode, the output loss priority is as follows:

- If the rate of traffic flow is less than the CIR, packets remain marked as low PLP.
- If the rate of traffic flow is greater than the CIR but less than the PIR, some of the packets are marked as medium-high PLP, and some of the packets remain marked as low PLP.
- If the rate of traffic flow is greater than the PIR, some of the packets are marked as high PLP, and some of the packets remain marked as low PLP.

Effect on Medium-Low PLP of Two-Rate Policer

Packets belonging to the yellow class have already been marked by a classifier with medium-low or medium-high PLP. The marking policer can leave the packet's PLP unchanged or increase the PLP to high. Therefore, these packets are metered against the PIR only.

For example, if a BA or multifield classifier marks a packet with medium-low PLP, and the two-rate TCM policer is in color-aware mode, the output loss priority is as follows:

- If the rate of traffic flow is less than the CIR, packets remain marked as medium-low PLP. (MPCs mark the packets as medium-high.)
- If the rate of traffic flow is greater than the CIR/CBS but less than the PIR, packets remain marked as medium-low PLP. (MPCs mark the packets as medium-high.)
- If the rate of traffic flow is greater than the PIR, some of the packets are marked as high PLP, and some of the packets remain marked as medium-low PLP.

Effect on Medium-High PLP of Two-Rate Policer

Packets belonging to the yellow class have already been marked by a classifier with medium-low or medium-high PLP. The marking policer can leave the packet's PLP unchanged or increase the PLP to high. Therefore, these packets are metered against the PIR only.

For example, if a BA or multifield classifier marks a packet with medium-high PLP, and the two-rate TCM policer is in color-aware mode, the output loss priority is as follows:

- If the rate of traffic flow is less than the CIR, packets remain marked as medium-high PLP.
- If the rate of traffic flow is greater than the CIR but less than the PIR, packets remain marked as medium-high PLP.
- If the rate of traffic flow is greater than the PIR, some of the packets are marked as high PLP, and some of the packets remain marked as medium-high PLP.

Effect on High PLP of Two-Rate Policer

Packets belonging to the red class have already been marked by a classifier with high PLP. The marking policer can only leave the packet's PLP unchanged. Therefore, these packets are not metered against the CIR or the PIR and all the packets remain marked as high PLP.

RELATED DOCUMENTATION

Configure and Apply TCM Policers | 248

Example: Configure and Verify Two-Rate TCM Policers

IN THIS SECTION

This topic provides several examples of how you can configure and verify two-rate TCM policers.

Requirements

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

Overview

This example configures a two-rate tricolor marking policer on an input interface and shows commands to verify its operation.

Configuration

IN THIS SECTION

To configure two-rate tricolor marking policers, perform these tasks:

CLI Quick Configuration

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

Applying a Policer to an Input Interface

```
set interfaces xe-1/2/1 unit 0 family inet filter input trtcm-filter
set firewall three-color-policer trtcm1 two-rate color-aware
set firewall three-color-policer trtcm1 two-rate committed-information-rate 100m
set firewall three-color-policer trtcm1 two-rate committed-burst-size 65536
set firewall three-color-policer trtcm1 two-rate peak-information-rate 200m
set firewall three-color-policer trtcm1 two-rate peak-burst-size 131072
set firewall filter trtcm-filter term one then three-color-policer two-rate trtcm1
```

Applying Profiles to an Output Interface

```
set class-of-service drop-profiles low-tcm fill-level 80 drop-probability 100
set class-of-service drop-profiles med-tcm fill-level 40 drop-probability 100
set class-of-service drop-profiles high-tcm fill-level 10 drop-probability 100
set class-of-service tri-color
set class-of-service interfaces ge-1/1/0 scheduler-map tcm-sched
set class-of-service scheduler-maps tcm-sched forwarding-class queue-0 scheduler q0-sched
set class-of-service scheduler-maps tcm-sched forwarding-class queue-3 scheduler q3-sched
set class-of-service schedulers q0-sched transmit-rate percent 50
set class-of-service schedulers q0-sched buffer-size percent 50
set class-of-service schedulers q0-sched drop-profile-map loss-priority low protocol any drop-
profile low-tcm
set class-of-service schedulers q0-sched drop-profile-map loss-priority medium-high protocol any
drop-profile med-tcm
set class-of-service schedulers q0-sched drop-profile-map loss-priority high protocol any drop-
profile high-tcm
set class-of-service schedulers q3-sched transmit-rate percent 50
set class-of-service schedulers q3-sched buffer-size percent 50
```

Marking Packets with Medium-Low Loss Priority

```
set interfaces xe-1/2/1 unit 0 family inet filter input 4PLP
set interfaces xe-1/2/1 unit 0 family inet policer input 4PLP
set interfaces xe-1/2/1 unit 0 family inet address 10.45.10.2/30
set firewall three-color-policer trTCM two-rate color-blind
set firewall three-color-policer trTCM two-rate committed-information-rate 400m
set firewall three-color-policer trTCM two-rate committed-burst-size 100m
set firewall three-color-policer trTCM two-rate peak-information-rate 1g
set firewall three-color-policer trTCM two-rate peak-burst-size 500m
set firewall policer 4PLP if-exceeding bandwidth-limit 40k
set firewall policer 4PLP if-exceeding burst-size-limit 4k
set firewall policer 4PLP then loss-priority medium-low
set firewall family inet filter 4PLP term 0 from precedence 1
set firewall family inet filter 4PLP term 0 then loss-priority medium-low
set firewall family inet filter filter_trTCM term default then three-color-policer two-rate trTCM
```

Example: Applying a Policer to an Input Interface

Step-by-Step Procedure

In the following example, the tricolor marking and policer are applied on the ingress interface. Incoming packets are metered. Packets that do not exceed the CIR are marked with low loss priority. Packets that exceed the CIR, but do not exceed the PIR, are marked with medium-high loss priority. Packets that exceed the PIR are marked with high loss priority.

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

1. Configure the three-color policer.

```
[edit]
user@host# edit firewall three-color-policer trtcm1 two-rate
user@host# set committed-information-rate 100m
user@host# set committed-burst-size 65536
user@host# set peak-information-rate 200m
user@host# set peak-burst-size 131072
```

2. Configure the policer in a firewall filter.

```
[edit]
user@host# set firewall filter trtcm-filter term one then three-color-policer two-rate trtcm1
```

3. Apply the firewall filter (policer) as an input filter on the logical interface.

```
[edit]
user@host# edit interfaces xe-1/2/1 unit 0 family inet
user@host# set filter input trtcm-filter
```

4. Confirm the configuration.

```
[edit]
user@host# show
```

```
interfaces {
xe-1/2/1 {
    unit 0 {
        family inet {
            filter {
                input trtcm-filter;
            }
        }
    }
}
firewall {
 three-color-policer trtcm1 {
        two-rate {
            color-aware;
            committed-information-rate 100m;
            committed-burst-size 65536;
            peak-information-rate 200m;
            peak-burst-size 131072;
        }
    }

 filter trtcm-filter {
        term one {
            then {
                three-color-policer {
                    two-rate trtcm1;
                }
            }
        }
    }
}
```

5. Save the configuration.

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

Example: Applying Profiles to an Output Interface

Step-by-Step Procedure

In the following example, transmission scheduling and weighted random early detection (WRED) profiles are applied on the output interface. The software drops traffic in the low, medium-high, and high drop priorities proportionally to the configured drop profiles.

1. Define the drop profile.

```
[edit]
user@host# edit class-of-service
user@host# set drop-profiles low-tcm fill-level 80 drop-probability 100
user@host# set drop-profiles med-tcm fill-level 40 drop-probability 100
user@host# set drop-profiles high-tcm fill-level 10 drop-probability 100
user@host# set tri-color
```

2. Specify the scheduler name and parameter values.

```
[edit class-of-service]
user@host# set schedulers q0-sched transmit-rate percent 50
user@host# set schedulers q0-sched buffer-size percent 50
user@host# set schedulers q0-sched drop-profile-map loss-priority low protocol any drop-
profile low-tcm
user@host# set schedulers q0-sched drop-profile-map loss-priority medium-high protocol any
drop-profile med-tcm
user@host# set schedulers q0-sched drop-profile-map loss-priority high protocol any drop-
profile high-tcm
user@host# set schedulers q3-sched transmit-rate percent 50
user@host# set schedulers q3-sched buffer-size percent 50
```

3. Specify a scheduler map name and associate it with the scheduler configuration and forwarding class.

```
[edit class-of-service]
user@host# set scheduler-maps tcm-sched forwarding-class queue-0 scheduler q0-sched
user@host# set scheduler-maps tcm-sched forwarding-class queue-3 scheduler q3-sched
```

4. Apply the scheduler map to the interface.

```
[edit class-of-service]
user@host# set interfaces ge-1/1/0 scheduler-map tcm-sched
```

5. Verify the configuration.

```
                        [edit class-of-service]

                        user@host show


drop-profiles {
    low-tcm {
        fill-level 80 drop-probability 100;
    }
    med-tcm {
        fill-level 40 drop-probability 100;
    }
    high-tcm {
        fill-level 10 drop-probability 100;
    }
}
tri-color;
interfaces {
    so-1/1/0 {
        scheduler-map tcm-sched;
    }
scheduler-maps {
    tcm-sched {
        forwarding-class queue-0 scheduler q0-sched;
        forwarding-class queue-3 scheduler q3-sched;
    }
}
schedulers {
    q0-sched {
        transmit-rate percent 50;
        buffer-size percent 50;
        drop-profile-map loss-priority low protocol any drop-profile low-tcm;
        drop-profile-map loss-priority medium-high protocol any drop-profile med-tcm;
        drop-profile-map loss-priority high protocol any drop-profile high-tcm;
    }
    q3-sched {
        transmit-rate percent 50;
        buffer-size percent 50;
    }
}
```

6. Save the configuration.

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

Example: Marking Packets with Medium-Low Loss Priority

Step-by-Step Procedure

In the following example, the 4PLP filter and policer causes certain packets to be marked with medium-low loss priority.

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

1. Configure the firewall filter.

   a. Define the three-color policer.

   ```
   [edit]
   user@host# edit firewall three-color-policer trTCM two-rate
   user@host# set color-blind
   user@host# set committed-information-rate 400m
   ```

```
user@host# set committed-burst-size 100m
user@host# set peak-information-rate 1g
user@host# set peak-burst-size 500m
```

b. Configure policer rate limits and actions.

```
[edit]
user@host# edit firewall policer 4PLP
user@host# set if-exceeding bandwidth-limit 40k
user@host# set if-exceeding burst-size-limit 4k
user@host# set then loss-priority medium-low
```

c. Configure the IPv4 firewall filter.

```
[edit]
user@host# edit firewall family inet filter 4PLP term 0
user@host# set from precedence 1
user@host# set then loss-priority medium-low
```

d. Define the terms of the IPv4 firewall filter.

```
[edit]
user@host# edit firewall family inet filter filter_trTCM
user@host# set term default then three-color-policer two-rate trTCM
```

2. Apply the filter to the interface.

```
[edit]
user@host# edit interfaces xe-1/2/1 unit 0 family inet
user@host# set filter input 4PLP
user@host# set policer input 4PLP
user@host# set address 10.45.10.2/30
```

Results

Confirm your configuration by entering the `show interfaces` and `show firewall` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show
```

```
interfaces {
 ge-1/2/1 {
        unit 0 {
            family inet {
                filter {
                    input 4PLP;
                }
                policer {
                    input 4PLP;
                }
                address 10.45.10.2/30;
            }
        }
    }
}
firewall {
  three-color-policer trTCM {
        two-rate {
            color-blind;
            committed-information-rate 400m;
            committed-burst-size 100m;
            peak-information-rate 1g;
            peak-burst-size 500m;
        }
    }
  policer 4PLP {
        if-exceeding {
            bandwidth-limit 40k;
            burst-size-limit 4k;
        }
        then loss-priority medium-low;
    }
  family inet {
    filter 4PLP {
            term 0 {
                from {
                    precedence 1;
                }
                then loss-priority medium-low;
            }
        }
    filter trtcm-filter {
        term one {
            then {
                three-color-policer {
                    two-rate trtcm1;
                }
            }
        }
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Verifying Two-Rate Tricolor Marking Operation

Action

The following operational mode commands are useful for checking the results of your configuration:

- `show class-of-service forwarding-table classifiers`
- `show interfaces` *`interface-name`* `extensive`
- `show interfaces queue` *`interface-name`*

For information about these commands, see the CLI Explorer.

RELATED DOCUMENTATION

Configuring Two-Rate TCM Policers | 259

Guidelines for Configuring Firewall Filters

Policer Overhead to Account for Rate Shaping in the Traffic Manager

IN THIS SECTION

Policer Overhead to Account for Rate Shaping Overview

If you configure ingress or egress traffic-shaping overhead values for an interface, the traffic manager cannot apply these values to any rate-limiting also applied to the interface. To enable the router to account for the additional Ethernet frame length when policing actions are being determined, you must configure the ingress or egress overhead values for policers separately.

NOTE: When a policer overhead value is changed, the PIC or DPC goes offline and then comes back online.

On supported platforms, you can control the rate of traffic that passes through all interfaces on the PIC or DPC by configuring a *policer overhead*. You can configure a policer ingress overhead and a policer egress overhead, each with values from 0 through 255 bytes. Junos adds the policer overhead values to the length of the final Ethernet frame when determining ingress and egress policer actions.

SEE ALSO

egress-policer-overhead

ingress-policer-overhead

Example: Configure Policer Overhead to Account for Rate Shaping

IN THIS SECTION

This example shows how to configure overhead values for policers when rate-shaping overhead is configured.

Requirements

Before you begin, make sure that interface for which you are applying ingress or egress policer overhead supports this feature. Use Feature Explorer to confirm platform and release support.

Overview

IN THIS SECTION

This example shows how to configure policer overhead values for all physical interfaces on a supported PIC or MPC so that the rate shaping value configured on a logical interface is accounted for in any policing on that logical interface.

Topology

The router hosts a Gigabit Ethernet IQ2 PIC, installed in PIC location 3 of the FPC in slot number 1. The physical interface on port 1 on that PIC is configured to receive traffic on logical interface 0 and send it back out on logical interface 1. CoS scheduling includes 100 Mbps of traffic rate-shaping overhead for the output traffic. A policer egress overhead of 100 bytes is configured on the entire PIC. Thus, for any policers applied to the output traffic, 100 bytes are added to the final Ethernet frame length when determining ingress and egress policer actions.

NOTE: Traffic rate-shaping and corresponding policer overhead are configured separately:

- You configure rate shaping at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`unit-number`*`]` hierarchy level.
- You configure policer overhead at the `[edit chassis fpc` *`slot-number`* `pic` *`pic-number`*`]` hierarchy level.

When a policer overhead value is changed, the PIC or DPC goes offline and then comes back online.

Configuration

IN THIS SECTION

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

To configure this example, perform the following tasks:

CLI Quick Configuration

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the `[edit]` hierarchy level.

```
set interfaces ge-1/3/1 per-unit-scheduler
set interfaces ge-1/3/1 vlan-tagging
set interfaces ge-1/3/1 unit 0 vlan-id 100
set interfaces ge-1/3/1 unit 0 family inet address 10.10.10.1/30
set interfaces ge-1/3/1 unit 1 vlan-id 101
set interfaces ge-1/3/1 unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
```

```
00:00:11:22:33:44
set class-of-service schedulers be transmit-rate percent 5
set class-of-service schedulers ef transmit-rate percent 30
set class-of-service schedulers af transmit-rate percent 30
set class-of-service schedulers nc transmit-rate percent 35
set class-of-service scheduler-maps my-map forwarding-class best-effort scheduler be
set class-of-service scheduler-maps my-map forwarding-class expedited-forwarding scheduler ef
set class-of-service scheduler-maps my-map forwarding-class network-control scheduler nc
set class-of-service scheduler-maps my-map forwarding-class assured-forwarding scheduler af
set class-of-service interfaces ge-1/3/1 unit 1 scheduler-map my-map
set class-of-service interfaces ge-1/3/1 unit 1 shaping-rate 100m
set firewall policer 500Kbps logical-interface-policer
set firewall policer 500Kbps if-exceeding bandwidth-limit 500k
set firewall policer 500Kbps if-exceeding burst-size-limit 625k
set firewall policer 500Kbps then discard
set chassis fpc 1 pic 3 ingress-policer-overhead 100
set chassis fpc 1 pic 3 egress-policer-overhead 100
set interfaces ge-1/3/1 unit 0 family inet policer input 500Kbps
```

Configuring the Logical Interfaces

Step-by-Step Procedure

To configure the logical interfaces:

1. Enable configuration of the interface

   ```
   [edit]
   user@host# edit interfaces ge-1/3/1
   ```

2. Enable multiple queues for each logical interface (so that you can associate an output scheduler with each logical interface).

   ```
   [edit interfaces ge-1/3/1]
   user@host# set per-unit scheduler
   user@host# set vlan-tagging
   ```

NOTE: For Gigabit Ethernet IQ2 PICs only, use the `shared-scheduler` statement to enable shared schedulers and shapers on a physical interface.

3. Configure logical interface `ge-1/3/1.0`.

```
[edit interfaces ge-1/3/1]
user@host# set unit 0 vlan-id 100
user@host# set unit 0 family inet address 10.10.10.1/30
```

4. Configure logical interface `ge-1/3/1.1`.

```
[edit interfaces ge-1/3/1]
user@host# set unit 1 vlan-id 101
user@host# set unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac 00:00:11:22:33:44
```

Results

Confirm the configuration of the interfaces by entering the `show interfaces` configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/1 {
    per-unit-scheduler;
    vlan-tagging;
    unit 0 {
        vlan-id 100;
        family inet {
            address 10.10.10.1/30;
        }
    }
    unit 1 {
        vlan-id 101;
        family inet {
            address 20.20.20.1/30 {
                arp 20.20.20.2 mac 00:00:11:22:33:44;
            }
        }
    }
}
```

Configuring Traffic Rate-Shaping on the Logical Interface That Carries Output Traffic

Step-by-Step Procedure

To configure traffic rate-shaping on the logical interface that carries output traffic:

1. Enable configuration of class-of-service features.

   ```
   [edit]
   user@host# edit class-of-service
   ```

2. Configure packet scheduling on logical interface `ge-1/3/1.0`.

   - Configure schedulers that specify the percentage of transmission capacity.

     ```
     [edit class-of-service]
     user@host# edit schedulers

     [edit class-of-service schedulers]
     user@host# set be transmit-rate percent 5
     user@host# set ef transmit-rate percent 30
     user@host# set af transmit-rate percent 30
     user@host# set nc transmit-rate percent 35
     ```

     A percentage of zero drops all packets in the queue. When the `rate-limit` option is specified, the transmission rate is limited to the rate-controlled amount. In contrast with the `exact` option, a scheduler with the `rate-limit` option shares unused bandwidth above the rate-controlled amount.

   - Configure a scheduler map to associate each scheduler with a forwarding class.

     ```
     [edit class-of-service]
     user@host# edit scheduler-maps my-map

     [edit class-of-service scheduler-maps my-map]
     user@host# set forwarding-class best-effort scheduler be
     user@host# set forwarding-class expedited-forwarding scheduler ef
     ```

```
user@host# set forwarding-class network-control scheduler nc
user@host# set forwarding-class assured-forwarding scheduler af
```

- Associate the scheduler map with logical interface `ge-1/3/1.0`.

```
[edit class-of-service]
user@host# edit interfaces ge-1/3/1 unit 1

[edit class-of-service interfaces ge-1/3/1 unit 1]
user@host# set scheduler-map my-map
```

3. Configure 100 Mbps of traffic rate-shaping overhead on logical interface `ge-1/3/1.1`.

```
[edit class-of-service interfaces ge-1/3/1 unit 1]
user@host# set shaping-rate 100
```

   Alternatively, you can configure a shaping rate for a logical interface and oversubscribe the physical interface by including the `shaping-rate` statement at the `[edit class-of-service traffic-control-profiles]` hierarchy level. With this configuration approach, you can independently control the delay-buffer rate.

Results

Confirm the configuration of the class-of-service features (including the 100 Mbp of shaping of the egress traffic) by entering the `show class-of-service` configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show class-of-service
interfaces {
    ge-1/3/1 {
        unit 1 {
            scheduler-map my-map;
            shaping-rate 100m;
        }
    }
}
scheduler-maps {
    my-map {
        forwarding-class best-effort scheduler be;
        forwarding-class expedited-forwarding scheduler ef;
        forwarding-class network-control scheduler nc;
        forwarding-class assured-forwarding scheduler af;
    }
}
schedulers {
    be {
        transmit-rate percent 5;
    }
    ef {
        transmit-rate percent 30;
    }
    af {
        transmit-rate percent 30;
    }
    nc {
        transmit-rate percent 35;
    }
}
```

Configuring Policer Overhead on the PIC or DPC That Hosts the Rate-Shaped Logical Interface

Step-by-Step Procedure

To configure policer overhead on the PIC or MPC that hosts the rate-shaped logical interface:

1. Enable configuration of the supported PIC or MPC.

   ```
   [edit]
   user@host# set chassis fpc 1 pic 3
   ```

2. Configure 100 bytes of policer overhead on the supported PIC or MPC.

   ```
   [edit chassis fpc 1 pic 3]
   user@host# set ingress-policer-overhead 100
   user@host# set egress-policer-overhead 100
   ```

NOTE: These values are added to the length of the final Ethernet frame when determining ingress and egress policer actions for all physical interfaces on the PIC or MPC.

You can specify policer overhead with values from 0 through 255 bytes.

Results

Confirm the configuration of the policer overhead on the physical interface to account for rate-shaping by entering the `show chassis` configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show chassis
chassis {
    fpc 1 {
        pic 3 {
            egress-policer-overhead 100;
            ingress-policer-overhead 100;
        }
    }
}
```

Applying a Policer to the Logical Interface That Carries Input Traffic

Step-by-Step Procedure

To apply a policer to the logical interface that carries input traffic:

1. Configure the logical interface (aggregate) policer.

```
[edit]
user@host# edit firewall policer 500Kbps

[edit firewall policer 500Kbps]
user@host# set logical-interface-policer
user@host# set if-exceeding bandwidth-limit 500k
user@host# set if-exceeding burst-size-limit 625k
user@host# set then discard
```

2. Apply the policer to Layer 3 input on the IPv4 logical interface.

```
[edit]
user@host# set interfaces ge-1/3/1 unit 0 family inet policer input 500Kbps
```

NOTE: The 100 Mbps policer overhead is added to the length of the final Ethernet frame when determining ingress and egress policer actions,

Results

Confirm the configuration of the policer with rate-shaping overhead by entering the `show firewall` and `show interfaces` configuration mode commands. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer 500Kbps {
    logical-interface-policer;
    if-exceeding {
        bandwidth-limit 500k;
        burst-size-limit 625k;
    }
    then discard;
}

[edit]
user@host# show interfaces
ge-1/3/1 {
    per-unit-scheduler;
    vlan-tagging;
    unit 0 {
        vlan-id 100;
        layer2-policer {
            input-policer 500Kbps;
        }
        family inet {
            address 10.10.10.1/30;
        }
    }
```

```
        unit 0 {
            vlan-id 101;
            family inet {
                address 20.20.20.1/30 {
                    arp 20.20.20.2 mac 00:00:11:22:33:44;
                }
            }
        }
    }
```

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

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Displaying Traffic Statistics and Policers for the Logical Interface

Purpose

Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

Action

Use the `show interfaces` operational mode command for logical interface `ge-1/3/1.0`, and include the `detail` or `extensive` option. The command output section for **Traffic statistics** lists the number of bytes and packets received and transmitted on the logical interface, and the **Protocol inet** section contains a **Policer** field that would list the policer `500Kbps` as an input or output policer as follows:

- **Input: 500Kbps-ge-1/3/1.0-log_int-i**
- **Output: 500Kbps-ge-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to Input traffic only.

Displaying Statistics for the Policer

Purpose

Verify the number of packets evaluated by the policer.

Action

Use the `show policer` operational mode command and optionally specify the name of the policer. The command output displays the number of packets evaluated by each configured policer (or the specified policer), in each direction. For the policer `500Kbps`, the input and output policer names are displayed as follows:

- **500Kbps-ge-1/3/1.0-log_int-i**
- **500Kbps-ge-1/3/1.0-log_int-o**

The **log_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to input traffic only.

SEE ALSO

egress-policer-overhead

ingress-policer-overhead

RELATED DOCUMENTATION

Two-Color Policer Configuration Overview

Guidelines for Applying Traffic Policers

Configuring a Policer Overhead

CLI Explorer

CHAPTER 5

Defining Forwarding Behavior with Forwarding Classes

IN THIS CHAPTER

Understanding How Forwarding Classes Assign Classes to Output Queues

IN THIS SECTION

This topic covers the following information:

Output Queue Assignments Based on Forwarding Class

It is helpful to think of forwarding classes as output queues. In effect, the end result of classification is the identification of an output queue for a particular packet.

CoS packet classification assigns an incoming packet to an output queue based on the packet's forwarding class. Each packet is associated with one of the following default forwarding classes:

- Expedited forwarding (EF)—Provides a low-loss, low-latency, low- *jitter*, assured bandwidth, end-to-end service.
- Assured forwarding (AF)—Provides a group of values you can define and includes four subclasses: AF1, AF2, AF3, and AF4, each with three drop probabilities: low, medium, and high.
- Best effort (BE)—Provides no service profile. For the best effort forwarding class, loss priority is typically not carried in a class-of-service (CoS) value and random early detection (RED) drop profiles are more aggressive.
- Network control (NC)—This class is typically high priority because it supports protocol control.

Devices That Support Up to 16 Forwarding Classes

Some Juniper Networks devices support up to 16 forwarding classes, which enables you to classify packets more granularly. For example, you can configure multiple classes of EF traffic: EF, EF1, and EF2. The Junos software supports up to eight output queues; therefore, if you configure more than eight forwarding classes, you must map multiple forwarding classes to single output queues.

Default and Configurable Packet Loss Priority Values

By default, the loss priority is low. On most devices, you can configure high, low, medium-high, or medium-low loss priority.

Configuration Statements Used to Configure and Apply Forwarding Classes

To configure CoS forwarding classes, include the `forwarding-classes` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
forwarding-classes {
    class class-name queue-num queue-number priority (high | low);
    queue queue-number class-name priority (high | low);
}
interfaces {
    interface-name {
        unit logical-unit-number {
            forwarding-class class-name;
        }
    }
}
restricted-queues {
    forwarding-class class-name queue queue-number;
}
```

RELATED DOCUMENTATION

Default Forwarding Classes | **288**

Configuring a Custom Forwarding Class for Each Queue | **291**

Applying Forwarding Classes to Interfaces | **315**

Configuring Up to 16 Custom Forwarding Classes | **293**

Controlling Network Access Using Traffic Policing Overview

Default Forwarding Classes

By default, Junos routers assign four queues to four forwarding classes, each with a queue number, name, and abbreviation.

These default mappings apply to all routers. The four forwarding classes defined by default are shown in Table 21 on page 288.

If desired, you can rename the forwarding classes associated with the queues supported on your hardware. Assigning a new class name to an output queue does not alter the default classification or scheduling that applies to that queue.

BEST PRACTICE: CoS configurations can be quite complicated, so unless your scenario requires it, we recommend that you not alter the default class names or queue number associations.

Some routers support eight queues. Queues 4 through 7 have no default mappings to forwarding classes. To use queues 4 through 7, you must create custom forwarding class names and map these forwarding classes to the queues.

Table 21: Default Forwarding Classes

Queue	Forwarding Class Name	Comments
Queue 0	`best-effort` (be)	The software does not apply any special CoS handling to packets with 000000 in the DiffServ field, a backward compatibility feature. These packets are usually dropped under congested network conditions.
Queue 1	`expedited-forwarding` (ef)	The software delivers assured bandwidth, low loss, low delay, and low delay variation (*jitter*) end-to-end for packets in this service class. Routers accept excess traffic in this class, but in contrast to AF, out-of-profile EF packets can be forwarded out of sequence or dropped.

Table 21: Default Forwarding Classes *(Continued)*

Queue	Forwarding Class Name	Comments
Queue 2	`assured-forwarding` (af)	The software offers a high level of assurance that the packets are delivered as long as the packet flow from the customer stays within a certain service profile that you define. The software accepts excess traffic, but applies a RED drop profile to determine if the excess packets are dropped and not forwarded. Depending on router type, up to four drop probabilities (low, medium-low, medium-high, and high) are defined for this service class.
Queue 3	`network-control` (nc)	The software delivers packets in this service class with a low priority. These packets are not delay sensitive. Typically, these packets represent routing protocol hello or keepalive messages. Because loss of these packets jeopardizes proper network operation, delay is preferable to discard.

The following rules govern queue assignment:

- If classifiers fail to classify a packet, the packet always receives the default classification to the class associated with queue 0.
- The number of queues is dependent on the hardware plugged into the chassis. CoS configurations are inherently contingent on the number of queues on the system. Only two classes, `best-effort` and `network-control`, are referenced in the default configuration. The default configuration works on all routers.
- CoS configurations that specify more queues than the router can support are not accepted. The commit fails with a detailed message that states the total number of queues available.
- All default CoS configuration is based on queue number. The name of the forwarding class that shows up when the default configuration is displayed is the forwarding class currently associated with that queue.

Here is the default configuration for the `forwarding-classes` statement:

```
[edit class-of-service]
forwarding-classes {
    queue 0 best-effort;
    queue 1 expedited-forwarding;
    queue 2 assured-forwarding;
    queue 3 network-control;
}
```

If you reassign the forwarding-class names, the `best-effort` forwarding-class name appears in the locations in the configuration previously occupied by `network-control` as follows:

```
[edit class-of-service]
forwarding-classes {
    queue 0 network-control;
    queue 1 assured-forwarding;
    queue 2 expedited-forwarding;
    queue 3 best-effort;
}
```

All the default rules of classification and scheduling that applied to Queue 3 still apply. Queue 3 is simply now renamed `best-effort`.

You can assign multiple forwarding classes to a single queue. If you do so, the first forwarding class that you assign to queue 0 acquires the default BE classification and scheduling. The first forwarding class that you assign to queue 1 acquires the default EF classification and scheduling. The first forwarding class that you assign to queue 2 acquires the default AF classification and scheduling. The first forwarding class that you assign to queue 3 acquires the default NC classification and scheduling. For more information, see "Configuring Up to 16 Custom Forwarding Classes" on page 293.

CAUTION: When you define a forwarding class for the same queue as one of the default forwarding classes, the default forwarding class is automatically removed. For example, if you define class `be` for queue 0, which is the queue for the default `best-effort` forwarding class, the `best-effort` class is removed.

If you define more than one forwarding class for a given queue number and use the name of a default forwarding class for one of the new classes, the new class with the default name is deleted.

- In the current default configuration:
    - Only IP precedence classifiers are associated with interfaces.
    - The only classes designated are `best-effort` and `network-control`.
    - Schedulers are not defined for the `expedited-forwarding` or `assured-forwarding` forwarding classes.
- You must explicitly classify packets to the `expedited-forwarding` or `assured-forwarding` forwarding class and define schedulers for these classes.

RELATED DOCUMENTATION

Understanding How Forwarding Classes Assign Classes to Output Queues | **286**

Configuring a Custom Forwarding Class for Each Queue | **291**

Change the Default Queuing and Marking of Host Outbound Traffic | **328**

CoS Features and Limitations on PTX Series Routers | **780**

Configuring a Custom Forwarding Class for Each Queue

By default, four queues are assigned to four default forwarding classes, each with a queue number, name, and abbreviation.

BEST PRACTICE: CoS configurations can be quite complicated, so unless it is required by your scenario, we recommend that you not alter the default class names or queue number associations.

If your network requires more than the four default forwarding classes, you can use the following procedure to create custom forwarding class names and assign each forwarding class to any queue number by including the `forwarding-classes` statement at the `[edit class-of-service]` hierarchy level.

The `class` and `queue` statements at the `[edit class-of-service forwarding-classes]` hierarchy level are mutually exclusive. If you want to configure one-to-one mapping of forwarding classes to output queues for up to eight forwarding classes, use the `queue` statement at the `[edit class-of-service forwarding-classes]` hierarchy level. If you want to configure up to 16 forwarding classes with multiple forwarding classes mapped to single output queues (see "Configuring Up to 16 Custom Forwarding Classes" on page 293), include the `class` statement at the `[edit class-of-service forwarding-classes]` hierarchy level.

You cannot commit a configuration that assigns the same forwarding class to two different queues.

CAUTION: We do not recommend classifying packets into a forwarding class that has no associated scheduler on the egress interface. Such a configuration can cause unnecessary packet drops because an unconfigured scheduling class might lack adequate buffer space. For example, if you configure a custom scheduler map that does not define queue 0, and the default classifier assigns incoming packets to the best-effort class (queue 0), the unconfigured egress queue for the best-effort forwarding class might not have enough space to accommodate even short packet bursts.

A default congestion and transmission control mechanism is used when an output interface is not configured for a certain forwarding class, but receives packets destined for that unconfigured forwarding class. This default mechanism uses the delay buffer and weighted round robin (WRR) credit allocated to the designated forwarding class, with a default drop profile. Because the buffer and WRR credit allocation is minimal, packets might be lost if a larger number of packets are forwarded without configuring the forwarding class for the interface.

CAUTION: When you define a forwarding class for the same queue as one of the default forwarding classes, the default forwarding class is automatically removed. For example, if you define class `be` for queue 0, which is the queue for the default `best-effort` forwarding class, the `best-effort` class is removed.

If you define more than one forwarding class for a given queue number and use the name of a default forwarding class for one of the new classes, the new class with the default name is deleted.

To create custom forwarding class names and assign each forwarding class to any queue number:

1. Access the CoS forwarding class configuration hierarchy.

```
[edit]
user@host# edit class-of-service forwarding-classes
```

2. Specify the forwarding class name and queue number.

```
[edit class-of-service forwarding-classes]
user@host# set  class-name queue queue-num
```

RELATED DOCUMENTATION

Understanding How Forwarding Classes Assign Classes to Output Queues | **286**

Default Forwarding Classes | **288**

Change the Default Queuing and Marking of Host Outbound Traffic | **328**

Configuring Up to 16 Custom Forwarding Classes

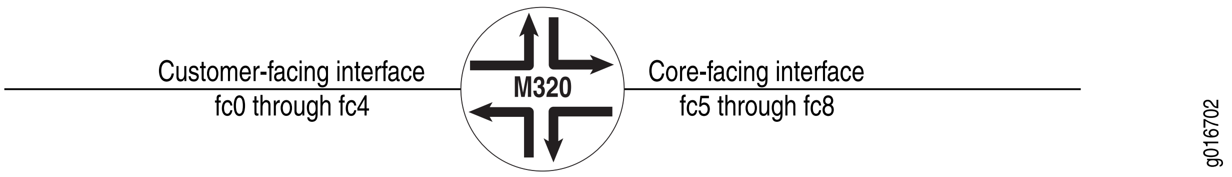
IN THIS SECTION

By default on many routers, four forwarding classes are mapped to four output queues, as shown in the topic "Default Forwarding Classes" on page 288. On these routers, you can configure more than four forwarding classes and queues; you can configure up to 16 forwarding classes and eight queues, with multiple forwarding classes assigned to single queues. The concept of assigning multiple forwarding classes to a queue is sometimes referred to as creating *forwarding-class aliases*.

NOTE: You cannot use CoS-based forwarding features if you configure more than eight forwarding classes on the device.

Mapping multiple forwarding classes to single queues is useful. Suppose, for example, that forwarding classes are set based on multifield packet classification, and the multifield classifiers are different for core-facing interfaces and customer-facing interfaces. Suppose you need four queues for a core-facing interface and five queues for a customer-facing interface, where `fc0` through `fc4` correspond to the classifiers for the customer-facing interface, and `fc5` through `fc8` correspond to classifiers for the core-facing interface, as shown in Figure 31 on page 294.

Figure 31: Customer-Facing and Core-Facing Forwarding Classes



In this example, you need nine classifiers and, therefore, nine forwarding classes. The forwarding class-to-queue mapping is shown in Table 22 on page 295.

Table 22: Sample Forwarding Class-to-Queue Mapping

Forwarding Class Names	Queue Number
`fc0` `fc5`	0
`fc1` `fc6`	1
`fc2` `fc7`	2
`fc3` `fc8`	3
`fc4`	4

To configure up to 16 forwarding classes, include the `class` and `queue-num` statements at the `[edit class-of-service forwarding-classes]` hierarchy level:

```
[edit class-of-service forwarding-classes]
class class-name queue-num queue-number;
```

You can configure up 16 different forwarding-class names. The corresponding output queue number can be from 0 through 7. Therefore, you can map multiple forwarding classes to a single queue. If you map multiple forwarding classes to a queue, the multiple forwarding classes must refer to the same scheduler (at the `[edit class-of-service scheduler-maps` *`map-name`* `forwarding-class` *`class-name`* `scheduler` *`scheduler-name`*`]` hierarchy level).

When you configure up to 16 forwarding classes, you can use them as you can any other forwarding class—in classifiers, schedulers, firewall filters (multifield classifiers), policers, and rewrite rules.

When you configure up to 16 forwarding classes, the following limitations apply:

- The class and queue statements at the [edit class-of-service forwarding-classes] hierarchy level are mutually exclusive. In other words, you can include one or the other of the following configurations, but not both:

```
[edit class-of-service forwarding-classes]
queue queue-number class-name;

[edit class-of-service forwarding-classes]
class class-name queue-num queue-number;
```

- When you use CoS-based forwarding features, you cannot configure more than eight forwarding classes with a forwarding policy. However, if you try to configure CoS-based forwarding with more than eight forwarding classes configured, commit fails with a message. Therefore, you can configure CBF on a router with eight or less than eight forwarding classes only. Under this condition, the forwarding class to queue mapping can be either one-to-one or one-to-many.

- A scheduler map that maps eight different forwarding classes to eight different schedulers can only be applied to interfaces that support eight queues. If you apply this type of scheduler map to an interface that only supports four queues, then the commit fails.

- We recommend that you configure the statements changing PICs to support eight queues and then applying an eight queue scheduler map in two separate steps. Otherwise, the commit might succeed but the PIC might not have eight queues when the scheduler map is applied, generating an error.

You can determine the ID number assigned to a forwarding class by issuing the show class-of-service forwarding-class command. You can determine whether the classification is fixed by issuing the show class-of-service forwarding-table classifier mapping command. In the command output, if the Table Type field appears as Fixed, the classification is fixed. For more information about fixed classification, see "Applying Forwarding Classes to Interfaces" on page 315.

Enabling Eight Queues on Interfaces

By default, some routers are restricted to a maximum of four egress queues per interface. The following procedures describe how to configure a maximum of eight egress queues on these interfaces.

NOTE: In addition to configuring eight queues at the [edit chassis] hierarchy level, the configuration at the [edit class-of-service] hierarchy level must support eight queues per interface.

The maximum number of queues per PIC can be 4 or 8. If you include the max-queues-per-interface statement, all ports on the PIC use configured mode and all interfaces on the IQ PIC have the same maximum number of queues.

To configure a maximum of eight egress queues on these PICs:

1. Specify the PIC you want to configure.

   ```
   [edit]
   user@host# edit chassis fpc slot-number pic pic-number
   ```

2. Configure a maximum of eight egress queues on these interfaces.

   ```
   [edit chassis fpc slot-number pic pic-number]
   user@host# set max-queues-per-interface 8
   ```

   The numerical value can be `4` or `8`.

To determine how many queues an interface supports, you can check the `CoS queues` output field of the `show interfaces interface-name extensive` command:

1. To view how many queues an interface supports:

   ```
   user@host> show interfaces so-1/0/0 extensive
   CoS queues: 8 supported
   ```

If you include the `max-queues-per-interface 4` statement, you can configure all four ports and configure up to four queues per port.

When you include the `max-queues-per-interface` statement and commit the configuration, all physical interfaces on the PIC are deleted and re-added. Also, the PIC is taken offline and then brought back online immediately. You do not need to take the PIC offline and online manually. You should change modes between four queues and eight queues only when there is no active traffic going to the PIC.

Assigning Multiple Forwarding Classes and Default Forwarding Classes

For queues 0 through 3, if you assign multiple forwarding classes to a single queue, default forwarding class assignment works as follows:

- The first forwarding class that you assign to queue 0 acquires the default BE classification and scheduling.
- The first forwarding class that you assign to queue 1 acquires the default EF classification and scheduling.
- The first forwarding class that you assign to queue 2 acquires the default AF classification and scheduling.

- The first forwarding class that you assign to queue 3 acquires the default NC classification and scheduling.

Of course you can override the default classification and scheduling by configuring custom classifiers and schedulers.

If you do not explicitly map forwarding classes to queues 0 through 3, then the respective default classes are automatically assigned to those queues. When you are counting the 16 forwarding classes, you must include in the total any default forwarding classes automatically assigned to queues 0 through 3. As a result, you can map up to 13 forwarding classes to a single queue when the single queue is queue 0, 1, 2, or 3. You can map up to 12 forwarding classes to a single queue when the single queue is queue 4, 5, 6, or 7. In summary, there must be at least one forwarding class each (default or otherwise) assigned to queue 0 through 3, and you can assign the remaining 12 forwarding classes (16–4) to any queue.

For example, suppose you assign two forwarding classes to queue 0 and you assign no forwarding classes to queues 1 through 3. The software automatically assigns one default forwarding class each to queues 1 through 3. This means 11 forwarding classes (16–5) are available for you to assign to queues 4 through 7.

For more information about forwarding class defaults, see "Default Forwarding Classes" on page 288.

Examples: Configuring Up to 16 Forwarding Classes

To configure 16 forwarding classes, map two forwarding classes to each queue. For example:

Specify each forwarding class and queue you want mapped.

```
[edit]
user@host# edit class-of-service forwarding-classes
user@host# set class fc0 queue-num 0
user@host# set class fc1 queue-num 0
user@host# set class fc2 queue-num 1
user@host# set class fc3 queue-num 1
user@host# set class fc4 queue-num 2
user@host# set class fc5 queue-num 2
user@host# set class fc6 queue-num 3
user@host# set class fc7 queue-num 3
user@host# set class fc8 queue-num 4
user@host# set class fc9 queue-num 4
user@host# set class fc10 queue-num 5
user@host# set class fc11 queue-num 5
user@host# set class fc12 queue-num 6
user@host# set class fc13 queue-num 6
```

```
user@host# set class fc14 queue-num 7
user@host# set class fc15 queue-num 7
```

For PICs restricted to four queues, map four forwarding classes to each queue:

1. Specify each forwarding class and queue you want mapped.

   ```
   [edit]
   user@host# edit class-of-service restricted-queues
   user@host# set forwarding-class fc0 queue 0
   user@host# set forwarding-class fc1 queue 0
   user@host# set forwarding-class fc2 queue 0
   user@host# set forwarding-class fc3 queue 0
   user@host# set forwarding-class fc4 queue 1
   user@host# set forwarding-class fc5 queue 1
   user@host# set forwarding-class fc6 queue 1
   user@host# set forwarding-class fc7 queue 1
   user@host# set forwarding-class fc8 queue 2
   user@host# set forwarding-class fc9 queue 2
   user@host# set forwarding-class fc10 queue 2
   user@host# set forwarding-class fc11 queue 2
   user@host# set forwarding-class fc12 queue 3
   user@host# set forwarding-class fc13 queue 3
   user@host# set forwarding-class fc14 queue 3
   user@host# set forwarding-class fc15 queue 3
   ```

If you map multiple forwarding classes to a queue, the multiple forwarding classes must refer to the same scheduler. To configure a scheduler map applicable to an interface restricted to four queues:

1. Specify a scheduler map name and associate it with the scheduler configuration and forwarding class.

   ```
   [edit]
   user@host# edit class-of-service scheduler-maps interface-restricted
   user@host# set forwarding-class be scheduler 00
   user@host# set forwarding-class ef scheduler Q1
   user@host# set forwarding-class ef1 scheduler Q1
   user@host# set forwarding-class ef2 scheduler Q1
   user@host# set forwarding-class af1 scheduler Q2
   user@host# set forwarding-class af scheduler Q2
   user@host# set forwarding-class nc scheduler Q3
   user@host# set forwarding-class nc1 scheduler Q3
   ```

2. Map the forwarding classes to the restricted queues.

```
[edit]
user@host# edit class-of-service restricted-queues
user@host# set forwarding-class be queue 0
user@host# set forwarding-class ef queue 1
user@host# set forwarding-class ef1 queue 1
user@host# set forwarding-class ef2 queue 1
user@host# set forwarding-class af queue 2
user@host# set forwarding-class af1 queue 2
user@host# set forwarding-class nc queue 3
user@host# set forwarding-class nc1 queue 3
```

RELATED DOCUMENTATION

Understanding How Forwarding Classes Assign Classes to Output Queues | **286**

Default Forwarding Classes | **288**

Classify Packets by Egress Interface

IN THIS SECTION

On supported platforms, you can classify unicast and multicast packets based on the egress interface. For unicast traffic, you can also use a multifield filter, but only egress interface classification applies to multicast traffic as well as unicast traffic. If you configure egress classification of an interface, you cannot perform DSCP rewrites on the interface. By default, the system does not perform any classification based on the egress interface.

To enable packet classification by the egress interface, you first configure a forwarding class map and one or more queue numbers for the egress interface at the `[edit class-of-service forwarding-class-map` *`forwarding-class-map-name`*`]` hierarchy level:

```
[edit class-of-service]
forwarding-class-map  forwarding-class-map-name {
    class class-name queue-num queue-number [ restricted-queue queue-number ];
}
```

NOTE: If you configure an output forwarding class map associating a forwarding class with a queue number, this map is not supported on multiservices link services IQ (`lsq-`) interfaces.

Once the forwarding class map has been configured, you apply the map to the logical interface by using the `output-forwarding-class-map` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
output-forwarding-class-map forwarding-class-map-name;
```

All parameters relating to the queues and forwarding class must be configured as well. For more information about configuring forwarding classes and queues, see "Configuring a Custom Forwarding Class for Each Queue" on page 291.

NOTE: You cannot apply a rewrite rule and output forwarding class map to the same logical interface (unit). Although a warning is issued, the CLI does not prevent this configuration. An error message appears when you attempt to commit the configuration.

This example shows how to configure an interface-specific forwarding-class map named `FCMAP1` that restricts queues 5 and 6 to different queues on four-queue systems and then applies `FCMAP1` to `unit 0` of interface `ge-6/0/0`:

```
[edit class-of-service]
forwarding-class-map FCMAP1 {
    class FC1 queue-num 6 restricted-queue 3;
    class FC2 queue-num 5 restricted-queue 2;
    class FC3 queue-num 3;
    class FC4 queue-num 0;
    class FC3 queue-num 0;
    class FC4 queue-num 1;
}

[edit class-of-service]
interfaces {
    ge-6/0/0 unit 0 {
        output-forwarding-class-map FCMAP1;
    }
}
```

Note that without the `restricted-queue` option in `FCMAP1`, the example would assign `FC1` and `FC2` to queues 2 and 1, respectively, on a system restricted to four queues.

Use the `show class-of-service forwarding-class` *`forwarding-class-map-name`* command to display the forwarding-class map queue configuration:

```
user@host> show class-of-service forwarding-class FCMAP2

Forwarding class                       ID      Queue  Restricted queue
  FC1                                   0       6          3
  FC2                                   1       5          2
  FC3                                   2       3          3
  FC4                                   3       0          0
  FC5                                   4       0          0
  FC6                                   5       1          1
  FC7                                   6       6          2
  FC8                                   7       7          3
```

Use the `show class-of-service interface` *`interface-name`* command to display the forwarding-class maps (and other information) assigned to a logical interface:

```
user@host> show class-of-service interface ge-6/0/0

Physical interface: ge-6/0/0, Index: 128
Queues supported: 8, Queues in use: 8
 Scheduler map: <default>, Index: 2
 Input scheduler map: <default>, Index: 3
 Chassis scheduler map: <default-chassis>, Index: 4

Logical interface: ge-6/0/0.0, Index: 67
 Object                       Name             Type                      Index
```

```
  Scheduler-map               sch-map1         Output                  6998
  Scheduler-map               sch-map1         Input                   6998
  Classifier                  dot1p            ieee8021p               4906
  forwarding-class-map        FCMAP1           Output                  1221

 Logical interface: ge-6/0/0.1, Index 68
  Object                      Name             Type                   Index
  Scheduler-map               <default>        Output                     2
  Scheduler-map               <default>        Input                      3

 Logical interface: ge-6/0/0.32767, Index 69
  Object                      Name             Type                   Index
  Scheduler-map               <default>        Output                     2
  Scheduler-map               <default>        Input                      3
```

Platform-Specific Forwarding Class Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behaviors for your platform:

Platform	Difference
MX Series	On an MX Series router that contains MPCs and MS-DPCs, multicast packets are dropped on the router and not processed properly if the router contains MLPPP LSQ logical interfaces that function as multicast receivers and if the network services mode is configured as enhanced IP mode on the router. This behavior is expected with LSQ interfaces with enhanced IP mode. In such a scenario, if enhanced IP mode is not configured, multicasting works correctly. However, if the router contains redundant LSQ interfaces and enhanced IP network services mode configured with FIB localization, multicast works properly.

Forwarding Policy Options Overview

IN THIS SECTION

Class-of-service (CoS)-based forwarding (CBF) enables you to control next-hop selection based on a packet's *class of service* and, in particular, the value of the IP packet's precedence bits.

For example, you might want to specify a particular interface or next hop to carry high-priority traffic while all best-effort traffic takes some other path. When a routing protocol discovers equal-cost paths, Junos picks a path at random or load-balance across the paths through either hash selection or round robin. CBF allows path selection based on class.

To configure CBF properties, include the following statements at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
forwarding-policy {
    next-hop-map map-name {
        forwarding-class class-name {
            next-hop [ next-hop-name ];
            lsp-next-hop [ lsp-regular-expression ];
            non-lsp-next-hop;
            discard;
        }
        forwarding-class-default {
            discard;
            lsp-next-hop [ lsp-regular-expression ];
            next-hop [next-hop-name];
            non-lsp-next-hop;
        }
    }
    class class-name {
        classification-override {
            forwarding-class class-name;
        }
    }
}
```

Platform-Specific CBF Behavior

Use the following table to review platform-specific behaviors for your platforms.

Table 23: Platform-Specific CBF Behavior

Platform	Difference
MX Series	• MX routers support configuring CBF for up to 16 forwarding classes. • To support up to 16 forwarding classes for CBF on MX routers, enable `enhanced-ip` at the `[edit chassis network-services]` hierarchy level.
PTX Series	• PTX routers support configuring CBF for up to 16 forwarding classes. • Enabling `enhanced-ip` is not necessary on PTX routers to support 16 forwarding classes for CBF.

RELATED DOCUMENTATION

Configuring CoS-Based Forwarding | 305

Example: Configuring CoS-Based Forwarding | 309

Configuring CoS-Based Forwarding

IN THIS SECTION

You can apply CoS-based forwarding (CBF) only to a defined set of routes. Therefore, you must configure a policy statement as in the following example:

```
[edit policy-options]
policy-statement my-cos-forwarding {
    from {
        route-filter destination-prefix match-type;
    }
    then {
        cos-next-hop-map map-name;
    }
}
```

This configuration specifies that routes matching the route filter are subject to the CoS next-hop mapping specified by *map-name*. For more information about configuring policy statements, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

NOTE: You can configure CBF on a device with the supported number or fewer forwarding classes plus a default forwarding class only. Under this condition, the forwarding class to queue mapping can be either one-to-one or one-to-many. However, you cannot configure CBF when the number of forwarding classes configured exceeds the supported number. Similarly, with CBF configured, you cannot configure more than the supported number of forwarding classes plus a default forwarding class.

To specify a CoS next-hop map, include the `forwarding-policy` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
forwarding-policy {
    next-hop-map map-name {
        forwarding-class class-name {
            discard;
            lsp-next-hop [ lsp-regular-expression ];
            next-hop [ next-hop-name ];
            non-lsp-next-hop;
        }
        forwarding-class-default {
            discard;
            lsp-next-hop [ lsp-regular-expression ];
            next-hop [next-hop-name];
            non-lsp-next-hop;
        }
    }
}
```

When you configure CBF with OSPF as the interior gateway protocol (IGP), you must specify the next hop as an interface name or next-hop alias, not as an IPv4 or IPv6 address. This is true because OSPF adds routes with the interface as the next hop for point-to-point interfaces; the next hop does not contain the IP address. For an example configuration, see "Example: Configuring CoS-Based Forwarding" on page 309.

For Layer 3 VPNs, when you use class-based forwarding for the routes received from the far-end provider edge (PE) router within a VRF instance, the software can match the routes based on the attributes that come with the received route only. In other words, the matching can be based on the route within RIB-in. In this case, the `route-filter` statement you include at the `[edit policy-options policy-statement my-cos-forwarding from]` hierarchy level has no effect because the policy checks the `bgp.l3vpn.0` table, not the *vrf*`.inet.0` table.

Junos applies the CoS next-hop map to the set of next hops previously defined; the next hops themselves can be located across any outgoing interfaces on the routing device. For example, the following configuration associates a set of forwarding classes and next-hop identifiers:

```
[edit class-of-service forwarding-policy]
next-hop-map map1 {
    forwarding-class expedited-forwarding {
        next-hop next-hop1;
        next-hop next-hop2;
    }
    forwarding-class best-effort {
        next-hop next-hop3;
        lsp-next-hop lsp-next-hop4;
    }
    forwarding-class-default {
        lsp-next-hop lsp-next-hop5;
    }
}
```

In this example, `next-hop` *N* is either an IP address or an egress interface for some next hop, and `lsp-next-hop` *N* is a regular expression corresponding to any next hop with that label. Q1 through Q*N* are a set of forwarding classes that map to the specific next hop. That is, when a packet is switched with Q1 through Q*N*, it is forwarded out the interface associated with the associated next hop.

This configuration has the following implications:

- A single forwarding class can map to multiple standard next hops or LSP next hops. This implies that load sharing is done across standard next hops or LSP next hops servicing the same class value. To make this work properly, Junos OS creates a list of the equal-cost next hops and forwards packets according to standard load-sharing rules for that forwarding class.

- If a forwarding class configuration includes LSP next hops and standard next hops, the LSP next hops are preferred over the standard next hops. In the preceding example, if both `next-hop3` and `lsp-next-hop4` are valid next hops for a route to which `map1` is applied, the forwarding table includes entry `lsp-next-hop4` only.

- If `next-hop-map` does not specify all possible forwarding classes, the default forwarding class is selected as the default. *default-forwarding class* defines the next hop for traffic that does not meet any forwarding class in the next hop map. If the default forwarding class is not specified in the next-hop map, a default is designated randomly. The default forwarding class is the class associated with queue 0.

- For LSP next hops, Junos uses UNIX `regex(3)`-style regular expressions. For example, if the following labels exist: `lsp`, `lsp1`, `lsp2`, `lsp3`, the statement `lsp-next-hop lsp` matches `lsp`, `lsp1`, `lsp2`, and `lsp3`. If you do not want this behavior, you must use the anchor characters `lsp-next-hop " ^lsp$"`, which match `lsp` only.

- The route filter does not work because the policy checks against the `bgp.l3vpn.0` table instead of the *vrf*`.inet.0` table.

The final step is to apply the route filter to routes exported to the forwarding engine. This is shown in the following example:

```
routing-options {
    forwarding-table {
        export my-cos-forwarding;
    }
}
```

This configuration instructs the routing process to insert routes to the forwarding engine matching `my-cos-forwarding` with the associated next-hop CBF rules.

The following algorithm is used when you apply a configuration to a route:

- If the route is a single next-hop route, all traffic goes to that route; that is, no CBF takes effect.

- For each next hop, associate the proper forwarding class. If a next hop appears in the route but not in the `cos-next-hop` map, it does not appear in the forwarding table entry.

- The default forwarding class is used if not all forwarding classes are specified in the next-hop map. If the default is not specified, the default is assigned to the lowest class defined in the next-hop map.

Platform-Specific CBF Behavior

Use the following table to review platform-specific behaviors for your platforms.

Table 24: Platform-Specific CBF Behavior

Platform	Difference
MX Series	• MX routers support configuring CBF for up to 16 forwarding classes. • To support up to 16 forwarding classes for CBF on MX routers, enable `enhanced-ip` at the `[edit chassis network-services]` hierarchy level.
PTX Series	• PTX routers support configuring CBF for up to 16 forwarding classes. • Enabling `enhanced-ip` is not necessary on PTX routers to support 16 forwarding classes for CBF.

RELATED DOCUMENTATION

Load Balancing VPLS Non-Unicast Traffic Across Member Links of an Aggregate Interface

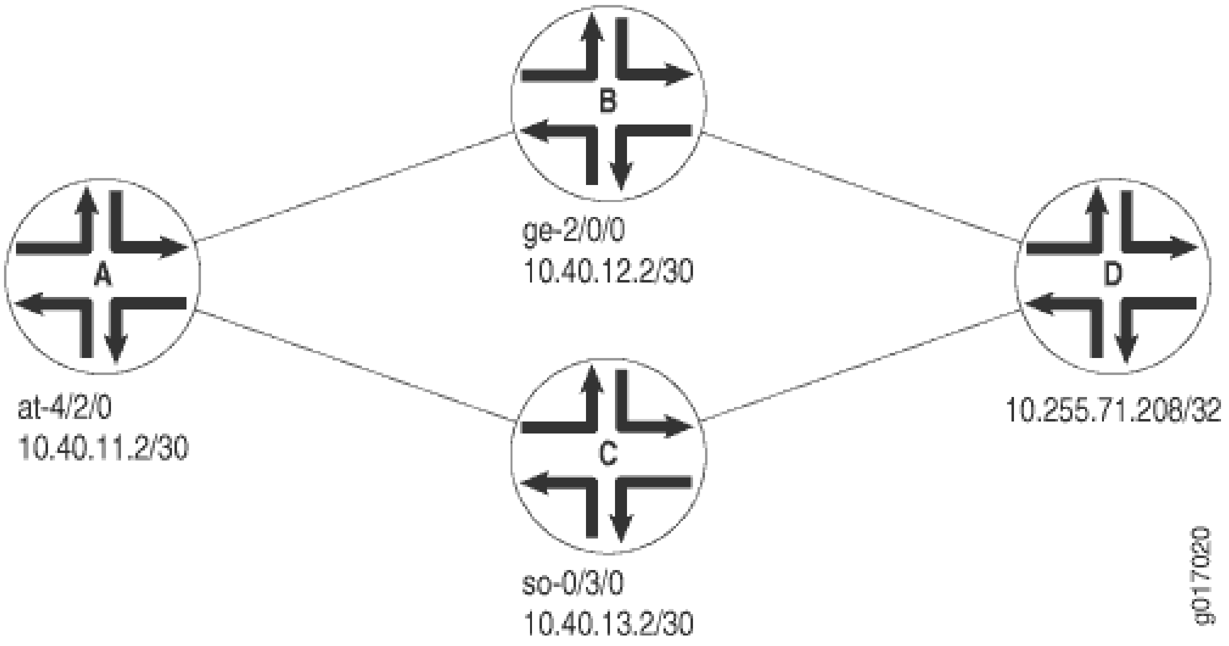
Forwarding Policy Options Overview | 304

Example: Configuring CoS-Based Forwarding

Router A has two routes to destination `10.255.71.208` on Router D. One route goes through Router B, and the other goes through Router C, as shown in Figure 32 on page 310.

Configure Router A with CoS-based forwarding (CBF) to select Router B for queue 0 and queue 2, and Router C for queue 1 and queue 3.

Figure 32: Sample CoS-Based Forwarding



When you configure CBF with OSPF as the IGP, you must specify the next hop as an interface name, not as an IPv4 or IPv6 address. The next hops in this example are specified as `ge-2/0/0.0` and `so-0/3/0.0`.

```
[edit class-of-service]
forwarding-policy {
    next-hop-map my_cbf {
        forwarding-class be {
            next-hop ge-2/0/0.0;
        }
        forwarding-class ef {
            next-hop so-0/3/0.0;
        }
        forwarding-class af {
            next-hop ge-2/0/0.0;
        }
        forwarding-class nc {
            next-hop so-0/3/0.0;
        }
    }
}
classifiers {
    inet-precedence inet {
        forwarding-class be {
            loss-priority low code-points [ 000 100 ];
        }
        forwarding-class ef {
            loss-priority low code-points [ 001 101 ];
        }
        forwarding-class af {
            loss-priority low code-points [ 010 110 ];
        }
        forwarding-class nc {
            loss-priority low code-points [ 011 111 ];
        }
    }
}
forwarding-classes {
    queue 0 be;
    queue 1 ef;
    queue 2 af;
    queue 3 nc;
}
```

```
interfaces {
    at-4/2/0 {
        unit 0 {
            classifiers {
                inet-precedence inet;
            }
        }
    }
}

[edit policy-options]
policy-statement cbf {
    from {
        route-filter 10.255.71.208/32 exact;
    }
    then cos-next-hop-map my_cbf;
}

[edit routing-options]
graceful-restart;
forwarding-table {
    export cbf;
}

[edit interfaces]
traceoptions {
    file trace-intf size 5m world-readable;
    flag all;
}
so-0/3/0 {
    unit 0 {
        family inet {
            address 10.40.13.1/30;
        }
        family iso;
        family mpls;
    }
}
ge-2/0/0 {
    unit 0 {
        family inet {
            address 10.40.12.1/30;
        }
        family iso;
        family mpls;
    }
}
at-4/2/0 {
    atm-options {
        vpi 1 {
            maximum-vcs 1200;
        }
    }
    unit 0 {
        vci 1.100;
        family inet {
            address 10.40.11.2/30;
        }
        family iso;
        family mpls;
    }
}
```

RELATED DOCUMENTATION

Forwarding Policy Options Overview | 304

Example: Configuring CoS-Based Forwarding for Different Traffic Types

One common use for CoS-based forwarding and next-hop maps is to enforce different handling for different traffic types, such as voice and video. For example, an LSP-based next hop can be used for voice and video, and a non-LSP next-hop can be used for best effort traffic.

Only the forwarding policy is shown in this example:

```
[edit class-of-service]
forwarding-policy {
    next-hop-map ldp-map {
        forwarding-class expedited-forwarding {
            lsp-next-hop voice;
            non-lsp-next-hop;
        }
        forwarding-class assured-forwarding {
            lsp-next-hop video;
            non-lsp-next-hop;
        }
        forwarding-class best-effort {
            non-lsp-next-hop;
            discard;
        }
    }
}
```

Example: Configuring CoS-Based Forwarding for IPv6

This example configures CoS-based forwarding (CBF) next-hop maps and CBF LSP next-hop maps for IPv6 addresses.

You can configure a next-hop map with both IPv4 and IPv6 addresses, or you can configure separate next-hop maps for IPv4 and IPv6 addresses and include the `from family (inet | inet6)` statements at the `[edit policy-options policy-options policy-statement` *`policy-name`* `term` *`term-name`*`]` hierarchy level to ensure that only next-hop maps of a specified protocol are applied to a specified route.

If you do not configure separate next-hop maps and include the `from family (inet | inet6)` statements in the configuration, when a route uses two next hops (whether IPv4, IPv6, interface, or LSP next hop) in at least two of the specified forwarding classes, CBF is used for the route; otherwise, the CBF policy is ignored.

1. Define the CBF next-hop map:

```
[edit class-of-service]
forwarding-policy {
    next-hop-map cbf-map {
        forwarding-class best-effort {
            next-hop [ ::192.168.139.38 192.168.139.38 ];
        }
        forwarding-class expedited-forwarding {
            next-hop [ ::192.168.140.5 192.168.140.5 ];
        }
        forwarding-class assured-forwarding {
            next-hop [ ::192.168.145.5 192.168.145.5 ];
        }
        forwarding-class network-control {
            next-hop [ ::192.168.141.2 192.168.141.2 ];
        }
    }
}
```

2. Define the CBF forwarding policy:

```
[edit policy-options]
policy-statement ls {
    then cos-next-hop-map cbf-map;
}
```

3. Export the CBF forwarding policy:

```
[edit routing-options]
forwarding-table {
    export ls;
}
```

Applying Forwarding Classes to Interfaces

You can configure *fixed classification* on a logical interface by specifying a forwarding class to be applied to all packets received by the logical interface, regardless of the packet contents.

To apply a forwarding class configuration to the input logical interface, include the `forwarding-class` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
forwarding-class class-name;
```

You can include interface wildcards for *`interface-name`* and *`logical-unit-number`*.

In the following example, all packets coming into the router from the `ge-3/0/0.0` interface are assigned to the `assured-forwarding` forwarding class:

```
[edit class-of-service]
interfaces {
    ge-3/0/0 {
        unit 0 {
            forwarding-class assured-forwarding;
        }
    }
}
```

RELATED DOCUMENTATION

forwarding-class

Understanding Queuing and Marking of Host Outbound Traffic

IN THIS SECTION

This topic covers the following information:

Host Outbound Traffic Overview

Host outbound traffic, also called locally generated traffic, consists of traffic generated by the Routing Engine and traffic generated by the distributed protocol handler.

Routing Engine Sourced Traffic

Traffic sent from the Routing Engine includes control plane packets such as OSPF Hello packets, ICMP echo reply (ping) packets, and TCP-related packets such as BGP and LDP control packets.

Distributed Protocol Handler Traffic

Distributed protocol handler traffic refers to traffic from the router's *periodic packet management* (PPM) process when it runs sessions distributed to the Packet Forwarding Engine (the default mode) in addition to the Routing Engine. The PPM process is responsible for periodic transmission of protocol Hello or other keepalive packets on behalf of its various client processes, such as Bidirectional Forwarding Detection (BFD) Protocol or Link Aggregation Control Protocol (LACP), and it also receives packets on behalf of client processes. In addition, PPM handles time-sensitive periodic processing and performs such tasks as sending process-specific packets and gathering statistics. By default, PPM sessions on the Routing Engine run distributed on the Packet Forwarding Engine, and this enables client processes to run on the Packet Forwarding Engine.

Distributed protocol handler traffic includes both IP (Layer 3) traffic such as BFD keepalivemessages and non-IP (Layer 2) traffic such as LACP control traffic on aggregated Ethernet.

Default Queuing and Marking of Host Outbound Traffic

By default, the router assigns host outbound traffic to the `best-effort` forwarding class (which maps to queue 0) or to the `network-control` forwarding class (which maps to queue 3) based on protocol. For more information, see "Default Routing Engine Protocol Queue Assignments" on page 319.

For most protocols, the router marks the type of service (ToS) field of Layer 3 packets in the host outbound traffic flow with DiffServ code point (DSCP) bits 000000 (which correlate with the `best-effort` forwarding class). For some protocols, such as BGP, the router marks the ToS field with 802.1p bits 110 (which correlate with the `network-control` forwarding class), while still assigning the packets to queue 0. The router does not remark Layer 2 traffic such as LACP control traffic on aggregated Ethernet. For more information, see "Default DSCP and DSCP IPv6 Classifiers" on page 65.

Configured Queuing and Marking of Host Outbound Traffic

You can configure a nondefault forwarding class and DSCP bits that the router uses to queue and remark host outbound traffic. These configuration settings apply to the following types of traffic:

- Packets generated by the Routing Engine
- Distributed protocol handler traffic for egress interfaces

To change these default settings, include the `forwarding-class` *`class-name`* statement and the `dscp-code-point` *`value`* statement at the `[edit class-of-service host-outbound-traffic]` hierarchy level. This feature does not affect transit traffic or incoming traffic.

The configured forwarding class override applies to all packets relating to Layer 2 protocols, Layer 3 protocols, and all application-level traffic (such as FTP or ping operations). The configured DSCP bits override value does not apply to MPLS EXP bits or IEEE 802.1p bits, however.

Configured Queuing and Marking of Outbound Routing Engine Traffic Only

To configure a nondefault forwarding class and DSCP bits that the router uses to queue and remark traffic generated by the Routing Engine only, attach an IPv4 firewall filter to the output of the router's loopback address. Use the `forwarding-class` and `dscp` filter actions to specify override values.

This feature overrides the `host-outbound-traffic` settings for the Routing Engine output traffic only.

Platform-Specific Behavior

Use the following table to review platform-specific behaviors for your platforms.

Table 25: Platform-Specific Behavior

Platform	Difference
MX80 Routers	• For interfaces on MX80 routers, LACP control traffic is sent through the Routing Engine rather than through the Packet Forwarding Engine.

RELATED DOCUMENTATION

Default Routing Engine Protocol Queue Assignments | 319

Default DSCP and DSCP IPv6 Classifiers | 65

Understanding Queuing and Marking of Host Outbound Traffic | 316

Forwarding Classes and Fabric Priority Queues

IN THIS SECTION

This topic covers the following information:

Default Fabric Priority Queuing

On Juniper devices, the default behavior is for fabric priority queuing on egress interfaces to match the scheduling priority you assign. High-priority egress traffic is automatically assigned to high-priority fabric queues. Likewise, low-priority egress traffic is automatically assigned to low-priority fabric queues.

Overriding Default Fabric Priority Queuing

You can override the default fabric priority queuing of egress traffic by including the `priority` statement at the `[edit class-of-service forwarding-classes queue *queue-number class-name*]` hierarchy level:

```
[edit class-of-service forwarding-classes queue queue-number class-name]
priority (high | low);
```

RELATED DOCUMENTATION

Associating Schedulers with Fabric Priorities | 432

Default Routing Engine Protocol Queue Assignments

Table 26 on page 320 lists the default output queues to which Routing Engine sourced traffic is mapped by protocol type. In general, control protocol packets are sent over queue 3 and management traffic is sent over queue 0. The following caveats apply to these default queue assignments:

- For all packets sent to queue 3 over a VLAN-tagged interface, the software sets the 802.1p bit to 110, except for VRRP packets, in which case the software sets the 802.1p bit to 111.
- Outgoing BFD packets should be marked with VLAN-tagged 802.1p bit to 110; however, this is true only for RE based BFD. For inline BFD, it does not modify by default.
- Although BGP and LDP TCP traffic is sent to queue 0, it is marked with 802.1p bits 110 normally used for network control.
- For IPv4 and IPv6 packets, the software copies the IP type-of-service (ToS) value into the 802.1p field independently of which queue the packets are sent out.
- For MPLS packets, the software copies the EXP bits into the 802.1p field.

Table 26: Default Queue Assignments for Packets Generated by the Routing Engine

Routing Engine Protocol	Default Queue Assignment
Adaptive Services PIC TCP tickle (keepalive packets for idle session generated with stateful firewall to probe idle TCP sessions)	Queue 0
Address Resolution Protocol (ARP)	Queue 0
Bidirectional Forwarding Detection (BFD) Protocol	Queue 3
BGP	Queue 0
BGP TCP Retransmission	Queue 3
Cisco High-Level Data Link Control (HDLC)	Queue 3
Distance Vector Multicast Routing Protocol (DVMRP)	Queue 3
Ethernet Operation, Administration, and Maintenanc e (OAM)	Queue 3
FTP	Queue 0

Table 26: Default Queue Assignments for Packets Generated by the Routing Engine *(Continued)*

Routing Engine Protocol	Default Queue Assignment
IS-IS Open Systems Interconnection (OSI)	Queue 3
Internet Control Message Protocol (ICMP)	Queue 0
Internet Group Management Protocol (IGMP) query	Queue 3
IGMP Report	Queue 0
Internet Key Exchange (IKE)	Queue 3
IP version 6 (IPv6) Neighbor Solicitation	Queue 3
IPv6 Neighbor Advertisement	Queue 3
IPv6 Router Advertisement	Queue 0
Label Distribution Protocol (LDP) User Datagram Protocol (UDP) hello	Queue 3
LDP keepalive and Session data	Queue 0
LDP TCP Retransmission	Queue 3
Link Aggregation Control Protocol (LACP)	Queue 3
Link Services (LS) PIC	If link fragmentation and interleaving (LFI) is enabled, all routing protocol packets larger than 128 bytes are transmitted from queue 0. This ensures that VoIP traffic is not affected. Fragmentation is supported on queue 0 only.
Multicast listener discovery (MLD)	Queue 0

Table 26: Default Queue Assignments for Packets Generated by the Routing Engine ***(Continued)***

Routing Engine Protocol	Default Queue Assignment
Multicast Source Discovery Protocol (MSDP)	Queue 0
MSDP TCP Retransmission	Queue 3
NETCONF	Queue 0
NetFlow	Queue 0
OSPF protocol data unit (PDU)	Queue 3
Point-to-Point Protocol (PPP)	Queue 3
Protocol Independent Multicast (PIM)	Queue 3
Precision Time Protocol (PTP)	Queue 3
Real-time performance monitoring (RPM) probe packets	Queue 3
RSVP	Queue 3
Routing Information Protocol (RIP)	Queue 3
SNMP	Queue 0
SSH	Queue 0
sFlow monitoring technology	Queue 0
Telnet	Queue 0

Table 26: Default Queue Assignments for Packets Generated by the Routing Engine *(Continued)*

Routing Engine Protocol	Default Queue Assignment
Two-Way Active Monitoring Protocol (TWAMP)	Queue 0
Virtual Router Redundancy Protocol (VRRP)	Queue 3
`xnm-clear-text`	Queue 0
`xnm-ssl`	Queue 0

Assigning Forwarding Class and DSCP Value for Routing Engine-Generated Traffic

You can set the forwarding class and differentiated service code point (DSCP) value for traffic originating in the Routing Engine. To configure forwarding class and DSCP values that apply to Routing Engine–generated traffic only, apply an output filter to the loopback (`lo.0`) interface and set the appropriate forwarding class and DSCP bit configuration for various protocols. For example, you can set the DSCP value on OSPF packets that originate in the Routing Engine to `10` and assign them to the AF (assured forwarding) forwarding class while the DSCP value on ping packets are set to `0` and use forwarding class BE (best effort).

This particular classification ability applies to packets generated by the Routing Engine only.

The following example assigns Routing Engine sourced ping packets (using ICMP) a DSCP value of `38` and a forwarding class of `af17`, OSPF packets a DSCP value of `12` and a forwarding class of `af11`, and BGP packets (using TCP) a DSCP value of `10` and a forwarding class of `af16`.

```
[edit class-of-service]
forwarding-classes {
    class af11 queue-num 7;
    class af12 queue-num 1;
    class af13 queue-num 2;
    class af14 queue-num 4;
    class af15 queue-num 5;
    class af16 queue-num 4;
    class af17 queue-num 6;
    class af18 queue-num 7;
}

[edit firewall filter family inet]
filter loopback-filter {
    term t1 {
        from {
            protocol icmp; # For pings
        }
        then {
            forwarding-class af17;
            dscp 38;
        }
    }
    term t2 {
        from {
            protocol ospf; # For OSPF
        }
        then {
            forwarding-class af11;
            dscp 12;
        }
    }
    term t3 {
        from {
            protocol tcp; # For BGP
        }
        then {
            forwarding-class af16;
            dscp 10;
        }
    }
    term t4 {
        then accept; # Do not forget!
    }
}

[edit interfaces]
lo0 {
    unit 0 {
        family inet {
            filter {
                output loopback-filter;
            }
        }
    }
}
```

NOTE: This is not a complete router configuration. You still have to assign resources to the queues, configure the routing protocols, addresses, and so on.

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

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 routers. 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 router 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 {
    so-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 router.
- 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.

Change the Default Queuing and Marking of Host Outbound Traffic

IN THIS SECTION

You can modify the default forwarding class and DSCP bits used in the ToS field of *host outbound traffic* (packets generated by the Routing Engine). You can also modify the queue assignment of the forwarding class to which you assign host outbound traffic.

Host Outbound Traffic Classification Overview

TCP-related packets, such as BGP or LDP, use queue 3 (network control) for retransmitted traffic. Changing the defaults for Routing Engine sourced traffic does not affect transit or incoming traffic. The changes apply to all packets relating to Layer 3 and Layer 2 protocols, but not MPLS EXP bits or IEEE 802.1p bits. This feature applies to all application-level traffic such as FTP or ping operations as well.

The forwarding class selected is global to the device. That is, the traffic is placed in the selected forwarding class on all egress interfaces. In the case of a restricted interface, the Routing Engine sourced traffic flows through the restricted queue.

The forwarding class selected must be properly configured on all interfaces.

To change the default forwarding class and DSCP bits for Routing Engine sourced traffic, include the `host-outbound-traffic` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
host-outbound-traffic {
    forwarding-class class-name;
    dscp-code-point value;
}
```

The following example places all Routing Engine sourced traffic into the network control forwarding class (assigned to queue 3 by default) with a DSCP value of 101010:

```
[edit class-of-service]
host-outbound-traffic {
    forwarding-class network-control;
    dscp-code-point 101010;
}
```

You can also modify the queue assignment of the forwarding class. For example:

```
[edit class-of-service]
forwarding-classes {
    class network-control queue-num 5;
}
```

Platform-Specific Host Outbound Traffic Classification Behavior

Use Feature Explorer to confirm platform and release support for host outbound traffic classification.

Use the following table to review platform-specific behaviors for your platform:

Platform	Difference
ACX7000 Series Routers	ACX7000 Series Routers support only the `forwarding-class` *`class-name`* option.

RELATED DOCUMENTATION

Understanding How Forwarding Classes Assign Classes to Output Queues | 286

Default Routing Engine Protocol Queue Assignments | 319

Default DSCP and DSCP IPv6 Classifiers | 65

Example: Configure Different Queuing and Marking Defaults for Outbound Routing Engine and Distributed Protocol Handler Traffic

IN THIS SECTION

This example shows how to configure a supported router in an IPv4 network so that traffic generated by the Routing Engine and traffic generated by the distributed protocol handler are assigned to different non-default queues and marked with different non-default DiffServ code point (DSCP) bits on all egress interfaces.

This configuration enables you to configure network-wide prioritization to control plane protocol hello packets and keep-alive packets generated by the router. This feature is supported for egress interfaces.

Requirements

This example uses the following hardware and software components:

- Two MX routers, R1 and R2, each with a 20-port Gigabit Ethernet MIC. The two routers are directly connected over an IPv4 network.
- Any supported Junos OS release.

Before you configure this example, configure a Bidirectional Forwarding Detection (BFD) session from port ge-1/0/19 on Router R1 and port ge-1/1/0 on Router R2.

Overview

In this example, you configure an MX router in an IPv4 network so that traffic generated by the Routing Engine and traffic generated by the distributed protocol handler are assigned to different non-default queues and marked with different non-default DSCP bits.

- Distributed protocol handler sourced traffic is placed in queue 7 on all egress interfaces. Of those packets, Layer 3 packets are marked at egress with DSCP bits 001010.
- Routing Engine sourced traffic is placed in queue 6 on all egress interfaces. Of those packets, Layer 3 packets are marked at egress with DSCP bits 000011.

Because the MX router in this example has interfaces hosted on a 20-port Gigabit Ethernet MIC, you can override the default queuing and DSCP marking behavior of host outbound traffic by including configuration statements at the `[edit class-of-service host-outbound-traffic]` hierarchy level. In this example, you use the `forwarding-class` and `dscp-code-point` statements to specify the override values for traffic generated by the distributed protocol handler.

NOTE: This configuration also affects traffic generated by the Routing Engine.

To configure different queuing and DSCP marking of Routing Engine sourced traffic, you must apply a second override configuration. You configure an IPv4 firewall filter that uses the `forwarding-class` and `dscp` actions to specify the override values, and you attach that filter to the egress of the router loopback address. This configuration affects the Routing Engine sourced traffic but not the distributed protocol handler sourced traffic.

Configuration

IN THIS SECTION

To configure different queuing and DSCP marking defaults for egress Routing Engine and distributed protocol handler traffic, perform these tasks:

CLI Quick Configuration

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

Router R1

```
set firewall family inet filter f_bfd_source term 1 from forwarding-class control-traffic then
count c_sent_bfd
set firewall family inet filter f_bfd_source term 1 then accept
set firewall family inet filter f_bfd_source term 2 from forwarding-class-except control-traffic
```

```
then count c_sent_other
set firewall family inet filter f_bfd_source term 2 then accept
set forwarding-options family inet filter output bfd_source
```

Router R2

```
set class-of-service forwarding-classes queue-num 7 bfd_keepalive
set class-of-service host-outbound-traffic forwarding-class bfd_keepalive
set class-of-service host-outbound-traffic dscp-code-point 110000
set class-of-service forwarding-classes queue-num 6 re_control
set firewall family inet filter f_out_loopback term 1 then forwarding-class re_control
set firewall family inet filter f_out_loopback term 1 then dscp 001010
set firewall family inet filter f_out_loopback term 1 then accept
set interfaces lo0 unit 0 family inet filter output f_out_loopback
```

Configuring R1 Packet Counting

Step-by-Step Procedure

To configure Router R1 to count packets that arrive marked for the `network-control` forwarding class:

1. Configure the IPv4 firewall filter term that counts packets marked for the `network-control` forwarding class.

```
[edit]
user@R1# set firewall family inet filter f_bfd_source term 1 from forwarding-class control-
traffic then count c_sent_bfd
user@R1# set firewall family inet filter f_bfd_source term 1 then accept
```

2. Configure the IPv4 firewall filter term that counts all other packets.

```
[edit]
user@R1# set firewall family inet filter f_bfd_source term 2 from forwarding-class-except
control-traffic then count c_sent_other
user@R1# set firewall family inet filter f_bfd_source term 2 then accept
```

3. Apply the firewall filter to all egress packets.

```
[edit]
user@R1# set forwarding-options family inet filter output bfd_source
```

Configuring R2 Queuing and Re-Marking of Host Outbound Traffic

Step-by-Step Procedure

To configure Router R2 to place host outbound traffic in queue 7 and re-mark Layer 3 packets with DSCP bits 110000:

1. Define the `bfd_keepalive` forwarding class and map it to queue 7.

```
[edit]
user@R2# set class-of-service forwarding-classes queue-num 7 bfd_keepalive
```

2. Configure the router to place distributed protocol handler sourced traffic (and also Routing Engine sourced traffic) in queue 7 on all egress interfaces.

```
[edit]
user@R2# set class-of-service host-outbound-traffic forwarding-class bfd_keepalive
```

3. Configure the router to re-mark Layer 3 distributed protocol handler sourced traffic (and also Routing Engine sourced traffic) with DSCP bits 110000, which is compatible with ToS bits 1100 0000.

```
[edit]
user@R2# set class-of-service host-outbound-traffic dscp-code-point 110000
```

Configuring R2 Queuing and Re-Marking of Routing Engine Sourced Traffic

Step-by-Step Procedure

To configure Router R2 to place Routing Engine sourced traffic only in queue 6 and re-mark Layer 3 packets with DSCP bits 001010:

1. Define the re_control forwarding class and map it to queue 6.

```
[edit]
user@R2# set class-of-service forwarding-classes queue-num 6 re_control
```

2. Define the IPv4 firewall filter f_out_loopback that places matched packets in queue 6, re-marks matched Layer 3 packets with DSCP bits 001010, and accepts all matched packets.

```
[edit]
user@R2# set firewall family inet filter f_out_loopback term 1 then forwarding-class
re_control
user@R2# set firewall family inet filter f_out_loopback term 1 then dscp 001010
user@R2# set firewall family inet filter f_out_loopback term 1 then accept
```

3. Attach the filter to the output of the router's loopback address so that the filter actions apply to Routing Engine sourced traffic only.

```
[edit]
user@R2# set interfaces lo0 unit 0 family inet filter output f_out_loopback
```

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

```
[edit]
user@R2# commit
```

Results

From configuration mode, confirm your configuration by entering the show class-of-service, show firewall, show forwarding-options, and show interfaces commands. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

Router R1

```
user@R1# show firewall
family inet {
    filter f_bfd_source {
        term 1 {
            from {
                forwarding-class control-traffic;
            }
            then {
                count c_sent_bfd;
                accept;
            }
        }
        term 2 {
            from {
                forwarding-class-except control-traffic;
            }
            then {
                count c_sent_other;
                accept;
            }
        }
    }
}
```

```
user@R1# show forwarding-options
family inet {
    filter {
        output bfd_source;
    }
}
```

Router R2

```
user@R2# show class-of-service
forwarding-classes {
    queue-num 6 re_control;
    queue-num 7 bfd_keepalive;
}
host-outbound-traffic {
    forwarding-class bfd_keepalive;
    dscp-code-point 110000;
}
```

```
user@R2# show firewall
family inet {
    filter f_out_loopback {
        term 1 {
            then {
                forwarding-class re_control;
                dscp 001010;
                accept;
            }
        }
    }
}
```

```
user@R2# show interfaces
lo0 {
    unit 0 {
        family inet {
            filter {
                output f_out_loopback;
            }
        }
    }
}
```

Verification

IN THIS SECTION

Before you begin verification, enable BFD sessions on both routers.

Confirm that the configuration is working properly.

Verifying the Queue Assignment of the Traffic That R1 Is Sending in the BFD Session

Purpose

Verify the class of service (CoS) forwarding class assignments and type of traffic sent from the BFD source endpoint on Router R1.

Action

From operational mode on Router R1, check that BFD packets are sent out the session endpoint on Router R1. With no CoS configuration present, the command output displays statistics about queued and transmitted traffic for the four forwarding classes and four egress queues in use.

```
user@R1> show interfaces queue ge-1/0/19 egress
Physical interface: ge-1/0/19, Enabled, Physical link is Up
  Interface index: 175, SNMP ifIndex: 121
Forwarding classes: 8 supported, 4 in use
Egress queues: 4 supported, 4 in use
Queue: 0, Forwarding classes: best-effort
  Queued:
    ...
  Transmitted:
    ...
Queue: 1, Forwarding classes: expedited-forwarding
  Queued:
    ...
 Transmitted:
    ...
Queue: 2, Forwarding classes: assured-forwarding
  Queued:
    ...
  Transmitted:
    ...
Queue: 3, Forwarding classes: network-control
  Queued:
    ...
  Transmitted:
    ...
```

Meaning

The statistics for egress queue 3 reflect BFD session traffic sent to Router R2.

Verifying That Router R1 Is Sending BFD Traffic

Purpose

Verify that Router R1 is sending BFD packets from its BFD session endpoint.

Action

From operational mode on Router R1, check that the count of BFD packets that R1 sends out the BFD session endpoint continues to increment.

```
user@R1> clear firewall filter f_bfd_source
user@R1> show firewall filter f_bfd_source
Filter: bfd_source
Counters:
Name                                Bytes              Packets
c_sent_bfd                           2770                   70
c_sent_other                            0                    0
```

```
user@R1> show firewall filter f_bfd_source
Filter: bfd_source
Counters:
Name                                Bytes              Packets
c_sent_bfd                        2182022                39482
c_sent_other                            0                    0
```

Verifying That Router R2 Is Receiving BFD Traffic

Purpose

Verify that Router R2 is receiving BFD packets at its BFD session endpoint.

Action

From operational mode on router R2, check that the BFD session endpoint receives packets destined for the Routing Engine with DSCP bits set to 110000, the default DSCP CoS value for the `network-control` forwarding class. The DSCP bits 110000 map to ToS bits 1100 0000, or 0xC0.

```
user@R2> monitor traffic extensive ge-1/1/0 layer2-headers
Address resolution is ON. Use <no-resolve> to avoid any reverse lookup delay.
Address resolution timeout is 4s.
Listening on ge-1/1/0, capture size 1514 bytes

03:23:10.830472 bpf_flags 0x83,  In
        Juniper PCAP Flags [Ext, no-L2, In], PCAP Extension(s) total length 16
          Device Media Type Extension TLV #3, length 1, value: Ethernet (1)
          Logical Interface Encapsulation Extension TLV #6, length 1, value: Ethernet (14)
          Device Interface Index Extension TLV #1, length 2, value: 132
          Logical Interface Index Extension TLV #4, length 4, value: 68
        -----original packet-----
        PFE proto 2 (ipv4): (tos 0xc0, ttl 255, id 1511, offset 0, flags [none], proto: UDP
(17), length: 52) 10.1.1.1.bfd-src > 10.1.1.2.bfd-ip: [udp sum ok]
        BFDv1, length: 24
        One-hop Control, State Up, Flags: [Control Plane Independent], Diagnostic: No Diagnostic
(0x00)
        Detection Timer Multiplier: 3 (30000 ms Detection time), BFD Length: 24
        My Discriminator: 0x00000002, Your Discriminator: 0x00000001
          Desired min Tx Interval:    10000 ms
          Required min Rx Interval:   10000 ms
          Required min Echo Interval:    0 ms
```

Meaning

The example input packet entry confirms that the original packet was marked with `tos 0xC0`, which correlates to the default forwarding class `network-control`.

RELATED DOCUMENTATION

Understanding Queuing and Marking of Host Outbound Traffic | **316**

Change the Default Queuing and Marking of Host Outbound Traffic | **328**

monitor traffic

show firewall

show interfaces queue

Overriding the Input Classification

For IPv4 or IPv6 packets, you can override the incoming classification, assigning them to the same forwarding class based on their input interface, input precedence bits, or destination address. You do so by defining a policy class when configuring CoS properties and referencing this class when configuring a routing policy.

When you override the classification of incoming packets, any mappings you configured for associated precedence bits or incoming interfaces to output transmission queues are ignored. Also, if the packet loss priority (PLP) bit was set in the packet by the incoming interface, the PLP bit is cleared.

To override the input packet classification, do the following:

1. Define the policy class by including the `class` statement at the `[edit class-of-service forwarding-policy]` hierarchy level:

   ```
   [edit class-of-service]
   forwarding-policy {
       class class-name {
           classification-override {
               forwarding-class class-name;
           }
       }
   }
   ```

   class-name is a name that identifies the routing policy class.

2. Associate the policy class with a routing policy by including it in a `policy-statement` statement at the `[edit policy-options]` hierarchy level. Specify the destination prefixes in the `route-filter` statement and the CoS policy class name in the `then` statement.

   ```
   [edit policy-options]
   policy-statement policy-name {
       term term-name {
           from {
               route-filter destination-prefix match-type <class class-name>
           }
   ```

```
        then class class-name;
    }
}
```

3. Apply the policy by including the `export` statement at the `[edit routing-options]` hierarchy level:

```
[edit routing-options]
forwarding-table {
    export policy-name;
}
```

RELATED DOCUMENTATION

classification-override

CHAPTER 6

Defining Output Queue Properties with Schedulers

IN THIS CHAPTER

How Schedulers Define Output Queue Properties

IN THIS SECTION

You use *schedulers* to define the class-of-service (CoS) properties of output queues. You configure CoS properties in a scheduler, then map the scheduler to a forwarding class. Forwarding classes are in turn mapped to output queues. Classifiers map incoming traffic into forwarding classes based on CoS values in well-known packet header fields (behavior aggregate classification) or on multiple packet header fields (multifield classification).

Output queue properties include the amount of interface bandwidth assigned to the queue, the size of the memory buffer allocated for storing packets, the scheduling priority of the queue, and the random early detection (RED) drop profiles associated with the queue to control packet drop during periods of congestion.

Scheduler maps map schedulers to forwarding classes. The output queue mapped to a forwarding class receives the port resources and properties defined in the scheduler mapped to that forwarding class. You apply a scheduler map to an interface to apply queue scheduling to a port. You can associate different scheduler maps with different interfaces to configure port-specific scheduling for forwarding classes (output queues).

To configure class-of-service (CoS) schedulers, include the following statements at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
interfaces {
    interface-name {
        scheduler-map map-name;
        scheduler-map-chassis map-name;
        shaping-rate rate;
        unit {
            output-traffic-control-profile profile-name;
            scheduler-map map-name;
            shaping-rate rate;
        }
    }
}
fabric {
    scheduler-map {
        priority (high | low) scheduler scheduler-name;
    }
}
scheduler-maps {
    map-name {
        forwarding-class class-name scheduler scheduler-name;
    }
}
schedulers {
    scheduler-name {
        buffer-size (percent percentage | remainder | temporal
microseconds                            );
        drop-profile-map loss-priority (any | low | medium-low | medium-high | high)protocol (any |
non-tcp | tcp) drop-profile profile-name;
        excess-priority (low | high);
        excess-rate percent percentage;
        excess-rate (percent percentage | proportion value);
        priority priority-level;
        transmit-rate (rate | percent percentage remainder) <exact | rate-limit>;
    }
}
traffic-control-profiles profile-name {
    delay-buffer-rate (percent percentage | rate);
    excess-rate percent percentage;
    guaranteed-rate (percent percentage | rate);
    scheduler-map map-name;
    shaping-rate (percent percentage | rate);
}
```

You cannot configure both the `shaping-rate` statement at the `[edit class-of-service interfaces` *`interface-name`*`]` hierarchy level and the `transmit-rate rate-limit` statement and option at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level. These statements are mutually exclusive. If you do configure both, you will not be able to commit the configuration:

```
[edit class-of-service]
'shaping-rate'
only one option (shaping-rate or transmit-rate rate-limit) can be configured at a time
error: commit failed (statements constraint check failed)
```

NOTE: For PTX Series Packet Transport Routers:

- The `fabric` and `traffic-control-profiles` statements at the `[edit class-of-service]` hierarchy level are not supported.

Queue Scheduling Components

Table 27 on page 345 provides a quick reference to the scheduler components you can configure to determine the bandwidth properties of output queues (forwarding classes).

Table 27: Output Queue Scheduler Components

Output Queue Scheduler Component	Description
Buffer size	Sets the size of the queue buffer.
Drop profile map	Maps a drop profile to a packet loss priority. Drop profile map components include: • Drop profile—Sets the probability of dropping packets as the queue fills up. • Loss priority—Sets the traffic packet loss priority to which a drop profile applies.
Excess priority	Sets the scheduling priority of excess bandwidth traffic on a scheduler.
Excess rate	Sets the percentage of extra bandwidth (bandwidth that is not used by other queues) a queue can receive. If not set, the device uses the transmit rate to determine how much extra bandwidth the queue can use. Extra bandwidth is the bandwidth remaining after all guaranteed bandwidth requirements are met.
Priority	Sets the scheduling priority applied to the queue.
Shaping rate	Sets a limit on excess bandwidth usage. The transmit rate configures the minimum bandwidth allocated to a queue. Configure the shaping rate as an absolute maximum usage and not the additional usage beyond the configured transmit rate. If you do not set a shaping rate, the default shaping rate is 100 percent, which is the same as no shaping at all.

Table 27: Output Queue Scheduler Components *(Continued)*

Output Queue Scheduler Component	Description
Transmit rate	Sets the minimum guaranteed bandwidth . By default, if you do not configure an excess rate, extra bandwidth is shared among queues in proportion to the transmit rate of each queue. On strict-high priority queues, sets the amount of bandwidth that receives strict-high priority forwarding treatment. Traffic that exceeds the transmit rate shares in the port excess bandwidth pool based on the strict-high priority excess bandwidth sharing weight of "1", which is not configurable. The actual amount of extra bandwidth that traffic exceeding the transmit rate receives depends on how many other queues consume excess bandwidth and the excess rates of those queues. If you configure two or more strict-high priority queues on a port, you must configure a transmit rate on those queues. However, we strongly recommend that you always configure a transmit rate on strict-high priority queues to prevent them from starving other queues. **NOTE**: On PTX10008 and PTX10016 routers, the default scheduler transmission rate is set to 25 percent for strict-high queues if you do not set the transmission rate for the schedulers for strict-high queues.

RELATED DOCUMENTATION

Understanding How Forwarding Classes Assign Classes to Output Queues | **286**

Default Schedulers Overview | **346**

Configuring Schedulers | **348**

Default Schedulers Overview

Each forwarding class has an associated scheduler priority. Only two forwarding classes, best effort and network control (queue 0 and queue 3), are used in the Junos default scheduler configuration.

By default, the BE forwarding class (queue 0) receives 95 percent of the bandwidth and buffer space for the output link, and the network control forwarding class (queue 3) receives 5 percent. The default drop profile causes the buffer to fill and then discard all packets until it has space.

The EF (queue 1) and AF (queue 2) classes have no reserved bandwidth or buffer space because, by default, no schedulers are assigned to those forwarding classes. However, you can manually configure resources for the EF and AF classes.

Also by default, each queue can exceed the assigned bandwidth if additional bandwidth is available from other queues. When a forwarding class does not fully use the allocated transmission bandwidth, the remaining bandwidth can be used by other forwarding classes if they receive a larger amount of the offered load than the bandwidth allocated. For more information, see "Allocation of Leftover Bandwidth" on page 375.

The following default scheduler is provided on Junos routers. These settings are not visible in the output of the `show class-of-service` command; rather, these settings are implicit.

```
[edit class-of-service]
schedulers {
    network-control {
        transmit-rate percent 5;
        buffer-size percent 5;
        priority low;
        drop-profile-map loss-priority any protocol any drop-profile terminal;
    }
    best-effort {
        transmit-rate percent 95;
        buffer-size percent 95;
        priority low;
        drop-profile-map loss-priority any protocol any drop-profile terminal;
    }
}
drop-profiles {
    terminal {
        fill-level 100 drop-probability 100;
    }
}
```

Configuring Schedulers

You configure schedulers to assign resources, priorities, and drop profiles to output queues. To configure a scheduler, include the `scheduler` statement at the `[edit class-of-service]` hierarchy level:

```
schedulers {
    scheduler-name {
        buffer-size (percent percentage | remainder | temporal microseconds);
        drop-profile-map loss-priority (any | low | medium-low | medium-high | high)protocol (any |
non-tcp | tcp) drop-profile profile-name;
        priority priority-level;
        shaping-rate (percent percentage | rate);
        transmit-rate (rate | percent percentage remainder) <exact | rate-limit>;
    }
}
```

NOTE: Committing changes to schedulers and queues interrupts traffic on affected ports while queue resources are reconfigured.

For detailed information about scheduler configuration statements, see the indicated topics:

- "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476
- "Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers" on page 469
- "Configuring Scheduler Transmission Rate" on page 373

Configuring Scheduler Maps

You configure a *scheduler map* to map forwarding classes to a various schedulers. To create a scheduler map, include the `scheduler-maps` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
scheduler-maps {
    map-name {
        forwarding-class class-name scheduler scheduler-name;
    }
}
```

Applying Scheduler Maps Overview

Physical interfaces (for example, `xe-0/0/0` and `ge-0/0/0`) support scheduling with any encapsulation type pertinent to that physical interface. For a single port, you cannot apply scheduling to the physical interface if you have applied scheduling to one or more of the associated logical interfaces.

Logical interfaces (for example, `xe-0/0/0 unit 0` and `ge-0/0/0 unit 0`) support scheduling on VLANs only.

In the Junos OS implementation, the term *logical interfaces* generally refers to interfaces you configure by including the `unit` statement at the `[edit interfaces` *`interface-name`*`]` hierarchy level. Logical interfaces have the `.`*`logical`* descriptor at the end of the interface name, as in `ge-0/0/0.1` or `xe-0/0/0:0.1`, where the logical unit number is `1`.

Within the `[edit class-of-service]` hierarchy level, you cannot use the `.`*`logical`* descriptor when you assign properties to logical interfaces. Instead, you must include the `unit` statement in the configuration. For example:

```
[edit class-of-service]
user@host# set interfaces t3-0/0/0 unit 0 scheduler-map map1
```

RELATED DOCUMENTATION

Applying Scheduler Maps to Physical Interfaces | **350**

Applying Scheduler Maps and Shaping Rate to Logical Interfaces and VLANs | **394**

Oversubscribing Interface Bandwidth | **366**

Providing a Guaranteed Minimum Rate | **376**

Forwarding Classes and Fabric Priority Queues | **319**

Associating Schedulers with Fabric Priorities | **432**

Applying Scheduler Maps to Physical Interfaces

After you have defined a scheduler map, as described in "Configuring Scheduler Maps" on page 348, you can apply it to an output interface. Include the `scheduler-map` statement at the `[edit class-of-service interfaces interface-name]` hierarchy level:

```
[edit class-of-service interfaces interface-name]
scheduler-map map-name;
```

Interface wildcards are supported. However, scheduler maps using wildcard interfaces are not checked against routing device interfaces at commit time and can result in a configuration that is incompatible with installed hardware. Fully specified interfaces, on the other hand, check the configuration against the hardware and report errors or warning if the hardware does not support the configuration.

Generally, you can associate schedulers with physical interfaces only. For some IQ interfaces, you can also associate schedulers with the logical interface. For more information, see "Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs" on page 394.

NOTE: For original Channelized OC12 PICs, limited CoS functionality is supported. For more information, contact Juniper Networks customer support.

When you apply a scheduler map to a physical interface, or when you modify the configuration of a scheduler map that is already applied to a physical interface, packets already in the output queues of the interface might get dropped. The amount of packet loss is not deterministic and depends on the offered traffic load at the time you apply or modify the scheduler map.

Configuring Traffic Control Profiles for Shared Scheduling and Shaping

Shared scheduling and shaping allows you to allocate separate pools of shared resources to subsets of logical interfaces belonging to the same physical port. You configure shared scheduling and shaping by first creating a traffic-control profile, which specifies a shaping rate and references a scheduler map. You must then share this set of shaping and scheduling resources by applying an instance of the traffic-control profile to a subset of logical interfaces. You can apply a separate instance of the same (or a different) traffic-control profile to another subset of logical interfaces, thereby allocating separate pools of shared resources.

Before you start this procedure:

- Make sure you define a scheduler map. For information about configuring schedulers and scheduler maps, see "Configuring Schedulers" on page 348 and "Configuring Scheduler Maps" on page 348. Gigabit Ethernet IQ2 interfaces support up to eight forwarding classes and queues.

To configure a traffic-control profile, perform the following steps:

1. Create the traffic control profile and configure a shaping rate for it.

   ```
   [edit]
   user@host# edit class-of-service traffic-control-profiles profile-name
   user@host# set shaping-rate (percent percentage | rate)
   ```

   You can configure the shaping rate as a percentage from 1 through 100 or as an absolute rate from 1000 through 6,400,000,000,000 bits per second (bps). The shaping rate corresponds to a peak information rate (PIR). For more information, see "Oversubscribing Interface Bandwidth" on page 366.
2. Define an association between the traffic-control profile and a previously configured scheduler map by including the `scheduler-map` statement at the `[edit class-of-service traffic-control-profiles` *`profile-name`*`]` hierarchy level.

   ```
   [edit class-of-service traffic-control-profiles profile-name]
   user@host# set scheduler-map map-name;
   ```
3. Configure the delay-buffer rate.

   If you do not include this statement, the delay-buffer rate is based on the guaranteed rate if one is configured, or on the shaping rate if no guaranteed rate is configured.

   ```
   [edit class-of-service traffic-control-profiles profile-name]
   user@host# set delay-buffer-rate (percent percentage | rate)
   ```

   You can configure the `delay-buffer` rate as a percentage from 1 through 100 or as an absolute rate from 1000 through 6,400,000,000,000 bits per second. The delay-buffer rate controls latency. For more information, see "Oversubscribing Interface Bandwidth" on page 366 and "Providing a Guaranteed Minimum Rate" on page 376.
4. Configure a guaranteed minimum rate for the traffic-control profile.

   ```
   [edit class-of-service traffic-control-profiles profile-name]
   user@host# set guaranteed-rate (percent percentage | rate)
   ```

You can configure the guaranteed rate as a percentage from 1 through 100 or as an absolute rate from 1000 through 6,400,000,000,000 bps. The guaranteed rate corresponds to a committed information rate (CIR). For more information, see "Providing a Guaranteed Minimum Rate" on page 376.

You must now share an instance of the traffic-control profile.

5. Enable shared-scheduling on the interface.

```
[edit]
user@host# edit interfaces interface-name
user@host# set shared-scheduler
```

This statement enables logical interfaces belonging to the same physical port to share one set of shaping and scheduling resources.

NOTE: On each physical interface, the `shared-scheduler` and `per-unit-scheduler` statements are mutually exclusive. Even so, you can configure one logical interface for each shared instance. This effectively provides the functionality of per-unit scheduling.

6. (Optional) Apply the traffic-control profile to an input interface.

```
[edit]
user@host# edit class-of-service interfaces interface-name unit logical-unit-number
user@host# set input-traffic-control-profile profile-name shared-instance instance-name
```

These statements are explained in Step 7.

7. (Optional) Apply the traffic-control profile to an output interface.

```
[edit]
user@host# edit class-of-service interfaces interface-name unit logical-unit-number
user@host# set output-traffic-control-profile profile-name shared-instance instance-name
```

The profile name references the traffic-control profile you configured in Step 1 through Step 4. The `shared-instance` name does not reference a configuration. It can be any text string you wish to apply to multiple logical interfaces that you want to share the set of resources configured in the traffic-control profile. Each logical interface shares a set of scheduling and shaping resources with other logical interfaces that are on the same physical port and that have the same shared-instance name applied.

NOTE: You cannot include the `output-traffic-control-profile` statement in the configuration if either the `scheduler-map` or `shaping-rate` statement is included in the logical interface configuration.

RELATED DOCUMENTATION

Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | 757

Applying Scheduler Maps and Shaping Rate to Logical Interfaces and VLANs | 394

Hierarchical Schedulers and Traffic Control Profiles | 447

Configuring an Input Scheduler on an Interface

As an alternative to shared input traffic-control profiles, you can configure each interface to use its own input scheduler. For each physical interface, you can apply an input scheduler map to the physical interface or its logical interfaces, but not both.

Before you start this procedure:

- Define a scheduler map at the `[edit class-of-service scheduler-maps]` hierarchy level.

To configure a separate input scheduler on the physical interface:

- Specify the name of the physical interface and the scheduler map.

```
[edit class-of-service interfaces interface-name]
user@host# set input-scheduler-map map-name
```

To configure a separate input scheduler on a logical interface:

1. Specify the name of the physical and logical interface and scheduler map.

```
[edit]
user@host# edit class-of-service interfaces interface-name unit logical-unit-number
user@host# set input-scheduler-map map-name
```

2. Enable the association of the scheduler map name and interface.

```
[edit]
user@host# edit interfaces interface-name
user@host# set per-unit-scheduler
```

The `per-unit-scheduler` statement enables one set of output queues for each logical interface configured under the physical interface.

RELATED DOCUMENTATION

Configuring Schedulers | 348

Configuring Scheduler Maps | 348

input-traffic-control-profile

Understanding Interface Sets

Although the interface set is applied at the `[edit interfaces]` hierarchy level, the CoS parameters for the interface set are defined at the `[edit class-of-service interfaces]` hierarchy level, usually with the `output-traffic-control-profile` *`profile-name`* statement.

This example applies a traffic control profile called `tcp-set1` to an interface set called `set-ge-0`:

```
[edit class-of-service interfaces]
interface-set set-ge-0 {
    output-traffic-control-profile tcp-set1;
}
```

RELATED DOCUMENTATION

output-traffic-control-profile

Configuring Interface Sets

To configure an interface set, include the `interface-set` statement at the `[edit class-of-service interfaces]` hierarchy level:

```
[edit class-of-service interfaces]
interface-set interface-set-name {
    ...interface-cos-configuration-statements ...
}
```

To apply the interface set to interfaces, include the `interface-set` statement at the `[edit interfaces]` hierarchy level:

```
[edit interfaces]
interface-set interface-set-name {
    interface ethernet-interface-name {
        ... interface-cos-configuration-statements ...
    }
}
```

Interface sets can be defined in two major ways:

- As a list of logical interfaces or aggregated Ethernet interfaces (`unit 100`, `unit 200`, and so on)
- At the stacked VLAN level using a list of outer VLAN IDs (`vlan-tags-outer 210`, `vlan-tags-outer 220`, and so on).

  The `svlan` *`number`* listing option with a single outer VLAN tag is a convenient way to specify a set of VLAN members having the same outer VLAN tags. Service providers can use these statements to group interfaces to apply scheduling parameters such as guaranteed rate and shaping rate to the traffic in the groups.

Whether using the logical interface listing option for a group of customer VLANs, aggregated Ethernet interfaces, or the S-VLAN set listing option for a group of VLAN outer tags, all traffic heading downstream must be gathered into an interface set with the `interface-set` statement at the `[edit class-of-service interfaces]` hierarchy level.

Regardless of listing convention, you can only use one of the types in an interface set. Examples of this limitation appear later in this section.

NOTE: Interface sets are currently used only by CoS, but they are applied at the `[edit interfaces]` hierarchy level to make them available to other services that might use them in future.

```
[edit interfaces]
interface-set interface-set-name {
    interface ethernet-interface-name {
        (unit logical-unit-number | vlan-tags-outer vlan-tag) {
            ...
        }
    }
}
```

The logical interface naming option lists Ethernet interfaces:

```
[edit interfaces]
interface-set unitl-set-ge-0 {
    interface ge-0/0/0 {
        unit 0;
        unit 1;
        ...
    }
}
```

The interface naming option lists aggregated Ethernet interfaces:

```
[edit interfaces]
    interface-set demuxset1 {
        interface demux0 {
            unit 1;
            ..
        }
    }
    demux0 {
        unit 1 {
            demux-options {
                underlying-interface ae0.1;
            }
            family inet {
                demux-source {
                    10.1.1.1/24;
                }
                address 10.1.1.1/24;
            }
        }
        ..
        ae0 {
            unit 1 {
            }
            ..
        }
    }
    class-of-service {
        interface-set demuxset1 {
            output-traffic-control-profile tcp2;
        }
    }
}
```

The S-VLAN option lists only one S-VLAN (outer) tag value:

```
[edit interfaces]
interface-set svlan-set {
    interface ge-1/0/0 {
        vlan-tags-outer 2000;
    }
}
```

The S-VLAN naming option lists S-VLAN (outer) tag values:

```
[edit interfaces]
interface-set svlan-set-tags {
    interface ge-2/0/0 {
        vlan-tags-outer 2000;
        vlan-tags-outer 2001;
        vlan-tags-outer 2002;
        ...
    }
}
```

NOTE: Ranges are not supported: you must list each VLAN or logical interface separately.

RELATED DOCUMENTATION

Interface Set Caveats | 358

Interface Set Caveats

When configuring interface sets, consider the following guidelines:

- Interface sets can be defined in two major ways: as a list of logical interfaces or groups of aggregated Ethernet logical interfaces (`unit 100`, `unit 200`, and so on), or at the stacked VLAN level using a list of outer VLAN IDs (`vlan-tags-outer 210`, `vlan-tags-outer 220`, and so on). You can configure sets of aggregated Ethernet interfaces on MIC or MPC interfaces only.
- You cannot specify an interface set mixing the *logical interface*, aggregated Ethernet, S-VLAN, or VLAN outer tag list forms of the `interface-set` statement.
- Keep the following guidelines in mind when configuring interface sets of logical interfaces over aggregated Ethernet:
  - Sets of aggregated Ethernet interfaces are supported on MIC and MPC interfaces only.
  - The supported interface stacks for aggregated Ethernet in an interface set include VLAN demux interfaces, IP demux interfaces, and PPPoE logical interfaces over VLAN demux interfaces.
  - The link membership list and scheduler mode of the interface set are inherited from the underlying aggregated Ethernet interface over which the interface set is configured.
  - When an aggregated Ethernet interface operates in link protection mode, or if the scheduler mode is configured to replicate member links, the scheduling parameters of the interface set are copied to each of the member links.
  - If the scheduler mode of the aggregated Ethernet interface is set to scale member links, the scheduling parameters are scaled based on the number of active member links and applied to each of the aggregated interface member links.
- A logical interface can only belong to one interface set. If you try to add the same logical interface to different interface sets, the commit operation fails.

This example generates a commit error:

```
[edit interfaces]
interface-set set-one {
    interface ge-2/0/0 {
        unit 0;
        unit 2;
    }
}
interface-set set-two {
    interface ge-2/0/0 {
        unit 1;
        unit 3;
        unit 0; # COMMIT ERROR! Unit 0 already belongs to set-one.
    }
}
```

- When you delete any interface configuration that is part of an interface set, the commit operation fails with the following message: "Interface unit should have configuration defined for adding to interface-set."

- Members of an interface set cannot span multiple physical interfaces. Only one physical interface is allowed to appear in an interface set.

  This configuration is not supported:

```
[edit interfaces]
interface-set set-group {
    interface ge-0/0/1 {
        unit 0;
        unit 1;
    }
    interface ge-0/0/2 { # This is NOT supported in the same interface set!
        unit 0;
        unit 1;
    }
}
```

RELATED DOCUMENTATION

Configuring Interface Sets | 355

Configuring Internal Scheduler Nodes

A node in the hierarchy is considered internal if either of the following conditions apply:

- Any one of its children nodes has a traffic control profile configured and applied.
- You include the `internal-node` statement at the `[edit class-of-service interfaces interface-set` *`set-name`*`]` hierarchy level.

Why would it be important to make a certain node internal? Generally, there are more resources available at the logical interface (unit) level than at the interface set level. Also, it might be desirable to configure all resources at a single level, rather than spread over several levels. The `internal-node` statement provides this flexibility. This can be a helpful configuration device when interface-set queuing without logical interfaces is used exclusively on the interface.

The `internal-node` statement can be used to raise the interface set without children to the same level as the other configured interface sets with children, allowing them to compete for the same set of resources.

In summary, using the `internal-node` statement allows statements to all be scheduled at the same level with or without children.

The following example makes the interfaces sets `if-set-1` and `if-set-2` internal:

```
[edit class-of-service interfaces]
interface-set {
    if-set-1 {
        internal-node;
        output-traffic-control-profile tcp-200m-no-smap;
    }
    if-set-2 {
        internal-node;
        output-traffic-control-profile tcp-100m-no-smap;
    }
}
```

If an interface set has logical interfaces configured with a traffic control profile, then the use of the `internal-node` statement has no effect.

Internal nodes can specify a `traffic-control-profile-remaining` statement.

Example: Configuring and Applying Scheduler Maps

IN THIS SECTION

This example shows how to configure and apply a scheduler map to a device’s interface.

Requirements

Before you begin:

- Create and configure the forwarding classes. See "Configuring a Custom Forwarding Class for Each Queue" on page 291.
- Create and configure the schedulers. See *Example: Configuring Class-of-Service Schedulers on a Security Device*.

Overview

After you define a scheduler, you can include it in a scheduler map, which maps a specified forwarding class to a scheduler configuration. You configure a scheduler map to assign a forwarding class to a scheduler, and then apply the scheduler map to any interface that must enforce DiffServ CoS.

After they are applied to an interface, the scheduler maps affect the hardware queues, packet schedulers, and RED drop profiles.

In this example, you create the scheduler map diffserv-cos-map and apply it to the device's Ethernet interface ge-0/0/0. The map associates the mf-classifier forwarding classes to the schedulers as shown in Table 28 on page 362.

Table 28: Sample diffserv-cos-map Scheduler Mapping

mf-classifier Forwarding Class	For CoS Traffic Type	diffserv-cos-map Scheduler
be-class	Best-effort traffic	be-scheduler
ef-class	Expedited forwarding traffic	ef-scheduler
af-class	Assured forwarding traffic	af-scheduler
nc-class	Network control traffic	nc-scheduler

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

```
 set class-of-service scheduler-maps diffserv-cos-map forwarding-class be-class scheduler be-
scheduler
set class-of-service scheduler-maps diffserv-cos-map forwarding-class ef-class scheduler ef-
scheduler
set class-of-service scheduler-maps diffserv-cos-map forwarding-class af-class scheduler af-
scheduler
set class-of-service scheduler-maps diffserv-cos-map forwarding-class nc-class scheduler nc-
scheduler
set class-of-service  interfaces ge-0/0/0 unit 0 scheduler-map diffserv-cos-map
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure and apply a scheduler map to a device’s interface:

1. Configure a scheduler map for DiffServ CoS.

   ```
   [edit class-of-service]
   user@host# edit scheduler-maps diffserv-cos-map
   ```

2. Configure a best-effort forwarding class and scheduler.

   ```
    [edit class-of-service scheduler-maps diffserv-cos-map]
   user@host# set forwarding-class be-class scheduler be-scheduler
   ```

3. Configure an expedited forwarding class and scheduler.

   ```
   [edit class-of-service scheduler-maps diffserv-cos-map]
   user@host# set forwarding-class ef-class scheduler ef-scheduler
   ```

4. Configure an assured forwarding class and scheduler.

   ```
   [edit class-of-service scheduler-maps diffserv-cos-map]
   user@host# set forwarding-class af-class scheduler af-scheduler
   ```

5. Configure a network control class and scheduler.

   ```
   [edit class-of-service scheduler-maps diffserv-cos-map]
   user@host# set forwarding-class nc-class scheduler nc-scheduler
   ```

6. Apply the scheduler map to an interface.

   ```
   [edit class-of-service]
   user@host# set interfaces ge-0/0/0 unit 0 scheduler-map diffserv-cos-map
   ```

Results

From configuration mode, confirm your configuration by entering the `show class-of-service` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host# show class-of-service
interfaces {
    ge-0/0/0 {
        unit 0 {
            scheduler-map diffserv-cos-map;
        }
    }
}
scheduler-maps {
    diffserv-cos-map {
        forwarding-class be-class scheduler be-scheduler;
        forwarding-class ef-class scheduler ef-scheduler;
        forwarding-class af-class scheduler af-scheduler;
        forwarding-class nc-class scheduler nc-scheduler;
    }
}
```

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

Verification

IN THIS SECTION

Verifying the Scheduler Map Configuration

Purpose

Verify that scheduler maps are configured properly.

Action

From operational mode, enter the `show class-of-service` command.

RELATED DOCUMENTATION

Default Schedulers Overview | 346

Configuring Schedulers | 348

Configuring Scheduler Maps | 348

CHAPTER 7

Controlling Bandwidth with Scheduler Rates

IN THIS CHAPTER

Oversubscribing Interface Bandwidth

IN THIS SECTION

The term *oversubscribing interface bandwidth* means configuring shaping rates (peak information rates [PIRs]) so that their sum exceeds the interface bandwidth.

Logical interfaces can be oversubscribed when there is leftover bandwidth. The oversubscription is capped to the configured PIR. Any unused bandwidth is distributed equally among oversubscribed logical interfaces or physical interfaces.

For networks that are not likely to experience congestion, oversubscribing interface bandwidth improves network utilization, thereby allowing more customers to be provisioned on a single interface. If the actual data traffic does not exceed the interface bandwidth, oversubscription allows you to sell more bandwidth than the interface can support.

We recommend avoiding oversubscription in networks that are likely to experience congestion. Be cautious not to oversubscribe a service by too much, because this can cause degradation in the performance of the routing platform during congestion. When you configure oversubscription, starvation of some output queues can occur if the actual data traffic exceeds the physical interface bandwidth. You can prevent degradation by using statistical multiplexing to ensure that the actual data traffic does not exceed the interface bandwidth.

NOTE: You cannot oversubscribe interface bandwidth when you configure traffic shaping using the method described in "Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs" on page 394.

To configure oversubscription of the interface, perform the following steps:

1. Include the `shaping-rate` statement at the `[edit class-of-service traffic-control-profiles profile-name]` hierarchy level:

   ```
   [edit class-of-service traffic-control-profiles profile-name]
   shaping-rate (percent percentage | rate);
   ```

   Alternatively, you can configure a shaping rate for a logical interface and oversubscribe the physical interface by including the `shaping-rate` statement at the `[edit class-of-service interfaces interface-name unit logical-unit-number]` hierarchy level. However, with this configuration approach, you cannot independently control the delay-buffer rate, as described in Step 2.

2. Optionally, you can base the delay-buffer calculation on a delay-buffer rate. To do this, include the `delay-buffer-rate` statement at the `[edit class-of-service traffic-control-profiles profile-name]` hierarchy level:

   ```
   [edit class-of-service traffic-control-profiles profile-name]
   delay-buffer-rate (percent percentage | rate);
   ```

The delay-buffer rate overrides the shaping rate as the basis for the delay-buffer calculation. In other words, the shaping rate or scaled shaping rate is used for delay-buffer calculations only when the delay-buffer rate is not configured.

The actual delay buffer is based on the calculations described in "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476. For an example showing how the delay-buffer rates are applied, see "Example: Oversubscribing Interface Bandwidth" on page 372.

Configuring large buffers on relatively slow-speed links can cause packet aging. To help prevent this problem, the software requires that the sum of the delay-buffer rates be less than or equal to the port speed.

This restriction does not eliminate the possibility of packet aging, so you should be cautious when using the `delay-buffer-rate` statement. Though some amount of extra buffering might be desirable for burst absorption, delay-buffer rates should not far exceed the service rate of the logical interface.

If you configure delay-buffer rates so that the sum exceeds the port speed, the configured delay-buffer rate is not implemented for the last logical interface that you configure. Instead, that logical interface receives a delay-buffer rate of zero, and a warning message is displayed in the CLI. If bandwidth becomes available (because another logical interface is deleted or deactivated, or the port speed is increased), the configured delay-buffer-rate is reevaluated and implemented if possible.

If you do not configure a delay-buffer rate or a guaranteed rate, the logical interface receives a delay-buffer rate in proportion to the shaping rate and the remaining delay-buffer rate available. In other words, the delay-buffer rate for each logical interface with no configured delay-buffer rate is equal to:

```
(remaining delay-buffer rate * shaping rate) / (sum of shaping rates)
```

where the remaining delay-buffer rate is equal to:

```
(interface speed) - (sum of configured delay-buffer rates)
```

3. To assign a scheduler map to the logical interface, include the `scheduler-map` statement at the `[edit class-of-service traffic-control-profiles` *`profile-name`*`]` hierarchy level:

```
[edit class-of-service traffic-control-profiles profile-name]
scheduler-map map-name;
```

For information about configuring schedulers and scheduler maps, see "Configuring Schedulers" on page 348 and "Configuring Scheduler Maps" on page 348.

4. Optionally, you can enable large buffer sizes to be configured. To do this, include the `q-pic-large-buffer` statement at the `[edit chassis fpc slot-number pic pic-number]` hierarchy level:

```
[edit chassis fpc slot-number pic pic-number]
q-pic-large-buffer;
```

   If you do not include this statement, the delay-buffer size is more restricted. We recommend restricted buffers for delay-sensitive traffic, such as voice traffic. For more information, see "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476.

5. To enable scheduling on logical interfaces, include the `per-unit-scheduler` statement at the `[edit interfaces interface-name]` hierarchy level:

```
[edit interfaces interface-name]
per-unit-scheduler;
```

6. To apply the traffic-scheduling profile , include the output-traffic-control-profile statement at the `[edit class-of-service interfaces interface-name unit logical-unit-number]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
output-traffic-control-profile profile-name;
```

   You cannot include the `output-traffic-control-profile` statement in the configuration if either the `scheduler-map` or `shaping-rate` statement is included in the logical interface configuration.

Table 29 on page 369 shows how the bandwidth and delay buffer are allocated in various configurations.

Table 29: Bandwidth and Delay Buffer Allocations by Configuration Scenario

Configuration Scenario	Delay Buffer Allocation
You do not oversubscribe the interface. You do not configure a guaranteed rate. You do not configure a shaping rate. You do not configure a delay-buffer rate.	Logical interface receives the remaining bandwidth and receives a delay buffer in proportion to the remaining bandwidth.

Table 29: Bandwidth and Delay Buffer Allocations by Configuration Scenario *(Continued)*

Configuration Scenario	Delay Buffer Allocation
You do not oversubscribe the interface. You configure a shaping rate at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level.	For backward compatibility, the shaped logical interface receives a delay buffer based on the shaping rate. The multiplicative factor depends on whether you include the `q-pic-large-buffer` statement. For more information, see "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476. Unshaped logical interfaces receive the remaining bandwidth and a delay buffer in proportion to the remaining bandwidth.
You oversubscribe the interface. You do not configure a guaranteed rate. You do not configure a shaping rate. You do not configure a delay-buffer rate.	Logical interface receives minimal bandwidth with no guarantees and receives a minimal delay buffer equal to four MTU-sized packets.
You oversubscribe the interface. You configure a shaping rate. You do not configure a guaranteed rate. You do not configure a delay-buffer rate.	Logical interface receives a delay buffer based on the scaled shaping rate: `scaled shaping rate = (shaping-rate * [physical interface bandwidth]) / SUM (shaping-rates of all logical interfaces on the physical interface)` The logical interface receives variable bandwidth, depending on how much oversubscription and statistical multiplexing is present. If the amount of oversubscription is low enough that statistical multiplexing does not make all logical interfaces active at the same time and the physical interface bandwidth is not exceeded, the logical interface receives bandwidth equal to the shaping rate. Otherwise, the logical interface receives a smaller amount of bandwidth. In either case, the logical interface bandwidth does not exceed the shaping rate.

Table 29: Bandwidth and Delay Buffer Allocations by Configuration Scenario *(Continued)*

Configuration Scenario	Delay Buffer Allocation
You oversubscribe the interface. You configure a shaping rate. You configure a delay-buffer rate.	Logical interface receives a delay buffer based on the delay-buffer rate. The multiplicative factor depends on whether you include the `q-pic-large-buffer` statement. For more information, see "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476. The logical interface receives variable bandwidth, depending on how much oversubscription and statistical multiplexing is present. If the amount of oversubscription is low enough that statistical multiplexing does not make all logical interfaces active at the same time and the physical interface bandwidth is not exceeded, the logical interface receives bandwidth equal to the shaping rate. Otherwise, the logical interface receives a smaller amount of bandwidth. In either case, the logical interface bandwidth does not exceed the shaping rate.
You oversubscribe the interface. You do not configure a shaping rate. You configure a guaranteed rate. You configure a delay-buffer rate.	Logical interface receives a delay buffer based on the delay-buffer rate.
You oversubscribe the interface. You do not configure a shaping rate. You do not configure a guaranteed rate. You configure a delay-buffer rate.	This scenario is not allowed. If you configure a delay-buffer rate, the traffic-control profile must also include either a shaping rate or a guaranteed rate.
You oversubscribe the interface. You configure a shaping rate. You configure a guaranteed rate. You do not configure a delay-buffer rate.	Logical interface receives a delay buffer based on the guaranteed rate.

Verifying Configuration of Bandwidth Oversubscription

To verify your configuration, you can issue this following operational mode commands:

- `show class-of-service interfaces`
- `show class-of-service traffic-control-profile` *`profile-name`*

Example: Oversubscribing Interface Bandwidth

Oversubscribing a Channelized Interface

Two logical interface units, `0` and `1`, are shaped to rates 2 Mbps and 3 Mbps, respectively. The delay-buffer rates are 750 Kbps and 500 Kbps, respectively. The actual delay buffers allocated to each logical interface are 1 second of 750 Kbps and 2 seconds of 500 Kbps, respectively. The 1-second and 2-second values are based on the following calculations:

```
delay-buffer-rate < [16 x 64 Kbps]): 1 second of delay-buffer-rate
delay-buffer-rate < [8 x 64 Kbps]): 2 seconds of delay-buffer-rate
```

For more information about these calculations, see "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476.

```
chassis {
    fpc 3 {
        pic 0 {
            q-pic-large-buffer;
        }
    }
}
interfaces {
    ge-3/0/0 {
        per-unit-scheduler;
    }
}
class-of-service {
    traffic-control-profiles {
        tc-profile1 {
            shaping-rate 2m;
            delay-buffer-rate 750k; # 750 Kbps is less than 16 x 64 Kbps
            scheduler-map sched-map1;
        }
        tc-profile2 {
            shaping-rate 3m;
            delay-buffer-rate 500k; # 500 Kbps is less than 8 x 64 Kbps
            scheduler-map sched-map2;
        }
    }
    interfaces {
        ge-3/0/0 {
            unit 0 {
                output-traffic-control-profile tc-profile1;
            }
            unit 1 {
                output-traffic-control-profile tc-profile2;
            }
        }
    }
}
```

Configuring Scheduler Transmission Rate

IN THIS SECTION

The transmission rate control determines the actual traffic bandwidth from each forwarding class you configure. The rate is specified in bits per second (bps). Each queue is allocated some portion of the bandwidth of the outgoing interface.

This bandwidth amount can be a fixed value, such as 1 megabit per second (Mbps), a percentage of the total available bandwidth, or the rest of the available bandwidth. You can limit the transmission bandwidth to the exact value you configure, or allow it to exceed the configured rate if additional bandwidth is available from other queues. This property allows you to ensure that each queue receives the amount of bandwidth appropriate to its level of service.

To configure transmission scheduling, include the `transmit-rate` statement at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
transmit-rate (rate | percent percentage | remainder) <exact | rate-limit>;
```

You can specify the transmit rate as follows:

- *rate*—Transmission rate, in bits per second.

- `percent` *`percentage`*—Percentage of transmission capacity.
- `remainder`—Use remaining rate available. In the configuration, you cannot combine the `remainder` and `exact` options.
- `exact`—(Optional) Enforce the exact transmission rate or percentage you configure with the `transmit-rate` *`rate`* or `transmit-rate percent` statement. Under sustained congestion, a rate-controlled queue that goes into negative credit fills up and eventually drops packets. You specify the `exact` option as follows:

  ```
  [edit class-of-service schedulers scheduler-name]
  transmit-rate rate exact;

  [edit class-of-service schedulers scheduler-name]
  transmit-rate percent percentage exact;
  ```

  In the configuration, you cannot combine the `remainder` and `exact` options.
- `rate-limit`—(Optional) Limit the transmission rate to the specified amount. You can configure this option for all 8 queues of a logical interface (unit) and apply it to shaped or unshaped logical interfaces. If you configure a zero rate-limited transmit rate, all packets belonging to that queue are dropped.

NOTE: You can apply a transmit rate limit to logical interfaces on Multiservices 100, 400, or 500 PICs. Typically, rate limits are used to prevent a strict-high queue (such as voice) from starving lower priority queues. You can only rate-limit one queue per logical interface. To apply a rate-limit to a Multiservices PIC interface, configure the rate limit in a scheduler and apply the scheduler map to the Multiservices (`lsq-`) interface at the `[edit class-of-service interfaces]` hierarchy level. For information about configuring other scheduler components, see "Configuring Schedulers" on page 348.

Example: Configuring Scheduler Transmission Rate

Configure the `best-effort` scheduler to use the remainder of the bandwidth on any interface to which it is assigned:

```
class-of-service {
    schedulers {
        best-effort {
            transmit-rate remainder;
        }
```

```
    }
}
```

Allocation of Leftover Bandwidth

The allocation of leftover bandwidth is a complex topic. It is difficult to predict and to test, because the behavior of the software varies depending on the traffic mix.

If a queue receives offered loads in excess of the queue's bandwidth allocation, the queue has negative bandwidth credit, and receives a share of any available leftover bandwidth. Negative bandwidth credit means the queue has used up its allocated bandwidth. If a queue's bandwidth credit is positive, meaning it is not receiving offered loads in excess of its bandwidth configuration, then the queue does not receive a share of leftover bandwidth. If the credit is positive, then the queue does not need to use leftover bandwidth, because it can use its own allocation.

This use of leftover bandwidth is the default. If you do not want a queue to use any leftover bandwidth, you must configure it for strict allocation by including the `transmit-rate` statement with the `exact` option at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level. With rate control in place, the specified bandwidth is strictly observed.

By default, excess bandwidth is shared in the ratio of the transmit rates. You can adjust this distribution by configuring the *excess-rate* statement at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level. You can specify the excess rate sharing by percentage or by proportion.

In summary, Junos devices, share excess bandwidth in the ratio of the transmit rates, but you can adjust this distribution.

RELATED DOCUMENTATION

How Schedulers Define Output Queue Properties | **342**

excess-rate

schedulers

Providing a Guaranteed Minimum Rate

IN THIS SECTION

You can configure guaranteed bandwidth, also known as a committed information rate (CIR). This allows you to specify a guaranteed rate for each logical interface. The guaranteed rate is a minimum. If excess physical interface bandwidth is available for use, the logical interface receives more than the guaranteed rate provisioned for the interface.

You cannot provision the sum of the guaranteed rates to be more than the physical interface bandwidth, or the bundle bandwidth for LSQ interfaces. If the sum of the guaranteed rates exceeds the interface or bundle bandwidth, the commit operation does not fail, but the software automatically decreases the rates so that the sum of the guaranteed rates is equal to the available bundle bandwidth.

To configure a guaranteed minimum rate, perform the following steps:

1. Include the `guaranteed-rate` statement at the `[edit class-of-service traffic-control-profile profile-name]` hierarchy level:

   ```
   [edit class-of-service traffic-control-profiles profile-name]
   guaranteed-rate (percent percentage | rate) <burst-size bytes>;
   ```

   On LSQ interfaces, you can configure the guaranteed rate as a percentage from 1 through 100.

2. Optionally, you can base the delay-buffer calculation on a delay-buffer rate. To do this, include the `delay-buffer-rate` statement `[edit class-of-service traffic-control-profiles profile-name]` hierarchy level:

   ```
   [edit class-of-service traffic-control-profiles profile-name]
   delay-buffer-rate (percent percentage | rate);
   ```

   On LSQ interfaces, you can configure the delay-buffer rate as a percentage from 1 through 100.

   The actual delay buffer is based on the calculations described in "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476. For an example showing how the delay-buffer rates are applied, see "Example: Providing a Guaranteed Minimum Rate" on page 379.

If you do not include the `delay-buffer-rate` statement, the delay-buffer calculation is based on the guaranteed rate, the shaping rate if no guaranteed rate is configured, or the scaled shaping rate if the interface is oversubscribed.

If you do not specify a shaping rate or a guaranteed rate, the logical interface receives a minimal delay-buffer rate and minimal bandwidth equal to four MTU-sized packets.

You can configure a rate for the delay buffer that is higher than the guaranteed rate. This can be useful when the traffic flow might not require much bandwidth in general, but in some cases traffic can be bursty and therefore needs a large buffer.

Configuring large buffers on relatively slow-speed links can cause packet aging. To help prevent this problem, the software requires that the sum of the delay-buffer rates be less than or equal to the port speed. This restriction does not eliminate the possibility of packet aging, so you should be cautious when using the `delay-buffer-rate` statement. Though some amount of extra buffering might be desirable for burst absorption, delay-buffer rates should not far exceed the service rate of the logical interface.

If you configure delay-buffer rates so that the sum exceeds the port speed, the configured delay-buffer rate is not implemented for the last logical interface that you configure. Instead, that logical interface receives a delay-buffer rate of 0, and a warning message is displayed in the CLI. If bandwidth becomes available (because another logical interface is deleted or deactivated, or the port speed is increased), the configured delay-buffer-rate is reevaluated and implemented if possible.

If the guaranteed rate of a logical interface cannot be implemented, that logical interface receives a delay-buffer rate of 0, even if the configured delay-buffer rate is within the interface speed. If at a later time the guaranteed rate of the logical interface can be met, the configured delay-buffer rate is reevaluated and if the delay-buffer rate is within the remaining bandwidth, it is implemented.

If any logical interface has a configured guaranteed rate, all other logical interfaces on that port that do not have a guaranteed rate configured receive a delay-buffer rate of 0. This is because the absence of a guaranteed rate configuration corresponds to a guaranteed rate of 0 and, consequently, a delay-buffer rate of 0.

3. To assign a scheduler map to the logical interface, include the `scheduler-map` statement at the `[edit class-of-service traffic-control-profiles` *`profile-name`*`]` hierarchy level:

```
[edit class-of-service traffic-control-profiles profile-name]
scheduler-map map-name;
```

For information about configuring schedulers and scheduler maps, see "Configuring Schedulers" on page 348 and "Configuring Scheduler Maps" on page 348.

4. To enable large buffer sizes to be configured, include the `q-pic-large-buffer` statement at the `[edit chassis fpc slot-number pic pic-number]` hierarchy level:

   ```
   [edit chassis fpc slot-number pic pic-number]
   q-pic-large-buffer;
   ```

   If you do not include this statement, the delay-buffer size is more restricted. For more information, see "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476.

5. To enable scheduling on logical interfaces, include the `per-unit-scheduler` statement at the `[edit interfaces interface-name]` hierarchy level:

   ```
   [edit interfaces interface-name]
   per-unit-scheduler;
   ```

6. To apply the traffic-scheduling profile to the logical interface, include the output-traffic-control-profile statement at the `[edit class-of-service interfaces interface-name unit logical-unit-number]` hierarchy level:

   ```
   [edit class-of-service interfaces interface-name unit logical-unit-number]
   output-traffic-control-profile profile-name;
   ```

Table 30 on page 378 shows how the bandwidth and delay buffer are allocated in various configurations.

Table 30: Bandwidth and Delay Buffer Allocations by Configuration Scenario

Configuration Scenario	Delay Buffer Allocation
You do not configure a guaranteed rate. You do not configure a delay-buffer rate.	Logical interface receives minimal bandwidth with no guarantees and receives a minimal delay buffer equal to 4 MTU-sized packets.
You configure a guaranteed rate. You do not configure a delay-buffer rate.	Logical interface receives bandwidth equal to the guaranteed rate and a delay buffer based on the guaranteed rate. The multiplicative factor depends on whether you include the `q-pic-large-buffer` statement. For more information, see "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476.

Table 30: Bandwidth and Delay Buffer Allocations by Configuration Scenario *(Continued)*

Configuration Scenario	Delay Buffer Allocation
You configure a guaranteed rate. You configure a delay-buffer rate.	Logical interface receives bandwidth equal to the guaranteed rate and a delay buffer based on the delay-buffer rate. The multiplicative factor depends on whether you include the `q-pic-large-buffer` statement. For more information, see "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476.

Verifying Configuration of Guaranteed Minimum Rate

To verify your configuration, you can issue this following operational mode commands:

- `show class-of-service interfaces`
- `show class-of-service traffic-control-profile` *`profile-name`*

Example: Providing a Guaranteed Minimum Rate

Two logical interface units, `0` and `1`, are provisioned with a guaranteed minimum of 750 Kbps and 500 Kbps, respectively. For logical unit `1`, the delay buffer is based on the guaranteed rate setting. For logical unit `0`, a delay-buffer rate of 500 Kbps is specified. The actual delay buffers allocated to each logical interface are 2 seconds of 500 Kbps. The 2-second value is based on the following calculation:

```
delay-buffer-rate < [8 x 64 Kbps]): 2 seconds of delay-buffer-rate
```

For more information about this calculation, see "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476.

```
chassis {
    fpc 3 {
        pic 0 {
            q-pic-large-buffer;
        }
    }
}
interfaces {
    ge-3/0/1 {
        per-unit-scheduler;
    }
}
class-of-service {
    traffic-control-profiles {
        tc-profile3 {
            guaranteed-rate 750k;
            scheduler-map sched-map3;
            delay-buffer-rate 500k; # 500 Kbps is less than 8 x 64
Kbps
        }
        tc-profile4 {
            guaranteed-rate 500k; # 500 Kbps is less than 8 x 64
Kbps
            scheduler-map sched-map4;
        }
    }
    interfaces {
        ge-3/0/1 {
        unit 0 {
            output-traffic-control-profile tc-profile3;
        }
        unit 1 {
            output-traffic-control-profile tc-profile4;
        }
    }
}
```

PIR-Only and CIR Mode

The actual behavior of many CoS parameters, especially the shaping rate and guaranteed rate, depend on whether the physical interface is operating in PIR-only or CIR mode.

In PIR-only mode, one or more nodes perform shaping. The physical interface is in the PIR-only mode if no child (or grandchild) node under the port has a guaranteed rate configured.

The mode of the port is important because in PIR-only mode, the scheduling across the child nodes is in proportion to their shaping rates (PIRs) and not the guaranteed rates (CIRs). This can be important if the observed behavior is not what is anticipated.

In CIR mode, one or more nodes applies a guaranteed rate and might perform shaping. A physical interface is in CIR mode if at least one child (or grandchild) node has a guaranteed rate configured.

In CIR mode, one or more nodes applies the guaranteed rates. In addition, any child or grandchild node under the physical interface can have a shaping rate configured. Only the guaranteed rate matters. In CIR mode, nodes that do not have a guaranteed rate configured are assumed to have a very small guaranteed rate (queuing weight).

NOTE: For nodes that do not have a guaranteed rate configured, Juniper recommends that you configure a delay buffer rate with the same value as the shaping rate.

Excess Rate and Excess Priority Configuration Examples

To configure the excess rate for nonqueuing Packet Forwarding Engines, include the *excess-rate* statement at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level.

To configure the excess priority for nonqueuing Packet Forwarding Engines, include the *excess-priority* statement at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level.

The relationship between the configured guaranteed rate, excess rate, guaranteed priority, excess priority, and offered load is not always obvious. The following tables show the expected throughput of a Gigabit Ethernet port with various bandwidth-sharing parameters configured on the queues.

The default behavior of a nonqueuing Gigabit Ethernet interface with multiple priority levels is shown in Table 31 on page 381. All queues in the table get their guaranteed rate. The excess bandwidth is first offered to the excess high-priority queues. Because these use all available bandwidth, these is no remaining excess bandwidth for the low-priority queues.

Table 31: Current Behavior with Multiple Priority Levels

Queue	Guaranteed (Transmit) Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q0	20%	high	high	600 Mbps	200 + 366.67 = 566.67 Mbps
Q1	10%	high	high	500 Mbps	100 + 183.33 = 283.33 Mbps
Q2	10%	low	low	500 Mbps	100 + 0 = 100 Mbps
Q3	5%	low	low	500 Mbps	50 + 0 = 50 Mbps

The default behavior of a nonqueuing Gigabit Ethernet interface with the same priority levels is shown in Table 32 on page 382. All queues in the table get their guaranteed rate. Because all queues have the same excess priority, they share the excess bandwidth and each queue gets excess bandwidth in proportion to the transmit rate.

Table 32: Current Behavior with Same Priority Levels

Queue	Guaranteed (Transmit) Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q0	20%	high	high	500 Mbps	200 + 244.44 = 444.44 Mbps
Q1	10%	high	high	500 Mbps	100 + 122.22 = 222.22 Mbps
Q2	10%	high	high	500 Mbps	100 + 122.22 = 222.22 Mbps
Q3	5%	high	high	500 Mbps	50 + 61.11= 111.11 Mbps

The default behavior of a nonqueuing Gigabit Ethernet interface with the at least one strict-high priority level is shown in Table 33 on page 382. First the high priority and strict-high are serviced in a weighted round-robin fashion. The high priority queue gets its guaranteed bandwidth and the strict-high queue gets what remains. The high excess priority queue gets all the excess bandwidth.

Table 33: Current Behavior with Strict-High Priority

Queue	Guaranteed (Transmit) Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q0	20%	strict-high	X	500 Mbps	500 Mbps
Q1	10%	high	high	500 Mbps	100 + 250 = 350 Mbps
Q2	10%	low	low	500 Mbps	100 + 0 = 100 Mbps
Q3	5%	low	low	500 Mbps	50 + 0= 50 Mbps

The default behavior of a nonqueuing Gigabit Ethernet interface with the at least one strict-high priority level and a higher offered load on Q0 is shown in Table 34 on page 383. First the high priority and strict-high are serviced in a weighted round-robin fashion. The high priority queue gets its guaranteed bandwidth and the strict-high queue gets what remains. (The high priority queue receives its guaranteed bandwidth unless a strict-high queue is configured, which in certain conditions might starve the high priority queue. To guarantee the configured transmit rate on high-priority queues, apply the `rate-limit` option to the transmit rate of the strict-high priority queue.) There is no excess bandwidth.

Table 34: Strict-High Priority with Higher Load

Queue	Guaranteed (Transmit) Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q0	20%	strict-high	X	1 Gbps	900 Mbps
Q1	10%	high	high	500 Mbps	100 + 0 = 100 Mbps
Q2	10%	low	low	500 Mbps	0 + 0 = 0 Mbps
Q3	5%	low	low	500 Mbps	0 + 0= 0 Mbps

Now consider the behavior of the queues with configured excess rates and excess priorities.

The behavior with multiple priority levels is shown in Table 35 on page 383. All queues get the guaranteed rate. The excess bandwidth is first offered to the excess high priority queues and these consume all the bandwidth. There is no remaining excess bandwidth for low priority queues.

Table 35: Sharing with Multiple Priority Levels

Queue	Guaranteed (Transmit) Rate	Excess Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q0	20%	10%	high	high	500 Mbps	200 + 275 = 475 Mbps
Q1	10%	20%	high	low	500 Mbps	100 + 0 = 100 Mbps
Q2	10%	10%	low	high	500 Mbps	100 + 275 = 275 Mbps

Table 35: Sharing with Multiple Priority Levels *(Continued)*

Queue	Guaranteed (Transmit) Rate	Excess Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q3	5%	20%	low	low	500 Mbps	50 + 0= 50 Mbps

The behavior with the same (high) priority levels is shown in Table 36 on page 384. All queues get the guaranteed rate. Because all queues have the same excess priority, they share the excess bandwidth in proportion to their transmit rate.

Table 36: Sharing with the Same Priority Levels

Queue	Guaranteed (Transmit) Rate	Excess Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q0	20%	10%	high	high	500 Mbps	200 + 91.67 = 291.67 Mbps
Q1	10%	20%	high	high	500 Mbps	100 + 183.33 = 283.33 Mbps
Q2	10%	10%	high	high	500 Mbps	100 + 91.67 = 191.67 Mbps
Q3	5%	20%	high	high	500 Mbps	50 + 183.33 = 233.33 Mbps

The behavior with at least one strict-high priority level is shown in Table 37 on page 384. The high priority and strict-high queues are serviced in a weighted round-robin fashion. The high priority queue gets its guaranteed rate and the strict-high queue gets the rest. The excess high-priority queue get all the excess bandwidth.

Table 37: Sharing with at Least One Strict-High Priority

Queue	Guaranteed (Transmit) Rate	Excess Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q0	20%	X	strict-high	X	500 Mbps	500 Mbps

Table 37: Sharing with at Least One Strict-High Priority *(Continued)*

Queue	Guaranteed (Transmit) Rate	Excess Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q1	10%	20%	high	low	500 Mbps	100 + 0 = 100 Mbps
Q2	10%	10%	low	high	500 Mbps	100 + 250 = 350 Mbps
Q3	5%	20%	low	low	500 Mbps	50 + 0 = 50 Mbps

The behavior with at least one strict-high priority level and a higher offered load is shown in Table 38 on page 385. The high priority and strict-high queues are serviced in a weighted round-robin fashion. The high priority queue gets its guaranteed rate and the strict-high queue gets the rest. There is no excess bandwidth.

Table 38: Sharing with at Least One Strict-High Priority and Higher Load

Queue	Guaranteed (Transmit) Rate	Excess Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q0	20%	X	strict-high	X	900 Mbps	900 Mbps
Q1	10%	20%	high	low	500 Mbps	100 + 0 = 100 Mbps
Q2	10%	10%	low	high	500 Mbps	0 + 0 = 0 Mbps
Q3	5%	20%	low	low	500 Mbps	0 + 0 = 0 Mbps

The behavior with at least one strict-high priority level and a rate limit is shown in Table 39 on page 386. Queue 0 and Queue 2 are rate limited, so the maximum bandwidth they are offered is the transmit bandwidth and they will not be offered any excess bandwidth. All other queues are offered the guaranteed bandwidth and the excess is shared by the non-rate-limited queues.

Table 39: Sharing with at Least One Strict-High Priority and Rate Limit

Queue	Guaranteed (Transmit) Rate	Rate Limit	Excess Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
Q0	20%	Yes	X	strict-high	X	500 Mbps	200 + 0 = 200 Mbps
Q1	10%	No	20%	high	low	500 Mbps	100 + 275 = 375 Mbps
Q2	10%	Yes	10%	low	high	500 Mbps	100 + 0 = 100 Mbps
Q3	5%	No	20%	low	low	500 Mbps	50 + 275 = 325 Mbps

Configuring the Schedulers

The following example configures schedulers, forwarding classes, and a scheduler map for an interface with excess rates and excess priorities.

```
[edit class-of-service schedulers]
scheduler-1 {
    transmit-rate percent 20;
    priority high;
    excess-rate percent 10;
    excess-priority low;
}
scheduler-2 {
    transmit-rate percent 10;
    priority strict-high;
}
scheduler-3 {
    transmit-rate percent 10;
    priority medium-high;
    excess-rate percent 20;
    excess-priority high;
}
```

```
scheduler-4 {
    transmit-rate percent 5;
    priority medium-high;
    excess-rate percent 30;
    excess-priority low;
}
```

Configuring the Forwarding Classes

```
[edit class-of-service]
forwarding-classes {
    class cp_000 queue-num 0;
    class cp_001 queue-num 1;
    class cp_010 queue-num 2;
    class cp_011 queue-num 3;
    class cp_100 queue-num 4;
    class cp_101 queue-num 5;
    class cp_110 queue-num 6;
    class cp_111 queue-num 7;
}
```

Configuring the Scheduler Map

```
[edit class-of-service scheduler-maps]
scheduler-map-1 {
    forwarding-class cp_000 scheduler scheduler-1;
    forwarding-class cp_001 scheduler scheduler-2;
    forwarding-class cp_010 scheduler scheduler-3;
    forwarding-class cp_011 scheduler scheduler-4;
}
```

Applying the Scheduler Map to the Interface

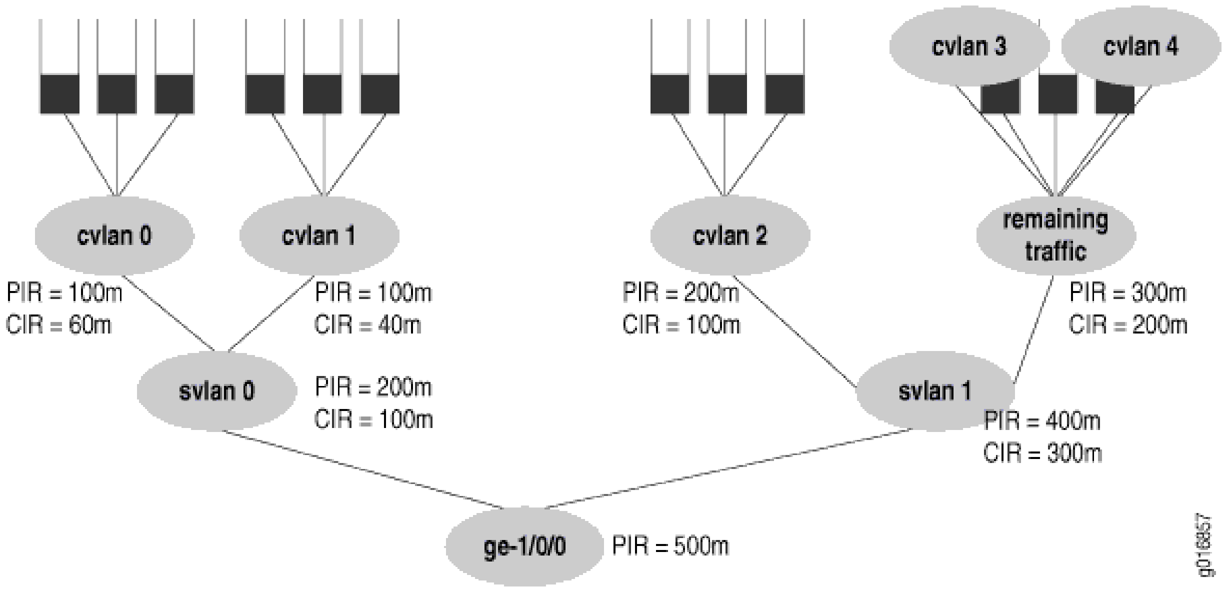
```
[edit interfaces]
ge-1/1/0 {
    scheduler-map scheduler-map-1;
    unit 0 {
        family inet {
            address 192.168.1.2/32;
        }
    }
}
```

Controlling Remaining Traffic

You can configure many logical interfaces under an interface. However, only a subset of them might have a traffic control profile attached. For example, you can configure three logical interfaces (units) over the same service VLAN, but apply a traffic control profile specifying best-effort and voice queues to only one of the logical interface units. Traffic from the two remaining logical interfaces is considered *remaining traffic*. To configure transmit rate guarantees for the remaining traffic, you configure the `output-traffic-control-profile-remaining` statement specifying a guaranteed rate for the remaining traffic. Without this statement, the remaining traffic gets a default, minimal bandwidth. In the same way, the `shaping-rate` and `delay-buffer-rate` statements can be specified in the traffic control profile referenced with the `output-traffic-control-profile-remaining` statement in order to shape and provide buffering for remaining traffic.

Consider the interface shown in Figure 33 on page 388. Customer VLANs 3 and 4 have no explicit traffic control profile. However, the service provider might want to establish a shaping and guaranteed transmit rate for aggregate traffic heading for those customer VLANs. The solution in to configure and apply a traffic control profile for all remaining traffic on the interface.

Figure 33: Handling Remaining Traffic



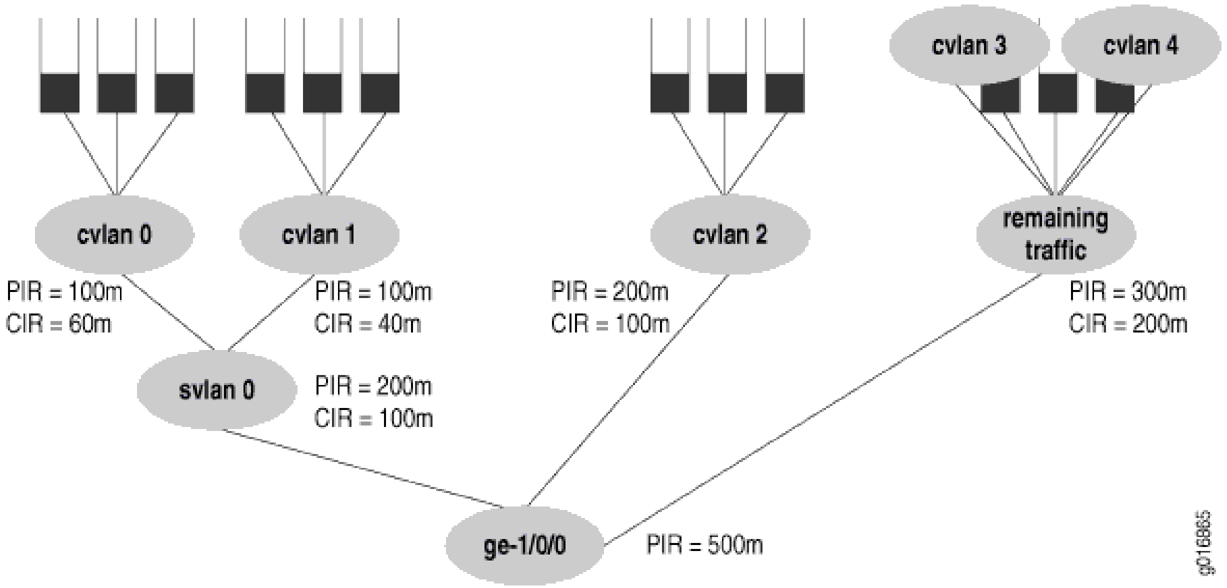
This example considers the case where customer VLANs 3 and 4 have no explicit traffic control profile, yet need to establish a shaping and guaranteed transmit rate for traffic heading for those customer VLANs. The solution is to add a traffic control profile to the `svlan1` interface set. This example builds on the earlier example and so does not repeat all configuration details, only those at the service VLAN level.

```
[edit class-of-service interfaces]
interface-set svlan0 {
    output-traffic-control-profile tcp-svlan0;
}
interface-set svlan1 {
    output-traffic-control-profile tcp-svlan1; # For explicitly shaped traffic.
    output-traffic-control-profile-remaining tcp-svlan1-remaining; # For all remaining traffic.
}

[edit class-of-service traffic-control-profiles]
tcp-svlan1 {
    shaping-rate 400m;
    guaranteed-rate 300m;
}
tcp-svlan1-remaining {
    shaping-rate 300m;
    guaranteed-rate 200m;
    scheduler-map smap-remainder; # this smap is not shown in detail
}
```

Next, consider the example shown in Figure 34 on page 390.

Figure 34: Another Example of Handling Remaining Traffic



In this example, `ge-1/0/0` has three logical interfaces (unit 1, unit 2, and unit 3), and SVLAN 2000, which are covered by the interface set:

- Scheduling for the interface set `svlan0` is specified by referencing an `output-traffic-control-profile` statement which specifies the `guaranteed-rate`, `shaping-rate`, and `delay-buffer-rate` statement values for the interface set. In this example, the output traffic control profile called `tcp-svlan0` guarantees 100 Mbps and shapes the interface set `svlan0` to 200 Mbps.

- Scheduling and queuing for remaining traffic of `svlan0` is specified by referencing an `output-traffic-control-profile-remaining` statement which references a `scheduler-map` statement that establishes queues for the remaining traffic. The specified traffic control profile can also configure guaranteed, shaping, and delay-buffer rates for the remaining traffic. In this example, `output-traffic-control-profile-remaining tcp-svlan0-rem` references `scheduler-map smap-svlan0-rem`, which calls for a best-effort queue for remaining traffic (that is, traffic on unit 3 and unit 4, which is not classified by the `svlan0` interface set). The example also specifies a `guaranteed-rate` of 200 Mbps and a `shaping-rate` of 300 Mbps for all remaining traffic.

- Scheduling and queuing for logical interface `ge-1/0/0 unit 1` is configured “traditionally” and uses an `output-traffic-control-profile` specified for that unit. In this example, `output-traffic-control-profile tcp-ifl1` specifies scheduling and queuing for `ge-1/0/0 unit 1`.

This example does not include the `[edit interfaces]` configuration.

```
[edit class-of-service interfaces]
interface-set {
    svlan0 {
        output-traffic-control-profile tcp-svlan0; # Guarantee & shaper for svlan0.
    }
}
ge-1/0/0 {
    output-traffic-control-profile-remaining tcp-svlan0-rem;
    # Unit 3 and 4 are not explicitly configured, but captured by “remaining”
    unit 1 {
        output-traffic-control-profile tcp-ifl1; # Unit 1 be & ef queues.
    }
}
```

Here is how the traffic control profiles for this example are configured:

```
[edit class-of-service traffic-control-profiles]
tcp-svlan0 {
    shaping-rate 200m;
    guaranteed-rate 100m;
}
tcp-svlan0-rem {
    shaping-rate 300m;
    guaranteed-rate 200m;
    scheduler-map smap-svlan0-rem; # This specifies queues for remaining traffic
}
tcp-ifl1 {
    scheduler-map smap-ifl1;
}
```

Finally, here are the scheduler maps and queues for the example:

```
[edit class-of-service scheduler-maps]
smap-svlan0-rem {
    forwarding-class best-effort scheduler sched-foo;
}
smap-ifl1 {
    forwarding-class best-effort scheduler sched-bar;
    forwarding-class assured-forwarding scheduler sched-baz;
}
```

The configuration for the referenced schedulers are not given for this example.

Bandwidth Sharing on Nonqueuing Packet Forwarding Engines Overview

You can configure bandwidth sharing rate limits, excess rate, and excess priority at the queue level on the following Juniper Networks routers and switches:

- EX Series switches
- MX Series 5G Universal Routing Platform with nonqueuing DPCs (rate limit, excess rate, and excess priority)

You configure rate limits when you have a concern that low-latency packets (such as high or strict-high priority packets for voice) might starve low-priority and medium-priority packets. In Junos OS, the low latency queue is implemented by rate-limiting packets to the transmit bandwidth. The rate-limiting is performed immediately before queuing the packet for transmission. All packets that exceed the rate limit are not queued, but dropped.

By default, if the excess priority is not configured for a queue, the excess priority will be the same as the normal queue priority. If none of the queues have an excess rate configured, then the excess rate will be the same as the transmit rate percentage. If at least one of the queues has an excess rate configured, then the excess rate for the queues that do not have an excess rate configured will be set to zero.

When the physical interface is on queuing hardware, these features are dependent on the PIC (or queuing DPC in the case of the MX Series router) configuration.

You cannot configure both rate limits and buffer sizes on these Packet Forwarding Engines.

Four levels of excess priorities are supported: low, medium-low, medium-high, and high.

NOTE: Rate limiting is implemented differently on Enhanced Queuing DPCs and non-queuing Packet Forwarding Engines. On Enhanced Queuing DPCs, rate-limiting is implemented using a single rate two-color policer. On non-queuing Packet Forwarding Engines, rate-limiting is achieved by shaping the queue to the transmit rate and keeping the queue delay buffers small to prevent too many packets from being queued once the shaping rate is reached.

Configuring Rate Limits on Nonqueuing Packet Forwarding Engines

On non-queuing Packet Forwarding Engines, rate-limiting is achieved by shaping the queue to the transmit rate and keeping the queue delay buffers small to prevent too many packets from being queued once the shaping rate is reached. To configure rate limits for non-queuing Packet Forwarding Engines, include the `transmit-rate` statement at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level.

NOTE: Rate limiting is implemented differently on MPCs and Enhanced Queuing DPCs than on non-queuing Packet Forwarding Engines. On MPCs and Enhanced Queuing DPCs, rate-limiting is implemented using a single-rate two-color policer. See "Example: Limiting Outbound Traffic Within Your Network by Configuring an Egress Single-Rate Two-Color Policer and Configuring Multifield Classifiers" on page 225 for an example of configuring a single-rate two-color policer to rate limit traffic.

Configuring the Schedulers

The following example configures schedulers, forwarding classes, and a scheduler map for a rate-limited interface.

```
[edit class-of-service schedulers]
scheduler-1 {
    transmit-rate percent 20 rate-limit;
    priority high;
}
scheduler-2 {
    transmit-rate percent 10 rate-limit;
    priority strict-high;
}
scheduler-3 {
    transmit-rate percent 40;
    priority medium-high;
}
scheduler-4 {
    transmit-rate percent 30;
    priority medium-high;
}
```

Configuring the Forwarding Classes

```
[edit class-of-service]
forwarding-classes {
    class cp_000 queue-num 0;
    class cp_001 queue-num 1;
    class cp_010 queue-num 2;
    class cp_011 queue-num 3;
    class cp_100 queue-num 4;
    class cp_101 queue-num 5;
    class cp_110 queue-num 6;
    class cp_111 queue-num 7;
}
```

Configuring the Scheduler Map

```
[edit class-of-service scheduler-maps]
scheduler-map-1 {
    forwarding-class cp_000 scheduler scheduler-1;
    forwarding-class cp_001 scheduler scheduler-2;
    forwarding-class cp_010 scheduler scheduler-3;
    forwarding-class cp_011 scheduler scheduler-4;
}
```

Applying the Scheduler Map to the Interface

```
[edit class-of-service interfaces]
ge-1/0/0 {
    scheduler-map scheduler-map-1;
}
```

Applying Scheduler Maps and Shaping Rate to Logical Interfaces and VLANs

By default, output scheduling is not enabled on logical interfaces. Logical interfaces without shaping configured share a default scheduler. This scheduler has a committed information rate (CIR) that equals 0. (The CIR is the guaranteed rate.) The default scheduler has a peak information rate (PIR) that equals the physical interface shaping rate.

NOTE: If you apply a shaping rate, you must keep in mind that the transit statistics for physical interfaces are obtained from the packet forwarding engine, but the traffic statistics are supplied by the PIC. Therefore, if shaping is applied to the PIC, the count of packets in the transit statistics fields do not always agree with the counts in the traffic statistics. For example, the IPv6 transit statistics will not necessarily match the traffic statistics on the interface. However, at the logical interface level, both transit and traffic statistics are obtained from the Packet Forwarding Engine and will not show any difference.

Logical interface scheduling (also called *per-unit scheduling*) allows you to enable multiple output queues on a logical interface and associate customized output scheduling and shaping for each queue.

NOTE: Ingress scheduling does not support logical interface scheduling.

For supported PICs, you can configure a shaping rate for a VLAN or logical interface and oversubscribe the physical interface by including the `shaping-rate` statement at the `[edit class-of-service traffic-control-profiles]` hierarchy level. With this configuration approach, you can independently control the delay-buffer rate, as described in "Oversubscribing Interface Bandwidth" on page 366.

Physical interfaces (for example, `xe-0/0/0`, `xe-0/0/0:0`, and `ge-0/0/0`) support scheduling with any encapsulation type pertinent to that physical interface. For a single port, you cannot apply scheduling to the physical interface if you apply scheduling to one or more of the associated logical interfaces.

NOTE: In the Junos OS implementation, the term *logical interfaces* generally refers to interfaces you configure by including the `unit` statement at the `[edit interfaces *interface-name*]` hierarchy level. As such, logical interfaces have the *`logical`* descriptor at the end of the interface name, as in `xe-0/0/0.1` or `xe-0/0/0:0.1`, where the logical unit number is `1`.

Within the `[edit class-of-service]` hierarchy level, you cannot use the `.`*`logical`* descriptor when you assign properties to logical interfaces. Instead, you must include the `unit` statement in the configuration. For example:

```
[edit class-of-service]
user@host# set interfaces xe-0/0/0 unit 0 scheduler-map map1
```

Table 40 on page 395 shows the MICs and MPCs that support fine-grained queuing and scheduling.

Table 40: Fine-Grained Queuing and Scheduling Support by MIC or MPC Type

MPC	MIC	Supported	Example Configuration
Fixed Configuration MPCs			

Table 40: Fine-Grained Queuing and Scheduling Support by MIC or MPC Type *(Continued)*

MPC	MIC	Supported	Example Configuration
16x10GE MPC	No	Yes	`[edit class-of-service interfaces ge-0/0/0 unit 1]` `scheduler-map map-1;`
32x10GE MPC4E	No	Yes	`[edit class-of-service interfaces ge-0/0/0 unit 1]` `scheduler-map map-1;`
2x100GE + 8x10GE MPC4E	No	Yes	`[edit class-of-service interfaces ge-0/0/0 unit 1]` `scheduler-map map-1;`
6x40GE + 24x10GE MPC5E	No	No	No
6x40GE + 24x10GE MPC5EQ	No	Yes	`[edit class-of-service interfaces ge-0/0/0 unit 1]` `scheduler-map map-1;`
2x100GE + 4x10GE MPC5E	No	No	No
2x100GE + 4x10GE MPC5EQ	No	Yes	`[edit class-of-service interfaces ge-0/0/0 unit 1]` `scheduler-map map-1;`
MPCs			
MPC1	No	No	No
MPC1E	No	No	No

Table 40: Fine-Grained Queuing and Scheduling Support by MIC or MPC Type *(Continued)*

MPC	MIC	Supported	Example Configuration
MPC1 Q	Any supported MIC	Yes	Example of supported configuration: `[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;`
MPC1E Q	Any supported MIC	Yes	Example of supported configuration: `[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;`
MPC2	No	No	No
MPC2E	No	No	No
MPC2 Q	Any supported MIC	Yes	Example of supported configuration: `[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;`
MPC2E Q	Any supported MIC	Yes	Example of supported configuration: `[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;`
MPC2 EQ	Any supported MIC	Yes	Example of supported configuration: `[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;`

Table 40: Fine-Grained Queuing and Scheduling Support by MIC or MPC Type *(Continued)*

MPC	MIC	Supported	Example Configuration
MPC2E EQ	Any supported MIC	Yes	Example of supported configuration: `[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;`
MPC2E P	No	No	No
MPC3E	10-Gigabit Ethernet MIC with SFP+	Yes	Example of supported configuration: `[edit class-of-service interfaces xe-0/0/0 unit 1] scheduler-map map-1;`
	40-Gigabit Ethernet MIC with QSFP+	Yes	Example of supported configuration: `[edit class-of-service interfaces et-0/0/0 unit 1] scheduler-map map-1;`
	100-Gigabit Ethernet MIC with CXP	Yes	Example of supported configuration: `[edit class-of-service interfaces et-0/0/0 unit 1] scheduler-map map-1;`
MPC6E	Any supported MIC	Yes	Example of supported configuration: `[edit class-of-service interfaces et-0/0/0 unit 1] scheduler-map map-1;`

To configure scheduling on logical interfaces:

1. Enable per-unit scheduling on the interface by including the `per-unit-scheduler` statement at the `[edit interfaces` *`interface-name`*`]` hierarchy level:

   ```
   [edit interfaces interface-name]
   per-unit-scheduler;
   ```

   When including the `per-unit-scheduler` statement, you must also include the `vlan-tagging` statement or the `flexible-vlan-tagging` statement (to apply scheduling to VLANs) or the `encapsulation frame-relay` statement (to apply scheduling to DLCIs) at the `[edit interfaces` *`interface-name`*`]` hierarchy level.

   When you include this statement, the maximum number of VLANs supported is 768 on a single-port Gigabit Ethernet IQ PIC. On a dual-port Gigabit Ethernet IQ PIC, the maximum number is 384.

   See "Scaling of Per-VLAN Queuing on Non-Queuing MPCs" on page 945 for scaling information on non-queuing MPCs.

2. Associate a scheduler with the interface by including the `scheduler-map` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level:

   ```
   [edit class-of-service interfaces interface-name unit logical-unit-number]
   scheduler-map map-name;
   ```

   Alternatively, associate a scheduler with the interface by including the `scheduler-map` statement at the `[edit class-of-service traffic-control-profiles` *`traffic control profile name`*`]` hierarchy level and then include the `output-traffic-control-profile` statement at the `[edit class-of-service interfaces` *`interface name`* `unit` *`logical unit number`*`]` hierarchy level.

   ```
   [edit class-of-service traffic-control-profiles traffic control profile name]
   scheduler-map map-name;
   ```

   ```
   [edit class-of-service interfaces interface-name unit logical-unit-number]
   output-traffic-control-profile traffic-control-profile-name;
   ```

3. Configure shaping on the interface by including the `shaping-rate` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level:

   ```
   [edit class-of-service interfaces interface-name unit logical-unit-number]
   shaping-rate rate;
   ```

NOTE: You can also apply the shaping rate to the traffic control profile.

By default, the logical interface bandwidth is the average of unused bandwidth for the number of logical interfaces that require default bandwidth treatment. You can specify a peak bandwidth rate in bps, either as a complete decimal number or as a decimal number followed by the abbreviation `k` (1000), `m` (1,000,000), or `g` (1,000,000,000).

RELATED DOCUMENTATION

per-unit-scheduler

Example: Applying Scheduling and Shaping to VLANs | 400

Example: Applying Scheduling and Shaping to VLANs

IN THIS SECTION

This example shows how to apply schedulers to individual logical interfaces.

Requirements

This example uses the following hardware and software components:

- MX Series routers
- Any supported Junos release

Overview

By default, output scheduling is not enabled on logical interfaces. Logical interfaces without shaping configured share a default scheduler. *Logical interface scheduling* (also called *per-unit scheduling*) allows you to enable multiple output queues on a logical interface and associate customized scheduling and shaping for each queue.

To enable per-unit scheduling, include the `per-unit-scheduler` statement at the `[edit interfaces` *`interface name`*`]` hierarchy level. When per-unit schedulers are enabled, you can define dedicated schedulers for logical interfaces by including the `scheduler-map` statement at the `[edit class-of-service interfaces` *`interface name`* `unit` *`logical unit number`*`]` hierarchy level. Alternatively, you can include the `scheduler-map` statement at the `[edit class-of-service traffic-control-profiles` *`traffic control profile name`*`]` hierarchy level and then include the `output-traffic-control-profile` statement at the `[edit class-of-service interfaces` *`interface name`* `unit` *`logical unit number`*`]` hierarchy level.

This example shows how to define schedulers for logical interfaces through the use of traffic control profiles.

Configuration

IN THIS SECTION

CLI Quick Configuration

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

```
set interfaces xe-9/0/3 per-unit-scheduler
set interfaces xe-9/0/3 vlan-tagging
set interfaces xe-9/0/3 unit 1 vlan-id 101
set interfaces xe-9/0/3 unit 1 family inet address 10.1.1.1/24
set interfaces xe-9/0/3 unit 2 vlan-id 102
set interfaces xe-9/0/3 unit 2 family inet address 10.2.1.1/24
set class-of-service classifiers inet-precedence c8 forwarding-class be loss-priority low code-
points 000
set class-of-service classifiers inet-precedence c8 forwarding-class ef loss-priority low code-
points 001
set class-of-service classifiers inet-precedence c8 forwarding-class af loss-priority low code-
points 010
set class-of-service classifiers inet-precedence c8 forwarding-class nc loss-priority low code-
points 011
```

```
set class-of-service classifiers inet-precedence c8 forwarding-class be1 loss-priority low code-
points 100
set class-of-service classifiers inet-precedence c8 forwarding-class ef1 loss-priority low code-
points 101
set class-of-service classifiers inet-precedence c8 forwarding-class af1 loss-priority low code-
points 110
set class-of-service classifiers inet-precedence c8 forwarding-class nc1 loss-priority low code-
points 111
set class-of-service forwarding-classes queue 0 be
set class-of-service forwarding-classes queue 1 ef
set class-of-service forwarding-classes queue 2 af
set class-of-service forwarding-classes queue 3 nc
set class-of-service forwarding-classes queue 4 be1
set class-of-service forwarding-classes queue 5 ef1
set class-of-service forwarding-classes queue 6 af1
set class-of-service forwarding-classes queue 7 nc1
set class-of-service traffic-control-profiles tcp_ifd shaping-rate 2500000000
set class-of-service traffic-control-profiles tcp_ifd overhead-accounting bytes -20
set class-of-service traffic-control-profiles tcp_gold scheduler-map gold
set class-of-service traffic-control-profiles tcp_gold shaping-rate 2500000000
set class-of-service traffic-control-profiles tcp_gold overhead-accounting bytes -20
set class-of-service traffic-control-profiles tcp_gold guaranteed-rate 1g
set class-of-service traffic-control-profiles tcp_silver scheduler-map silver
set class-of-service traffic-control-profiles tcp_silver shaping-rate 1g
set class-of-service traffic-control-profiles tcp_silver overhead-accounting bytes -20
set class-of-service traffic-control-profiles tcp_silver guaranteed-rate 500m
set class-of-service interfaces xe-9/0/3 output-traffic-control-profile tcp_ifd
set class-of-service interfaces xe-9/0/3 unit 1 output-traffic-control-profile tcp_gold
set class-of-service interfaces xe-9/0/3 unit 2 output-traffic-control-profile tcp_silver
set class-of-service scheduler-maps gold forwarding-class be1 scheduler gold_internet
set class-of-service scheduler-maps gold forwarding-class ef1 scheduler gold_video
set class-of-service scheduler-maps gold forwarding-class af1 scheduler gold_voice
set class-of-service scheduler-maps gold forwarding-class nc1 scheduler gold_reserved
set class-of-service scheduler-maps silver forwarding-class be scheduler silver_internet
set class-of-service scheduler-maps silver forwarding-class ef scheduler silver_video
set class-of-service scheduler-maps silver forwarding-class af scheduler silver_voice
set class-of-service scheduler-maps silver forwarding-class nc scheduler silver_reserved
set class-of-service schedulers gold_internet excess-rate percent 40
set class-of-service schedulers gold_internet buffer-size percent 20
set class-of-service schedulers gold_internet priority low
set class-of-service schedulers gold_video transmit-rate percent 50
set class-of-service schedulers gold_video buffer-size percent 50
set class-of-service schedulers gold_voice shaping-rate percent 10
set class-of-service schedulers gold_voice buffer-size percent 10
set class-of-service schedulers gold_voice priority strict-high
set class-of-service schedulers gold_reserved excess-rate percent 20
set class-of-service schedulers gold_reserved buffer-size percent 10
set class-of-service schedulers gold_reserved priority low
set class-of-service schedulers silver_internet excess-rate percent 40
set class-of-service schedulers silver_internet buffer-size percent 20
set class-of-service schedulers silver_internet priority low
set class-of-service schedulers silver_video transmit-rate percent 50
set class-of-service schedulers silver_video buffer-size percent 50
set class-of-service schedulers silver_voice shaping-rate percent 10
set class-of-service schedulers silver_voice buffer-size percent 10
set class-of-service schedulers silver_voice priority strict-high
set class-of-service schedulers silver_reserved excess-rate percent 20
set class-of-service schedulers silver_reserved buffer-size percent 10
set class-of-service schedulers silver_reserved priority low
```

Procedure

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see the Junos OS CLI User Guide.

1. Configure the device interfaces.

```
[edit interfaces]
user@PE1# set xe-9/0/3 per-unit-scheduler
user@PE1# set xe-9/0/3 vlan-tagging
user@PE1# set xe-9/0/3 unit 1 vlan-id 101
user@PE1# set xe-9/0/3 unit 1 family inet address 10.1.1.1/24
user@PE1# set xe-9/0/3 unit 2 vlan-id 102
user@PE1# set xe-9/0/3 unit 2 family inet address 10.2.1.1/24
```

2. Configure the classifiers.

```
[edit class-of-service]
user@PE1# set classifiers inet-precedence c8 forwarding-class be loss-priority low code-
points 000
user@PE1# set classifiers inet-precedence c8 forwarding-class ef loss-priority low code-
```

```
points 001
user@PE1# set classifiers inet-precedence c8 forwarding-class af loss-priority low code-
points 010
user@PE1# set classifiers inet-precedence c8 forwarding-class nc loss-priority low code-
points 011
user@PE1# set classifiers inet-precedence c8 forwarding-class be1 loss-priority low code-
points 100
user@PE1# set classifiers inet-precedence c8 forwarding-class ef1 loss-priority low code-
points 101
user@PE1# set classifiers inet-precedence c8 forwarding-class af1 loss-priority low code-
points 110
user@PE1# set classifiers inet-precedence c8 forwarding-class nc1 loss-priority low code-
points 111
```

3. Configure the forwarding classes.

```
[edit class-of-service]
user@PE1# set forwarding-classes queue 0 be
user@PE1# set forwarding-classes queue 1 ef
user@PE1# set forwarding-classes queue 2 af
user@PE1# set forwarding-classes queue 3 nc
user@PE1# set forwarding-classes queue 4 be1
user@PE1# set forwarding-classes queue 5 ef1
user@PE1# set forwarding-classes queue 6 af1
user@PE1# set forwarding-classes queue 7 nc1
```

4. Configure the traffic control profiles.

```
[edit class-of-service]
user@PE1# set traffic-control-profiles tcp_ifd shaping-rate 2500000000
user@PE1# set traffic-control-profiles tcp_ifd overhead-accounting bytes -20
user@PE1# set traffic-control-profiles tcp_gold scheduler-map gold
user@PE1# set traffic-control-profiles tcp_gold shaping-rate 2500000000
user@PE1# set traffic-control-profiles tcp_gold overhead-accounting bytes -20
user@PE1# set traffic-control-profiles tcp_gold guaranteed-rate 1g
user@PE1# set traffic-control-profiles tcp_silver scheduler-map silver
user@PE1# set traffic-control-profiles tcp_silver shaping-rate 1g
user@PE1# set traffic-control-profiles tcp_silver overhead-accounting bytes -20
user@PE1# set traffic-control-profiles tcp_silver guaranteed-rate 500m
```

5. Map the traffic control profiles to their respective physical or logical interface.

```
[edit class-of-service]
user@PE1# set interfaces xe-9/0/3 output-traffic-control-profile tcp_ifd
user@PE1# set interfaces xe-9/0/3 unit 1 output-traffic-control-profile tcp_gold
user@PE1# set interfaces xe-9/0/3 unit 2 output-traffic-control-profile tcp_silver
```

6. Configure the scheduler maps.

```
[edit class-of-service]
user@PE1# set scheduler-maps gold forwarding-class be1 scheduler gold_internet
user@PE1# set scheduler-maps gold forwarding-class ef1 scheduler gold_video
user@PE1# set scheduler-maps gold forwarding-class af1 scheduler gold_voice
user@PE1# set scheduler-maps gold forwarding-class nc1 scheduler gold_reserved
user@PE1# set scheduler-maps silver forwarding-class be scheduler silver_internet
user@PE1# set scheduler-maps silver forwarding-class ef scheduler silver_video
user@PE1# set scheduler-maps silver forwarding-class af scheduler silver_voice
user@PE1# set scheduler-maps silver forwarding-class nc scheduler silver_reserved
```

7. Configure the schedulers.

```
[edit class-of-service]
user@PE1# set schedulers gold_internet excess-rate percent 40
user@PE1# set schedulers gold_internet buffer-size percent 20
user@PE1# set schedulers gold_internet priority low
user@PE1# set schedulers gold_video transmit-rate percent 50
user@PE1# set schedulers gold_video buffer-size percent 50
user@PE1# set schedulers gold_voice shaping-rate percent 10
user@PE1# set schedulers gold_voice buffer-size percent 10
user@PE1# set schedulers gold_voice priority strict-high
user@PE1# set schedulers gold_reserved excess-rate percent 20
user@PE1# set schedulers gold_reserved buffer-size percent 10
user@PE1# set schedulers gold_reserved priority low
user@PE1# set schedulers silver_internet excess-rate percent 40
user@PE1# set schedulers silver_internet buffer-size percent 20
user@PE1# set schedulers silver_internet priority low
user@PE1# set schedulers silver_video transmit-rate percent 50
user@PE1# set schedulers silver_video buffer-size percent 50
user@PE1# set schedulers silver_voice shaping-rate percent 10
user@PE1# set schedulers silver_voice buffer-size percent 10
```

```
user@PE1# set schedulers silver_voice priority strict-high
user@PE1# set schedulers silver_reserved excess-rate percent 20
user@PE1# set schedulers silver_reserved buffer-size percent 10
user@PE1# set schedulers silver_reserved priority low
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces` and `show class-of-service` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@PE1# show interfaces
interfaces {
    xe-9/0/3 {
        per-unit-scheduler;
        vlan-tagging;
        unit 1 {
            vlan-id 101;
            family inet {
                address 10.1.1.1/24;
            }
        }
        unit 2 {
            vlan-id 102;
            family inet {
                address 10.2.1.1/24;
            }
        }
    }
}
```

```
user@PE1# show class-of-service
class-of-service {
    classifiers {
        inet-precedence c8 {
            forwarding-class be {
                loss-priority low code-points 000;
            }
            forwarding-class ef {
                loss-priority low code-points 001;
            }
            forwarding-class af {
                loss-priority low code-points 010;
            }
            forwarding-class nc {
                loss-priority low code-points 011;
            }
            forwarding-class be1 {
                loss-priority low code-points 100;
            }
            forwarding-class ef1 {
                loss-priority low code-points 101;
            }
            forwarding-class af1 {
                loss-priority low code-points 110;
            }
            forwarding-class nc1 {
                loss-priority low code-points 111;
            }
        }
    }
    forwarding-classes {
        queue 0 be;
        queue 1 ef;
        queue 2 af;
        queue 3 nc;
        queue 4 be1;
        queue 5 ef1;
        queue 6 af1;
        queue 7 nc1;
    }
    traffic-control-profiles {
        tcp_ifd {
            shaping-rate 2500000000;
            overhead-accounting bytes -20;
        }
        tcp_gold {
            scheduler-map gold;
            shaping-rate 2500000000;
            overhead-accounting bytes -20;
            guaranteed-rate 1g;
        }
        tcp_silver {
            scheduler-map silver;
            shaping-rate 1g;
            overhead-accounting bytes -20;
            guaranteed-rate 500m;
        }
    }
    interfaces {
        xe-9/0/3 {
            output-traffic-control-profile tcp_ifd;
            unit 1 {
                output-traffic-control-profile tcp_gold;
            }
            unit 2 {
                output-traffic-control-profile tcp_silver;
            }
        }
    }
    scheduler-maps {
        gold {
            forwarding-class be1 scheduler gold_internet;
            forwarding-class ef1 scheduler gold_video;
            forwarding-class af1 scheduler gold_voice;
            forwarding-class nc1 scheduler gold_reserved;
        }
        silver {
            forwarding-class be scheduler silver_internet;
            forwarding-class ef scheduler silver_video;
            forwarding-class af scheduler silver_voice;
            forwarding-class nc scheduler silver_reserved;
        }
    }
    schedulers {
        gold_internet {
            excess-rate percent 40;
            buffer-size percent 20;
            priority low;
        }
        gold_video {
            transmit-rate percent 50;
            buffer-size percent 50;
        }
        gold_voice {
            shaping-rate percent 10;
            buffer-size percent 10;
            priority strict-high;
        }
        gold_reserved {
            excess-rate percent 20;
            buffer-size percent 10;
            priority low;
        }
        silver_internet {
            excess-rate percent 40;
            buffer-size percent 20;
            priority low;
        }
        silver_video {
            transmit-rate percent 50;
            buffer-size percent 50;
        }
        silver_voice {
            shaping-rate percent 10;
            buffer-size percent 10;
            priority strict-high;
        }
        silver_reserved {
            excess-rate percent 20;
            buffer-size percent 10;
            priority low;
        }
    }
}
```

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

RELATED DOCUMENTATION

Applying Scheduler Maps and Shaping Rate to Logical Interfaces and VLANs | **394**

per-unit-scheduler

Applying a Shaping Rate to Physical Interfaces Overview

On supported platforms, you can configure physical interfaces to shape traffic based on the rate-limited bandwidth of the total interface bandwidth. This allows you to shape the output of the physical interface, so that the interface transmits less traffic than it is physically capable of carrying.

If you do not configure a shaping rate on the physical interface, the default physical interface bandwidth is based on the channel bandwidth and the time slot allocation.

In general, the physical interface speed is the basis for calculating the various queue parameters for a physical interface such as delay buffer size, weighted round- robin (WRR) weight, drop profile, and so forth. However, when you apply a shaping rate by including the `shaping-rate` statement, the shaping rate on that physical interface becomes the basis for calculating all the queue parameters for that physical interface.

The minimum shaping rate value for the physical interface depends on the platform and line card. The maximum value of shaping rate is limited by the maximum transmission rate of the interface.

RELATED DOCUMENTATION

Configuring the Shaping Rate for Physical Interfaces | 410

Configuring the Shaping Rate for Physical Interfaces

To configure the shaping rate on the physical interface, either include the `shaping-rate` statement at the `[edit class-of-service interfaces` *`interface-name`*`]` hierarchy level or include the `output-traffic-control-profile` statement at the `[edit class-of-service interfaces` *`interface-name`*`]` hierarchy level.

You can specify a peak bandwidth rate in bps, either as a complete decimal number or as a decimal number followed by the abbreviation `k` (1000), `m` (1,000,000), or `g` (1,000,000,000). For physical interfaces, the range is from 1000 through 6,400,000,000,000 bps.

The maximum value of `shaping-rate` is limited by the maximum transmission rate of the interface.

The following are two sample configurations for applying a shaping rate of 5 Gbps on a physical interface (xe-4/0/0):

Apply a shaping rate at the `[edit class-of-service interfaces` *`interface-name`*`]` hierarchy:

```
[edit class-of-service]
interfaces {
    xe-4/0/0 {
        shaping-rate 5g;
    }
}
```

Apply a shaping rate using a traffic control profile:

```
[edit class-of-service]
traffic-control-profiles {
    shaping-output {
        shaping-rate 5g;
    }

}
interfaces {
    xe-4/0/0 {
        output-traffic-control-profile shaping-output;
    }
}
```

To view the results of your configuration, issue the following `show` commands:

- **show class-of-service interface *interface-name***
- **show interfaces *interface-name* extensive**

RELATED DOCUMENTATION

Applying a Shaping Rate to Physical Interfaces Overview | 410

Burst Size Configuration

IN THIS SECTION

You can explicitly configure the burst size for shapers in a traffic control profile.

The shaping burst size determines the maximum number of bytes that can be sent through a shaper during a burst. The guaranteed burst size determines when the scheduler moves from green to yellow.

The burst size limits the number of credits that can be accumulated for scheduling. Configuring a burst size is only useful in the case when traffic is sent after a long lull period so that credits can be accumulated until the burst size limit is reached. When traffic is continuous, credits are not accumulated, and the burst size limit is not reached.

If no burst size value is specified when the shaping rate or guaranteed rate is configured, then a default burst size (expressed as a time value) is applied. The default shaping burst size is 10 ms of the shaping rate (that is, 10*shaping rate/1000 bytes). The minimum value is 2048 bytes to accommodate the minimum of 1 MTU.

The burst size value is adjusted and rounded off to meet the restrictions enforced by the hardware. Thus, the actual burst size in the hardware might vary slightly from the configured value.

To enable this feature, include the `burst-size` statement at the following hierarchy levels:

```
[edit class-of-service traffic-control-profiles shaping-rate]
[edit class-of-service traffic-control-profiles guaranteed-rate]
```

NOTE: The `guaranteed-rate` burst size value cannot be greater than the `shaping-rate` burst size.

Configuring Burst Size for Shapers

This section shows how to set the `burst-size` while configuring the `shaping-rate` and `guaranteed-rate` under the `[edit class-of-service traffic-control-profiles` *`profile-name`*`]` hierarchy level.

In following configuration for tcp1, the `shaping-rate` burst size is set to 5 KB, and the `guaranteed-rate` burst size is set to 3 KB under the `traffic-control-profiles` statement. To apply this configuration to a logical interface (ifl), the traffic-control-profile is attached to the ifl.

```
class-of-service {
    traffic-control-profiles {
        tcp1 {
            shaping-rate 100m burst-size 5k;
            guaranteed-rate 50m burst-size 3k;
        }
        tcp2 {
            shaping-rate 100m burst-size 5k;
        }
    }

    interfaces {
        interface-set ifset1 {
            output-traffic-control-profile tcp1;
        }
        ge-1/2/1 {
            unit 0 {
                output-traffic-control-profile tcp1;
            }
        }
        ge-1/2/2 {
            output-traffic-control-profile tcp2;
        }
    }
}
```

RELATED DOCUMENTATION

guaranteed-rate

shaping-rate

Example: Limiting Egress Traffic on an Interface Using Port Shaping for CoS

IN THIS SECTION

This example shows how using port shaping as a form of class of service (CoS) enables you to limit traffic on an interface, so that you can control the amount of traffic passing through the interface.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos OS. The example shown here was tested and verified on MX Series routers running a supported Junos OS release.

Overview

IN THIS SECTION

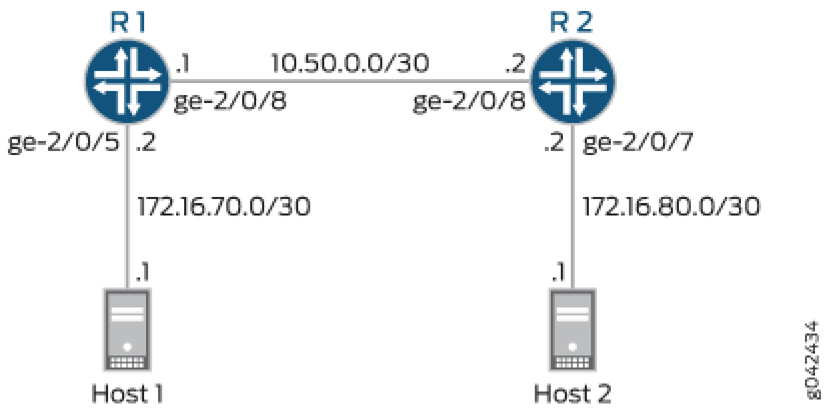
The purpose of this example is to demonstrate how port shaping enables you to shape the traffic passing through an interface to a rate that is less than the line rate for that interface. When you configure port shaping on an interface, you are essentially specifying a value that indicates the maximum amount of traffic that can pass through the interface. This value must be less than the maximum bandwidth for that interface. When you configure port shaping, you can specify either the maximum rate at which traffic can pass through the interface or as a percentage of the bandwidth of the interface.

In this example the port shaping is done on Device R1. The information required for implementing port shaping on Device R2 is not included in this example. However, you can use the port shaping information in Device R1 (making changes for the interfaces used) and apply it to Device R2 to achieve port shaping on Device R2.

Topology

This example uses the topology in Figure 35 on page 415.

Figure 35: Port Shaping Scenario



Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1 Using Only Class of Service

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set class-of-service interfaces ge-2/0/8 shaping-rate 160k
set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Device R1 Using Traffic Control Profiles and Class of Service

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set class-of-service traffic-control-profiles output shaping-rate 160k
set class-of-service traffic-control-profiles output shaping-rate burst-size 30k
set class-of-service interfaces ge-2/0/8 output-traffic-control-profile output
set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Device R2

```
set interfaces ge-2/0/7 description to-Host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.14.1/32
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

You can configure port shaping on network interfaces, aggregated Ethernet interfaces (also known as link aggregation groups (LAGs)), and loopback interfaces.

To configure Device R1:

1. Configure the device interfaces.

```
[edit]
user@R1# set interfaces ge-2/0/5 description to-Host
user@R1# set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set interfaces ge-2/0/8 description to-R2
user@R1# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set interfaces lo0 unit 0 description looback-interface
user@R1# set interfaces lo0 unit 0 family inet address 192.168.13.1/32
```

2. Configure port shaping using only class of service.

```
[edit]
user@R1# set class-of-service interfaces ge-2/0/8 shaping-rate 160k
```

3. Configure port shaping using traffic control profiles and class of service.

NOTE: If you configure a fixed shaping rate, you can configure an optional burst size in bytes. If you configure the shaping rate as a percentage, the `burst-size` option is not allowed.

```
[edit]
user@R1# set class-of-service traffic-control-profiles output shaping-rate 160k
user@R1# set class-of-service traffic-control-profiles output shaping-rate burst-size 30k
user@R1# set class-of-service interfaces ge-2/0/8 output-traffic-control-profile output
```

4. Configure OSPF.

```
[edit]
user@R1# set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@R1# set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

To configure Device R2:

1. Configure the device interfaces.

```
[edit interfaces]
set interfaces ge-2/0/7 description to-Host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.14.1/32
```

2. Configure OSPF.

```
[edit ]
user@R1# set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
user@R1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@R1# set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces` , `show class-of-service`, and `show protocols ospf` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Results for R1

```
user@R1# show interfaces
ge-2/0/5 {
    description to-Host;
    unit 0 {
        family inet {
            address 172.16.70.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R2;
    unit 0 {
        family inet {
            address 10.50.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
```

Configuring Port Shaping Using only Class-of-Service

```
user@R1# show class-of-service
interfaces {
    ge-2/0/8 {
        shaping-rate 160k;
    }
}
```

Configuring Port Shaping Using Traffic Control Profiles and Class of Service

```
user@R1# show class-of-service
traffic-control-profiles {
    output {
        shaping-rate 160k burst-size 30k;
    }
}
interfaces {
    ge-2/0/8 {
        output-traffic-control-profile output;
    }
}
```

```
user@R1# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/5.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R1, enter `commit` from configuration mode.

Results for R2

```
user@R2# show interfaces
ge-2/0/7 {
    description to-Host;
    unit 0 {
        family inet {
            address 172.16.80.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.50.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.14.1/32;
        }
    }
}
```

```
user@R2# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/7.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R2, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Clearing the Counters

Purpose

Confirm that the interface counters are cleared.

Action

On Device R1, run the `clear interfaces statistics ge-2/0/8` command to reset the interface statistics to 0.

```
user@R1> clear interfaces statistics ge-2/0/8
```

Sending TCP Traffic into the Network and Monitoring the Port Shaping

Purpose

Make sure that the traffic is rate-limited on the output interface (ge-2/0/8) on Device R1 by sending traffic into the network using the host connected to Device R1.

Action

1. Use a traffic generator to send several continuous streams of TCP packets with a source port of 80.

   The -s flag sets the source port. The -k flag causes the source port to remain steady at 80 instead of incrementing. The -d flag sets the packet size. The -c flag sets the packet count to be sent.

   The destination IP address of 172.16.80.1 represents a user that is downstream of Device R2. The user has requested a webpage from the host (the webserver emulated by the traffic generator), and the packets are sent in response to the request.

   NOTE: Remember in this example the port shaping has been set to 160 Kbps.

```
[user@host]#  hping 172.16.80.1 -s 80 -k -d 1500 -c 20 &
 hping 172.16.80.1 -s 80 -k -d 1500 -c 20 &
.
.
.
```

2. On Device R1, check the interface counters by using the `show interfaces extensive ge-2/0/8` command.

```
user@R1> show interfaces extensive ge-2/0/8
  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0                            17244                 3741                13470
    1                               13                   13                    0
    2                                0                    0                    0
    3                           149363               149363                    0
  Queue number:         Mapped forwarding classes
    0                   best-effort
    1                   expedited-forwarding
    2                   assured-forwarding
    3                   network-control
```

Meaning

In the output you can see that 13470 packets have been dropped. These are the packets that exceeded the 160 Kbps shaping rate configured on ge-2/0/8.

RELATED DOCUMENTATION

Routing Policies, Firewall Filters, and Traffic Policers User Guide

Example: Configuring a Two-Rate Three-Color Policer

Configuring Input Shaping Rates for Both Physical and Logical Interfaces

You can apply input shaping rates to both the physical interface and its logical interfaces. The rate specified at the physical level is distributed among the logical interfaces based on their input shaping-rate ratio.

To configure an input shaper on the physical interface:

- Specify the phyiscal interface and associated shaping rate.

```
[edit]
user@host# edit class-of-service interfaces interface-name
user@host# set input-shaping-rate rate
```

To configure an input shaper on the logical interface:

- Specify the physical and logical interface names and associated shaping rate.

```
[edit]
user@host# edit class-of-service interfaces interface-name unit logical-unit-number
user@host# set input-shaping-rate (percent percentage | rate)
```

  For each logical interface, you can specify a percentage of the physical rate or an actual rate. The Junos OS software converts actual rates into percentages of the physical rate.

CHAPTER 8

Setting Transmission Order with Scheduler Priorities and Hierarchical Scheduling

IN THIS CHAPTER

Transmission Priority Scheduling

SUMMARY

Junos supports multiple levels of transmission priority, which in order of increasing priority are `low`, `low-medium`, `low-high`, `medium-low`, `medium-high`, `high`, `strict-high`, and `low-latency`. This allows the software to service higher-priority queues before lower-priority queues. Which transmission priority levels are supported can vary depending on the platform and software release.

IN THIS SECTION

Prioritize Traffic through Priority Scheduling

IN THIS SECTION

Priority scheduling determines the order in which an output interface transmits traffic from its queues, thus ensuring that queues containing important traffic have better access to the outgoing interface. Junos accomplishes priority scheduling by examining the assigned priority of each individual queue and whether each individual queue is within its defined bandwidth profile. Junos determines whether an individual queue is within its bandwidth profile by comparing, at regular intervals, the amount of data transmitted by the queue against the amount of bandwidth allocated to it by the configured scheduler transmission rate (`transmit-rate`) defined at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level. When the transmitted amount is less than the allocated amount, the queue is considered to be *in profile*. A queue is *out of profile* when its transmitted amount is larger than its allocated amount.

The queues for a given output physical or logical interface are divided into sets based on their priority. Any such set contains queues of the same priority.

Junos traverses the sets in descending order of priority. If at least one of the queues in the set has a packet to transmit, the software selects that set. A queue from the set is selected based on the weighted round robin (WRR) algorithm, which operates within the set.

Junos performs priority queuing using the following steps:

1. The software locates all high-priority queues that are currently in profile. These queues are serviced first in a weighted round-robin fashion.
2. The software locates all medium-high priority queues that are currently in profile. These queues are serviced second in a weighted round-robin fashion.
3. The software locates all medium-low priority queues that are currently in profile. These queues are serviced third in a weighted round-robin fashion.
4. The software locates all low-high priority queues that are currently in profile. These queues are serviced second in a weighted round-robin fashion.
5. The software locates all low-medium priority queues that are currently in profile. These queues are serviced third in a weighted round-robin fashion.
6. The software locates all low-priority queues that are currently in profile. These queues are serviced fourth in a weighted round-robin fashion.

7. The software locates all high-priority queues that are currently out of profile and are not rate limited. The weighted round-robin algorithm is applied to these queues for servicing.

8. The software locates all medium-high priority queues that are currently out of profile and are not rate limited. The weighted round-robin algorithm is applied to these queues for servicing.

9. The software locates all medium-low priority queues that are currently out of profile and are not rate limited. The weighted round-robin algorithm is applied to these queues for servicing.

10. The software locates all low-high priority queues that are currently out of profile and are not rate limited. The weighted round-robin algorithm is applied to these queues for servicing.

11. The software locates all low-medium priority queues that are currently out of profile and are not rate limited. The weighted round-robin algorithm is applied to these queues for servicing.

12. The software locates all low-priority queues that are currently out of profile and are also not rate limited. These queues are serviced last in a weighted round-robin manner.

Low Latency Queuing (LLQ) Overview

On platforms that support low latency queuing (LLQ). LLQ enables delay-sensitive data to have preferential treatment over other traffic. A `low-latency` queue has the highest priority over any other priority queues, including `strict-high` queues, as well as a low delay scheduling profile.

For port scheduling of virtual output queues (VOQs), low latency VOQs receive their own dedicated egress queue. High priority VOQs receive a second dedicated egress queue, and low priority VOQs receive a third dedicated egress queue.

Due to the scheduling hierarchy of hierarchical class of service (HCoS), a hierarchical scheduling can use a maximum of two egress queues. Therefore for hierarchical scheduling of VOQs, low latency VOQs and high priority VOQs receive a common dedicated egress queue, and low priority VOQs receive the second dedicated egress queue.

NOTE: We recommend the following when configuring low-latency VOQs:

- Use policers to normalize the burstiness of traffic before it reaches a low-latency VOQ.
- Configure a maximum of two low-latency VOQs on a physical or logical interface.
- Classify and schedule traffic (that is, reserve bandwidth) for low-latency VOQs so that there is no congestion for those queues.

NOTE: Low-latency queues receive the same buffers as other queues to efficiently use the limited hardware VOQ buffer profiles.

Strict-High Priority Configuration Overview

Depending on the platform can configure one or more queues per interface to have `strict-high` priority, which works the same as `high` priority, but provides unlimited transmission bandwidth. As long as the queue with `strict-high` priority has traffic to send, it receives precedence over all other queues, except queues with `high` priority. Queues with `strict-high` and `high` priority take turns transmitting packets until the `strict-high` queue is empty, the `high` priority queues are empty, or the `high` priority queues run out of bandwidth credit. Only when these conditions are met can lower priority queues send traffic.

On platforms that support multiple `strict-high` queues per interface, the hardware services the queues in the descending order of queue numbers marked with `strict-high` priority.

When you configure a queue to have `strict-high` priority, you do not need to include the `transmit-rate` statement in the queue configuration at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level because the transmission rate of a `strict-high` priority queue is not limited by the WRR configuration. If you do configure a transmission rate on a `strict-high` priority queue, it does not affect the WRR operation. The transmission rate does, however, affect the calculation of the delay buffer and also serves as a placeholder in the output of commands such as the `show interface queue` command.

`strict-high` priority queues might starve `low` priority queues, and under certain circumstances might limit `high` priority queues. The `high` priority allows you to protect traffic classes from being starved by traffic in a `strict-high` queue. For example, a network-control queue might require a small bandwidth allocation (say, 5 percent). You can assign `high` priority to this queue to prevent it from being underserved.

A queue with `strict-high` priority supersedes bandwidth guarantees for queues with lower priority; therefore, we recommend that you use the `strict-high` priority to ensure proper ordering of special traffic, such as voice traffic. You can preserve bandwidth guarantees for queues with lower priority by allocating to the queue with `strict-high` priority only the amount of bandwidth that it generally requires by applying the `rate-limit` option to the `strict-high` queue's transmission rate. For example, consider the following allocation of transmission bandwidth:

- Q0 BE—20 percent, low priority
- Q1 EF—30 percent, strict-high priority
- Q2 AF—40 percent, low priority
- Q3 NC—10 percent, low priority

This bandwidth allocation assumes that, in general, the EF forwarding class requires only 30 percent of an interface's transmission bandwidth. However, if short bursts of traffic are received on the EF forwarding class, and the `rate-limit` option is not applied, 100 percent of the bandwidth is given to the EF forwarding class because of the `strict-high` setting.

Configure Schedulers for Priority Scheduling

This topic describes how to configure priority scheduling.

```
[edit class-of-service schedulers scheduler-name]
priority priority-level;
```

The priority level can be `low`, `low-medium`, `low-high`, `medium-low`, `medium-high`, `high`, `strict-high`, or `low-latency`. The priorities map to numeric priorities in the underlying hardware. In some cases, different priorities behave similarly, because two software priorities behave differently only if they map to two distinct hardware priorities.

Higher-priority queues transmit packets ahead of lower priority queues as long as the higher-priority forwarding classes retain enough bandwidth credit. When you configure a higher-priority queue with a significant fraction of the transmission bandwidth, the queue might lock out (or *starve*) lower priority traffic.

In the following example procedure, you create a scheduler, configure the mapping between the scheduler and the forwarding class, and assign the scheduler to an interface.

1. Configure a scheduler, `be-sched`, with `medium-low` priority.

   ```
   [edit]
   user@host# edit class-of-service schedulers be-sched
   user@host# set priority medium-low
   ```

2. Configure a scheduler map, `be-map`, that associates `be-sched` with the `best-effort` forwarding class.

   ```
   [edit class-of-service]
   user@host# set scheduler-maps be-map forwarding-class best-effort scheduler be-sched
   ```

3. Assign the `be-map` scheduler map to an interface, `et-0/0/0`.

   ```
   [edit class-of-service]
   user@host# set interfaces et-0/0/0 scheduler-map be-map
   ```

4. Verify your configuration.

```
[edit class-of-service]
user@host#  show
```

```
schedulers {
    be-sched {
        priority medium-low;
    }
}
scheduler-maps {
    be-map {
        forwarding-class best-effort scheduler be-sched;
    }
}
et-0/0/0 {
    scheduler-map be-map;
}
```

5. Save your configuration.

```
[edit class-of-service]
user@host#  commit
```

Platform-Specific Priority Schedulers Behavior

Use Feature Explorer to confirm platform and release support for scheduling priority.

Use the following table to review platform-specific behavior for your platforms:

Platforms	Platform-specific Behavior
ACX5048 and ACX5096 routers	• ACX5048 and ACX5096 routers support CIR among strict-priority queues. There is no implicit queue number-based priority among the strict-priority queues. • ACX5048 and ACX5096 routers support configuring drop profiles for loss-priority `low`, `medium-high`, and `high` for non-TCP protocols as well.
ACX7000 Series	• ACX7000 routers support multiple queues with the same priority. • ACX7000 routers support multiple `strict-high` queues per interface. • ACX7000 Series routers support all eight levels of priority for port scheduling and six levels for HCoS scheduling (all except `low-medium` and `low-high`). • ACX7000 Series routers do not guarantee round-robin distribution between queues of the same priority. • You can configure `trasmit-rate` only on low priority queues.
EX4400	On EX4400 switches, applying strict-high priority schedulers to queues 0 through 3 also applies strict-high priority to queues 8 through 11. Therefore, Juniper recommends applying strict-high priority schedulers only to queues 4 through 7.

(Continued)

Platforms	Platform-specific Behavior
EX4600 and QFX5100	On QFX5100 and EX4600 switches, you can configure only one queue as a strict-high priority queue. We recommend that you always apply a shaping rate to strict-high priority queues to prevent them from starving other queues. A shaping rate (shaper) sets the maximum amount of bandwidth a queue can consume. (Unlike using the transmit rate on a QFX10000 switch to limit traffic that receives strict-high priority treatment, traffic that exceeds the shaping rate is dropped, and is not treated as best-effort traffic that shares in excess bandwidth.) If you do not apply a shaping rate to limit the amount of bandwidth a strict-high priority queue can use, then the strict-high priority queue can use all of the available port bandwidth and starve other queues on the port.
QFX10000	On QFX10000 switches, you can configure as many strict-high priority queues as you want. On QFX10000 switches, we strongly recommend that you apply a transmit rate to strict-high priority queues to prevent them from starving other queues. A transmit rate configured on a strict-high priority queue limits the amount of traffic that receives strict-high priority treatment to the amount or percentage set by the transmit rate.

RELATED DOCUMENTATION

How Schedulers Define Output Queue Properties | **342**

Associating Schedulers with Fabric Priorities

IN THIS SECTION

On supported platforms, you can associate a scheduler with a class of traffic that has a specific priority while transiting the fabric. Traffic transiting the fabric can have two priority values: `low` or `high`. To associate a scheduler with a fabric priority, include the `priority` and `scheduler` statements at the `[edit class-of-service fabric scheduler-map]` hierarchy level:

```
[edit class-of-service fabric scheduler-map]
priority (high | low) scheduler scheduler-name;
```

NOTE: For a scheduler that you associate with a fabric priority, include only the `drop-profile-map` statement at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level. You cannot include the `buffer-size`, `transmit-rate`, and `priority` statements at that hierarchy level.

Example: Associating a Scheduler with a Fabric Priority

Associate a scheduler with a class of traffic that has a specific priority while transiting the fabric:

```
[edit class-of-service]
schedulers {
    fab-be-scheduler {
        drop-profile-map loss-priority low protocol any drop-profile fab-profile-1;
        drop-profile-map loss-priority high protocol any drop-profile fab-profile-2;
    }
    fab-ef-scheduler {
        drop-profile-map loss-priority low protocol any drop-profile fab-profile-3;
        drop-profile-map loss-priority high protocol any drop-profile fab-profile-4;
    }
}
drop-profiles {
    fab-profile-1 {
        fill-level 100 drop-probability 100;
        fill-level 85 drop-probability 50;
    }
    fab-profile-2 {
        fill-level 100 drop-probability 100;
        fill-level 95 drop-probability 50;
    }
    fab-profile-3 {
        fill-level 75 drop-probability 100;
        fill-level 95 drop-probability 50;
    }
    fab-profile-4 {
        fill-level 100 drop-probability 100;
        fill-level 80 drop-probability 50;
    }
}
fabric {
    scheduler-map {
        priority low scheduler fab-be-scheduler;
        priority high scheduler fab-ef-scheduler;
    }
}
```

RELATED DOCUMENTATION

Forwarding Classes and Fabric Priority Queues | 319

Hierarchical Class of Service Overview

IN THIS SECTION

Hierarchical class of service (HCoS) is the ability to apply traffic schedulers and shapers to a hierarchy of *scheduler nodes*. Each level of the scheduler hierarchy can be used to shape traffic based on different criteria such as application, user, VLAN, slice, and physical port.

This allows you to support the requirements of different services, applications, and users on the same physical device and physical infrastructure.

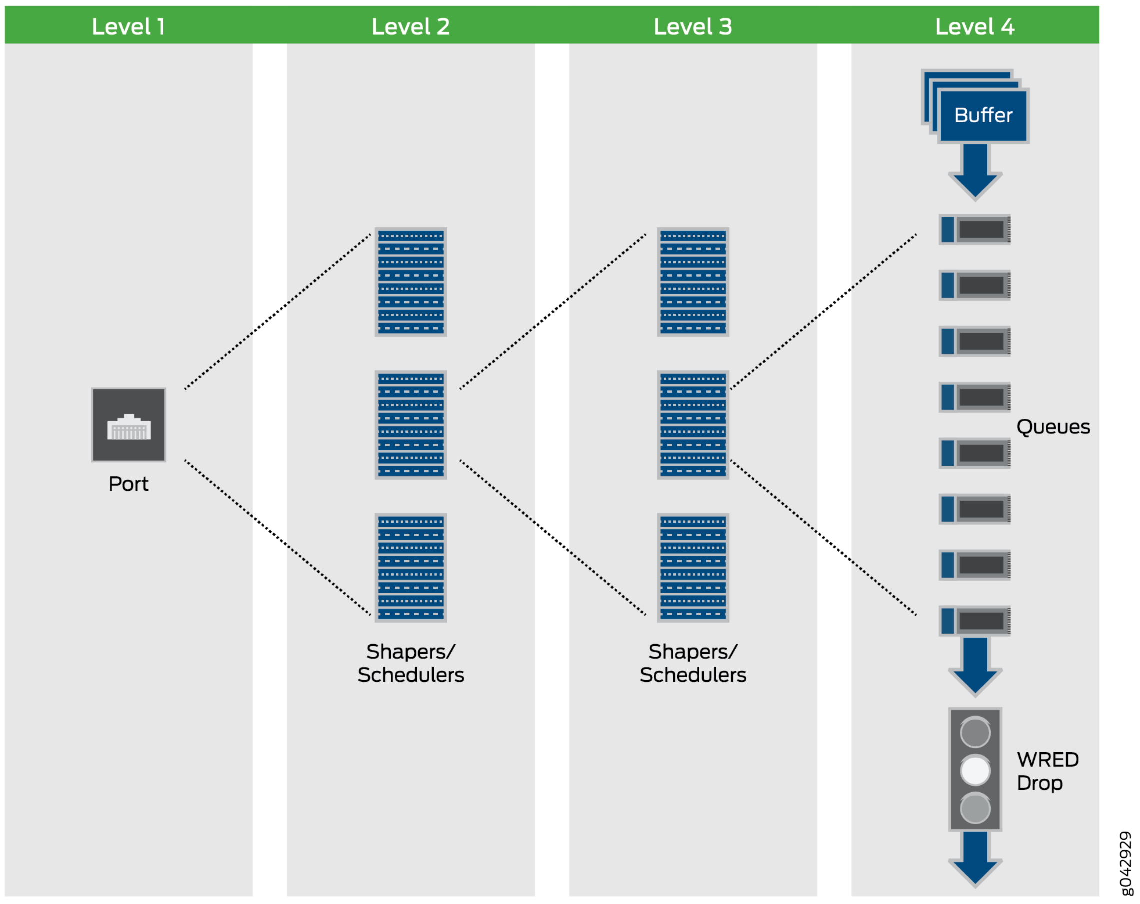
HCoS is implemented primarily using traffic classifiers at the ingress and hierarchical schedulers and shapers at the egress.

A classifier is a filter that labels traffic at the device ingress based on configurable parameters such as application or destination. Traffic is classified into what is called a forwarding equivalence class (FEC). The FEC defines a class of traffic that receives common treatment.

Schedulers, and their associated shapers, are the functions that control the traffic bandwidth, *jitter* (delay variation), and packet loss priority at the egress of the device.

Hierarchical schedulers are used to apply multiple levels of scheduling and shaping with each level applied to different classifications such as forwarding equivalence class, VLAN, and physical interface (port) as shown in Figure 36 on page 435.

Figure 36: Hierarchical Scheduling Architecture



NOTE: Hierarchical class of service is also referred to as Hierarchical Quality of Service (HQoS) in other vendor's documentation.

A typical application of HCoS is to configure multiple levels of egress schedulers and shapers, at the subscriber edge, using dynamic profiles to provide traffic shaping and prioritization at the subscriber VLAN level and for multiple classes of traffic.

Dynamic profiles are a mechanism that allows you to dynamically apply schedulers and shapers to individual subscribers or groups of subscribers.

To learn more about HCoS, the following topics are very helpful:

- "Junos CoS on MX Series 5G Universal Routing Platforms Overview" on page 749
- "CoS Features and Limitations on MX Series Routers" on page 750
- "CoS Features of the Router Hardware, PIC, MIC, and MPC Interface Families" on page 864

- "How Schedulers Define Output Queue Properties" on page 342
- *Subscriber Access Network Overview*
- *CoS for Subscriber Access Overview*
- *Hierarchical Class of Service for Subscriber Management Overview*

The Junos OS hierarchical schedulers support up to five levels of scheduler hierarchies on MX Series devices when using enhanced queuing Dense Port Concentrators (DPCs) or fine-grained queuing Modular Port Concentrators (MPCs), and Modular Interface Cards (MICs). It is important to know the capabilities of your hardware with respect to HCoS. The following are a few tips to help you:

- Only certain hardware supports the five-level scheduler hierarchy of HCoS.
- The number of queues and logical interfaces supported is dependent upon exactly what hardware you are using.
- The MX Series Packet Forwarding Engine handles guaranteed bandwidth and scheduler node weight differently than other Packet Forwarding Engines.
- The fine-grained queuing MPCs and MICs have a certain granularity with respect to the shaping and delay buffer values. The values used are not necessarily exactly the values configured.

To learn more about platform support for HCoS, use the Juniper Networks Feature Explorer (https://pathfinder.juniper.net/feature-explorer/). In the Feature Explorer, search on *hierarchical schedulers*.

In addition, it is important to note the following:

- HCoS is most frequently used to enforce service level agreements at the subscriber edged using dynamic traffic control profiles.
- Hierarchical schedulers can also be applied to Ethernet pseudowire interfaces, aggregated Ethernet interfaces, Layer 2 Tunnel Protocol (L2TP) network server (LNS) inline services, and GRE tunnels.
- Hierarchical ingress policing is a feature that is complimentary to and often used in conjunction with HCoS.
- There are other features in Junos OS that have similar sounding names.

NOTE: The *hierarchical scheduler and shaper* feature supported on the SRX Series Firewalls is not the HCoS feature described here.

Before planning HCoS for you network, you should learn about HCoS, define you needs, plan how you want to implement HCoS, and test the operation in a simulated environment.

Table 41: Resources for Learning More About HCoS

Document	Description
Day One: Deploying Basic QoS Juniper Networks Books	This book is a good resource for learning the basics of CoS on Juniper Networks devices.
Juniper MX-Series O'Reilly Media	Learn about the advanced features of HCoS. This book provides an in-depth description of how HCoS works and how it can be deployed. It also provides a lab tested topology and configuration example.
Day One: Dynamic Subscriber Management Juniper Networks Books	Learn how to use HCoS in conjunction with dynamic traffic control profiles for subscriber management. This book also includes troubleshooting.
QoS Enabled Networks John Wiley & Sons	This book is an additional source for studying QoS.

Documentation related to HCoS is consolidated in the Hierarchical Class of Service User Guide.

Platform-Specific HCoS Behavior

Use the following table to review platform-specific behaviors for your platform.

Table 42: Platform-Specific HCoS Behavior

Platform	Difference
PTX10008	By default, PTX10008 routers boot with PTX10K-LC1301-36DD line cards in interop mode. This mode does not support hierarchical class of service (HCoS) configuration on the PTX10K-LC1301-36DD line card. To enable HCoS on this line card, run the `set chassis interoperability express5-enhanced` command. Commit the change and reboot the router. After the reboot, the PTX10008 with the PTX10K-LC1301-36DD line card supports HCoS configuration.

RELATED DOCUMENTATION

Hierarchical Class of Service for Subscriber Management Overview

Hierarchical Class of Service Network Scenarios | 438

Understanding Hierarchical Scheduling | 440

Hierarchical Class of Service Network Scenarios

IN THIS SECTION

Hierarchical class of service (HCoS) can be used to provide granular control of traffic for a variety of different applications.

NOTE: Hierarchical class of service is also referred to as Hierarchical Quality of Service (HQoS) in other vendor's documentation.

Hierarchical class of service is most frequently used in the following scenarios:

Services to Subscribers

Multiservice network operators face a challenge to provide different types of services on the same infrastructure to residential and business subscribers. The network operator needs to make sure each subscriber gets the network resources they paid for and each service gets the network resources it needs to operate properly.

If no CoS is applied, one service could consume most of the bandwidth of the transmission infrastructure and starve the other services.

Using hierarchical class of service, the network edge device can have up to five levels of scheduling and prioritization. So the traffic can be shaped and prioritized per customer and per service type. Controlling traffic in this way provides the ability to deliver the required service level for each subscriber for each service type.

By allowing network operators to consolidate different services and multiple customers on the same physical infrastructure, hierarchical class of service helps maximize the ability to offer revenue generating services while simultaneously minimizing capital cost.

Services to Businesses

Hierarchical class of service is a valuable tool for service providers that support business customers who are running applications with different prioritization and scheduling requirements over the same infrastructure. In this scenario hierarchical class of service allows lower priority traffic to fully utilize the available bandwidth on a port, while simultaneously ensuring low latency and guaranteed bandwidth to higher priority traffic on the same port.

This allows a provider to consolidate different services on the same physical device and physical infrastructure thus optimizing network resources while maintaining the required level of service.

All of this maximizes revenue and minimizes cost

Wireless Backhaul

In a cellular network the operator might want to offer business services along with its cell tower traffic. One of the main challenges is to make sure that the time-sensitive cell traffic is not affected by the business services running on the same infrastructure. Each type of traffic has its own priority flows and bandwidth constraints. For example, wireless backhaul is very sensitive to fluctuations in the packet stream (Jitter) because it relies on synchronization.

In this scenario, hierarchical class of service allows each type of traffic to receive the required resources and quality of service while being delivered over the same infrastructure.

By consolidate different services on the same physical infrastructure, HCoS helps maximize revenue and minimize cost.

RELATED DOCUMENTATION

Hierarchical Class of Service Overview | 433

Hierarchical Class of Service for Subscriber Management Overview

Understanding Hierarchical Scheduling

IN THIS SECTION

Hierarchical class of service (HCoS) is a set of capabilities that enable you to apply unique CoS treatment for network traffic based on criteria such as user, application, VLAN, and physical port.

This allows you to support the requirements of different services, applications, and users on the same physical device and physical infrastructure.

This topic covers the following information:

Hierarchical Scheduling Terminology

Hierarchical scheduling introduces some new CoS terms and also uses some familiar terms in different contexts:

- Customer VLAN (C-VLAN)—A C-VLAN, defined by IEEE 802.1ad. A stacked VLAN contains an outer tag corresponding to the S-VLAN, and an inner tag corresponding to the C-VLAN. A C-VLAN often corresponds to CPE. Scheduling and shaping is often used on a C-VLAN to establish minimum and maximum bandwidth limits for a customer. See also *S-VLAN*.

- Interface set—A logical group of interfaces that describe the characteristics of set of service VLANs, logical interfaces, customer VLANs, or aggregated Ethernet interfaces. Interface sets establish the set and name the traffic control profiles. See also *Service VLAN*.

- Scheduler— A scheduler defines the scheduling and queuing characteristics of a queue. Transmit rate, scheduler priority, and buffer size can be specified. In addition, a drop profile may be referenced to describe WRED congestion control aspects of the queue. See also *Scheduler map*.

- Scheduler map—A scheduler map is referenced by traffic control profiles to define queues. The scheduler map establishes the queues that comprise a scheduler node and associates a forwarding class with a scheduler. See also *Scheduler*.

- Stacked VLAN—An encapsulation on an S-VLAN with an outer tag corresponding to the S-VLAN, and an inner tag corresponding to the C-VLAN. See also *Service VLAN* and *Customer VLAN*.

- Service VLAN (S-VLAN)—An S-VLAN, defined by IEEE 802.1ad, often corresponds to a network aggregation device such as a DSLAM. Scheduling and shaping is often established for an S-VLAN to provide CoS for downstream devices with little buffering and simple schedulers. See also *Customer VLAN*.

- Traffic control profile—Defines the characteristics of a scheduler node. Traffic control profiles are used at several levels of the CLI, including the physical interface, interface set, and *logical interface* levels. Scheduling and queuing characteristics can be defined for the scheduler node using the `shaping-rate`, `guaranteed-rate`, and `delay-buffer-rate` statements. Queues over these scheduler nodes are defined by referencing a scheduler map. See also *Scheduler* and *Scheduler map*.

- VLAN—Virtual LAN, defined on an Ethernet logical interface.

Scheduler Node-Level Designations in Hierarchical Scheduling

Scheduler hierarchies are composed of nodes and queues. Queues terminate the hierarchy. Nodes can be either root nodes, leaf nodes, or internal (non-leaf) nodes. Internal nodes are nodes that have other nodes as “children” in the hierarchy.

Scheduler hierarchies consist of levels, starting with Level 1 at the physical port. This chapter establishes a four-level scheduler hierarchy which, when fully configured, consists of the physical interface (Level 1), the interface set (Level 2), one or more logical interfaces (Level 3), and one or more queues (Level 4).

NOTE: Certain Juniper devices and line cards support up to five levels of scheduler hierarchies. The concepts presented in this topic apply similarly to five scheduler hierarchy levels.

Table 43 on page 441 describes the possible combinations of scheduler nodes and their corresponding node level designations for a hierarchical queuing MIC or MPC.

Table 43: Node Levels Designations in Hierarchical Scheduling

Scheduler Configuration for Hierarchical CoS	Hierarchical CoS Scheduler Nodes			
	Root Node	Internal (Non-Leaf) Nodes		Leaf Node
	Level 1	Level 2	Level 3	Level 4
One or more traffic control profiles configured on logical interfaces, but no interface-sets configured	Physical interfa ce	—	One or more logical interface s	One or more queues

Table 43: Node Levels Designations in Hierarchical Scheduling *(Continued)*

Scheduler Configuration for Hierarchical CoS	Hierarchical CoS Scheduler Nodes			
	Root Node	Internal (Non-Leaf) Nodes		Leaf Node
	Level 1	Level 2	Level 3	Level 4
Interface-sets (collections of logical interfaces) configured, but no traffic-control profiles configured on logical interfaces	Physical interfa ce	—	Interface-set	One or more queues
Fully configured scheduler nodes	Physical interfa ce	Interface-set	One or more logical interface s	One or more queues

The table illustrates how the configuration of an interface set or logical interface affects the terminology of hierarchical scheduler nodes. For example, suppose you configure an `interface-set` statement with logical interfaces (such as `unit 0` and `unit 2`) and a queue. In this case, the interface-set is an internal node at Level 2 of the scheduler node hierarchy. However, if there are no traffic control profiles attached to logical interfaces, then the interface set is at Level 3 of the hierarchy.

Hierarchical Scheduling at Non-Leaf Nodes

Whereas standard CoS scheduling is based on the scheduling and queuing characteristics of a router's egress ports and their queues, hierarchical CoS scheduling is based on the scheduling and queuing characteristics that span a hierarchy of *scheduler nodes* over a port. The hierarchy begins at Level 1, a *root node* at the physical interface (port) level of the CLI hierarchy and terminates at Level 4, a *leaf node* at the queue level. Between the root and leaf nodes of any scheduler hierarchy are one or more *internal nodes*, which are non-root nodes that have other nodes as "children" in the hierarchy.

Whereas you configure standard CoS scheduling by applying a *scheduler map* to each egress port to specify a forwarding class and a queue priority level, you configure hierarchical CoS scheduling with additional parameters. To configure hierarchical CoS scheduling, you apply a scheduler map to the queue level (Level 4) of a scheduler hierarchy, and you can apply a different *traffic control profile* at each of the other levels. A traffic control profile specifies not only a scheduler map (forwarding class and queue priority level) but also optional shaping rate (PIR), guaranteed transmit rate (CIR), burst rate, delay buffer rate, and drop profile.

Priority Propagation in Hierarchical Scheduling

Priority propagation is useful for mixed traffic environments when, for example, you want to make sure that the voice traffic of one customer does not suffer due to the data traffic of another customer. Nodes and queues are serviced in the order of their priority. The default priority of a queue is low, and you can explicitly configure a queue priority by including the `priority` statement at the `[edit class-of-service schedulers *scheduler-name*]` hierarchy level.

You cannot directly configure the priorities of all hierarchical scheduling elements. The priorities of internal nodes, for example, are determined as follows:

- The highest priority of an active child, that is, a child currently containing traffic. (Interface sets only take the highest priority of their active children.)
- Whether the node is above its configured guaranteed rate (CIR) or not (this is only relevant if the physical interface is in CIR mode).

Each queue has a configured priority and a hardware priority. The usual mapping between the configured priority and the hardware priority is shown in Table 44 on page 443.

Table 44: Queue Priority

Configured Priority	Hardware Priority
Strict-high	0
High	0
Medium-high	1
Medium-low	1
Low	2
MPCs also have configurable CLI priorities of `excess-priority high`, `excess-priority medium-high`, `excess-priority medium-low`, and `excess-priority low`. These priorities only take effect above the guaranteed rate.	

In CIR mode, the priority for each internal node depends on whether the highest active child node is above or below the guaranteed rate. The mapping between the highest active child's priority and the hardware priority below and above the guaranteed rate is shown in Table 45 on page 444.

Table 45: Internal Node Queue Priority for CIR Mode

Configured Priority of Highest Active Child Node	Hardware Priority Below Guaranteed Rate	Hardware Priority Above Guaranteed Rate
Strict-high	0	0
High	0	3
Medium-high	1	3
Medium-low	1	3
Low	2	3
Excess-priority high*	N/A	3
Excess-priority medium-high*	N/A	3
Excess-priority medium-low*	N/A	4
Excess-priority low*	N/A	4
* MPCs only		

In PIR-only mode, nodes cannot send if they are above the configured shaping rate. The mapping between the configured priority and the hardware priority is for PIR-only mode is shown in Table 46 on page 444.

Table 46: Internal Node Queue Priority for PIR-Only Mode

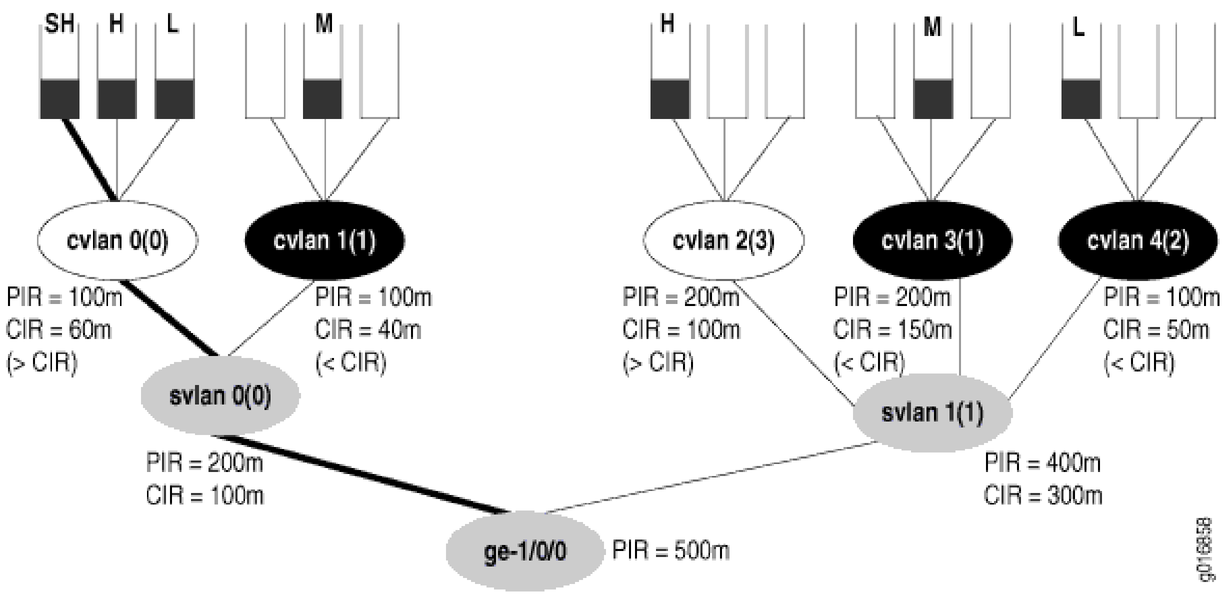
Configured Priority	Hardware Priority
Strict-high	0
High	0

Table 46: Internal Node Queue Priority for PIR-Only Mode *(Continued)*

Configured Priority	Hardware Priority
Medium-high	1
Medium-low	1
Low	2

A physical interface with hierarchical schedulers configured is shown in Figure 37 on page 445. The configured priorities are shown for each queue at the top of the figure. The hardware priorities for each node are shown in parentheses. Each node also shows any configured shaping rate (PIR) or guaranteed rate (CIR) and whether or not the queues is above or below the CIR. The nodes are shown in one of three states: above the CIR (clear), below the CIR (dark), or in a condition where the CIR does not matter (gray).

Figure 37: Hierarchical Schedulers and Priorities



In the figure, the strict-high queue for customer VLAN 0 (cvlan 0) receives service first, even though the customer VLAN is above the configured CIR (see Table 45 on page 444 for the reason: strict-high always has hardware priority 0 regardless of CIR state). Once that queue has been drained, and the priority of the node has become 3 instead of 0 (due to the lack of strict-high traffic), the system moves on to the medium queues next (cvlan 1 and cvlan 3), draining them in a round robin fashion (empty queue lose their hardware priority). The low queue on cvlan 4 (priority 2) is sent next, because that mode is below the CIR. Then the high queues on cvlan 0 and cvlan2 (both now with priority 3) are drained in a round robin fashion, and finally the low queue on cvlan 0 is drained (thanks to svlan 0 having a priority of 3).

RELATED DOCUMENTATION

Enhanced Queuing DPC CoS Properties | **916**

CoS Features and Limitations on MIC and MPC Interfaces | **938**

Understanding Hierarchical Scheduling for MIC and MPC Interfaces | **992**

Hierarchical CoS for Metro Ethernet Environments

In metro Ethernet environments, a virtual LAN (VLAN) typically corresponds to a customer premises equipment (CPE) device and the VLANs are identified by an inner VLAN tag on Ethernet frames (called the customer VLAN, or C-VLAN, tag). A set of VLANs can be grouped at the DSL access multiplexer (DSLAM) and identified by using the same outer VLAN tag (called the service VLAN, or S-VLAN, tag). The service VLANs are typically gathered at the Broadband Remote Access Server (B-RAS) level. Hierarchical schedulers let you provide shaping and scheduling at the service VLAN level as well as other levels, such as the physical interface. In other words, you can group a set of logical interfaces and then apply scheduling and shaping parameters to the logical interface set as well as to other levels.

You can apply CoS shaping and scheduling at one of four different levels, including the VLAN set level. (Some devices support up to five levels of scheduler hierarchies.)

The supported scheduler hierarchy is as follows:

- The physical interface (level 1)
- The service VLAN (level 2)
- The logical interface or customer VLAN (level 3)
- The queue (level 4)

Users can specify a traffic control profile (`output-traffic-control-profile`) that can specify a shaping rate, a guaranteed rate, and a scheduler map with transmit rate and buffer delay. The scheduler map contains the mapping of queues (forwarding classes) to their respective schedulers (schedulers define the properties for the queue). Queue properties can specify a transmit rate and buffer management parameters such as buffer size and drop profile.

To configure CoS hierarchical scheduling, you must enable hierarchical scheduling by including the `hierarchical-scheduler` statement at the physical interface.

RELATED DOCUMENTATION

Understanding Hierarchical Scheduling | 440

Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 992

Understanding Hierarchical CoS for Subscriber Interfaces

Hierarchical Schedulers and Traffic Control Profiles

When used, the interface set level of the hierarchy falls between the physical interface level (Level 1) and the *logical interface* (Level 3). Queues are always the highest level of the hierarchy. Certain devices and line cards support up to five levels of scheduler hierarchies. The concepts presented in this topic apply similarly to five scheduler hierarchy levels.

Hierarchical schedulers add CoS parameters to the interface-set level of the configuration. They use traffic control profiles to set values for parameters such as shaping rate (the peak information rate [PIR]), guaranteed rate (the committed information rate [CIR] on these interfaces), scheduler maps (assigning queues and resources to traffic), and so on.

The following CoS configuration places the following parameters in traffic control profiles at various levels:

- Traffic control profile at the port level (`tcp-port-level1`):
  - A shaping rate (PIR) of 100 Mbps
  - A delay buffer rate of 100 Mbps
- Traffic control profile at the interface set level (`tcp-interface-level2`):
  - A shaping rate (PIR) of 60 Mbps
  - A guaranteed rate (CIR) of 40 Mbps
- Traffic control profile at the logical interface level (`tcp-unit-level3`):
  - A shaping rate (PIR) of 50 Mbps
  - A guaranteed rate (CIR) of 30 Mbps
  - A scheduler map called `smap1` to hold various queue properties (level 4)
  - A delay buffer rate of 40 Mbps

In this case, the traffic control profiles look like this:

```
[edit class-of-service traffic-control-profiles]
tcp-port-level1 { # This is the physical port level
    shaping-rate 100m;
    delay-buffer-rate 100m;
}
tcp-interface-level2 { # This is the interface set level
    shaping-rate 60m;
    guaranteed-rate 40m;
}
tcp-unit-level3 { # This is the logical interface level
    shaping-rate 50m;
    guaranteed-rate 30m;
    scheduler-map smap1;
    delay-buffer-rate 40m;
}
```

Once configured, the traffic control profiles must be applied to the proper places in the CoS interfaces hierarchy.

```
[edit class-of-service interfaces]
interface-set level-2 {
    output-traffic-control-profile tcp-interface-level-2;
}
ge-0/1/0 {
    output-traffic-control-profile tcp-port-level-1;
    unit 0 {
        output-traffic-control-profile tcp-unit-level-3;
    }
}
```

In all cases, the properties for level 4 of the hierarchical schedulers (the queues) are determined by the scheduler map.

RELATED DOCUMENTATION

Oversubscribing Interface Bandwidth | 366

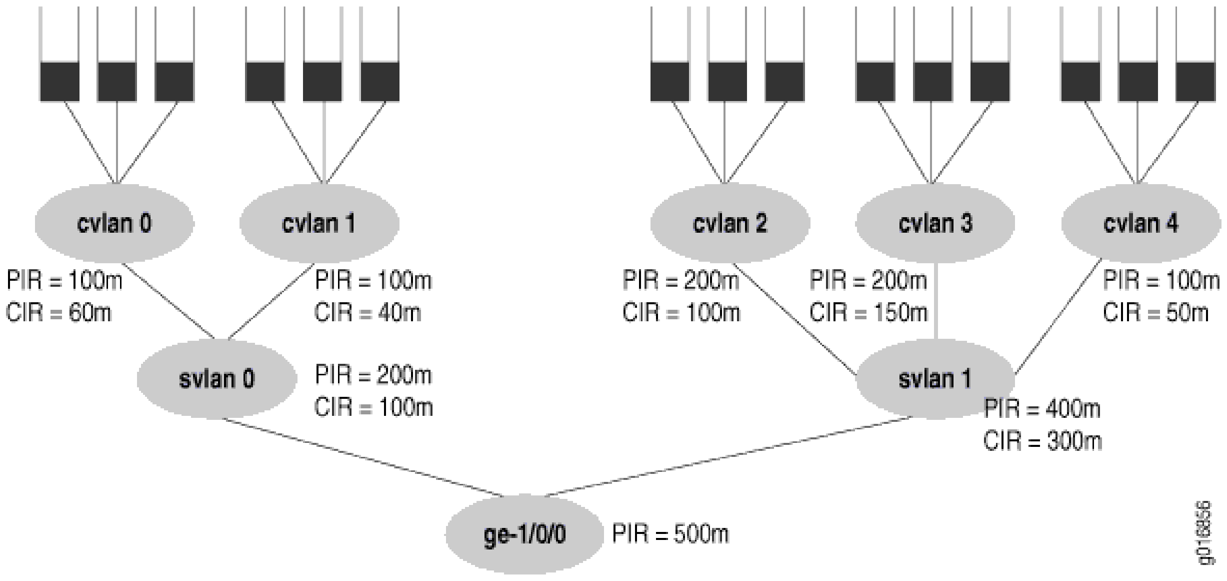
Providing a Guaranteed Minimum Rate | 376

Example: Building a Four-Level Hierarchy of Schedulers

IN THIS SECTION

This section provides a more complete example of building a 4-level hierarchy of schedulers. The configuration parameters are shown in Figure 38 on page 450. The queues are shown at the top of the figure with the other three levels of the hierarchy below.

Figure 38: Building a Scheduler Hierarchy



The figure’s PIR values are configured as the shaping rates and the CIRs are configured as the guaranteed rate on the Ethernet interface `ge-1/0/0`. The PIR can be oversubscribed (that is, the sum of the children PIRs can exceed the parent’s, as in `svlan 1`, where 200 + 200 + 100 exceeds the parent rate of 400)). However, the sum of the children node level’s CIRs must never exceed the parent node’s CIR, as shown in all the service VLANs (otherwise, the guaranteed rate could never be provided in all cases).

This configuration example presents all details of the CoS configuration for the interface in the figure (`ge-1/0/0`), including:

Configuring the Interface Sets

```
[edit interfaces]
interface-set svlan-0 {
    interface ge-1/0/0 {
        unit 0;
        unit 1;
    }
}
interface-set svlan-1 {
    interface ge-1/0/0 {
        unit 2;
        unit 3;
        unit 4;
    }
}
```

Configuring the Interfaces

The keyword to configure hierarchical schedulers is at the physical interface level, as is VLAN tagging and the VLAN IDs. In this example, the interface sets are defined by logical interfaces (units) and not outer VLAN tags. All VLAN tags in this example are customer VLAN tags.

```
[edit interface ge-1/0/0]
hierarchical-scheduler;
vlan-tagging;
unit 0 {
    vlan-id 100;
}
unit 1 {
    vlan-id 101;
}
unit 2 {
    vlan-id 102;
}
unit 3 {
    vlan-id 103;
}
unit 4 {
    vlan-id 104;
}
```

Configuring the Traffic Control Profiles

The traffic control profiles hold parameters for levels above the queue level of the scheduler hierarchy. This section defines traffic control profiles for both the service VLAN level (logical interfaces) and the customer VLAN (VLAN tag) level.

```
[edit class-of-service traffic-control-profiles]
tcp-500m-shaping-rate {
    shaping-rate 500m;
}
tcp-svlan0 {
    shaping-rate 200m;
    guaranteed-rate 100m;
    delay-buffer-rate 300m; # This parameter is not shown in the figure.
}
tcp-svlan1 {
    shaping-rate 400m;
    guaranteed-rate 300m;
    delay-buffer-rate 100m; # This parameter is not shown in the figure.
}
tcp-cvlan0 {
    shaping-rate 100m;
    guaranteed-rate 60m;
    scheduler-map tcp-map-cvlan0; # Applies scheduler maps to customer VLANs.
}
tcp-cvlan1 {
    shaping-rate 100m;
    guaranteed-rate 40m;
    scheduler-map tcp-map-cvlan1; # Applies scheduler maps to customer VLANs.
}
tcp-cvlan2 {
    shaping-rate 200m;
    guaranteed-rate 100m;
    scheduler-map tcp-map-cvlanx; # Applies scheduler maps to customer VLANs.
}
tcp-cvlan3 {
    shaping-rate 200m;
    guaranteed-rate 150m;
    scheduler-map tcp-map-cvlanx; # Applies scheduler maps to customer VLANs
}
tcp-cvlan4 {
    shaping-rate 100m;
    guaranteed-rate 50m;
    scheduler-map tcp-map-cvlanx; # Applies scheduler maps to customer VLANs
}
```

Configuring the Schedulers

The schedulers hold the information about the queues, the last level of the hierarchy. Note the consistent naming schemes applied to repetitive elements in all parts of this example.

```
[edit class-of-service schedulers]
sched-cvlan0-qx {
    priority low;
    transmit-rate 20m;
    buffer-size temporal 100ms;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}
sched-cvlan1-q0 {
    priority high;
    transmit-rate 20m;
    buffer-size percent 40;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}
sched-cvlanx-qx {
    transmit-rate percent 30;
    buffer-size percent 30;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}
sched-cvlan1-qx {
    transmit-rate 10m;
    buffer-size temporal 100ms;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}
```

Configuring the Drop Profiles

This section configures the drop profiles for the example. For more information about interpolated drop profiles, see "RED Drop Profiles for Congestion Management" on page 462.

```
[edit class-of-service drop-profiles]
dp-low {
    interpolate fill-level 80 drop-probability 80;
    interpolate fill-level 100 drop-probability 100;
}
dp-high {
    interpolate fill-level 60 drop-probability 80;
    interpolate fill-level 80 drop-probability 100;
}
```

Configuring the Scheduler Maps

This section configures the scheduler maps for the example. Each one references a scheduler configured in "Configuring the Schedulers" on page 452.

```
[edit class-of-service scheduler-maps]
tcp-map-cvlan0 {
    forwarding-class voice scheduler sched-cvlan0-qx;
    forwarding-class video scheduler sched-cvlan0-qx;
    forwarding-class data scheduler sched-cvlan0-qx;
}
tcp-map-cvlan1 {
    forwarding-class voice scheduler sched-cvlan1-q0;
    forwarding-class video scheduler sched-cvlan1-qx;
    forwarding-class data scheduler sched-cvlan1-qx;
}
tcp-map-cvlanx {
    forwarding-class voice scheduler sched-cvlanx-qx;
    forwarding-class video scheduler sched-cvlanx-qx;
    forwarding-class data scheduler sched-cvlanx-qx;
}
```

Applying the Traffic Control Profiles

This section applies the traffic control profiles to the proper levels of the hierarchy.

NOTE: Although a shaping rate can be applied directly to the physical interface, hierarchical schedulers must use a traffic control profile to hold this parameter.

```
[edit class-of-service interfaces]
ge-1/0/0 {
    output-traffic-control-profile tcp-500m-shaping-rate;
    unit 0 {
        output-traffic-control-profile tcp-cvlan0;
    }
    unit 1 {
        output-traffic-control-profile tcp-cvlan1;
    }
    unit 2 {
        output-traffic-control-profile tcp-cvlan2;
    }
    unit 3 {
        output-traffic-control-profile tcp-cvlan3;
    }
    unit 4 {
        output-traffic-control-profile tcp-cvlan4;
    }
}
interface-set svlan0 {
    output-traffic-control-profile tcp-svlan0;
}
interface-set svlan1 {
    output-traffic-control-profile tcp-svlan1;
}
```

NOTE: You should be careful when using a `show interfaces queue` command that references nonexistent class-of-service logical interfaces. When multiple logical interfaces (units) are not configured under the same interface set or physical interface, but are referenced by a command such as `show interfaces queue ge-10/0/1.12 forwarding-class be` or `show interfaces queue ge-10/0/1.13 forwarding-class be` (where logical units 12 and 13 are not configured as a class-of-service interfaces), these interfaces display the same traffic statistics for each logical interface. In other words, even if there is no traffic passing through a particular unconfigured logical interface, as long as one or more of the other unconfigured logical interfaces under the same interface set or physical interface is passing traffic, this particular logical interface displays statistics counters showing the total amount of traffic passed through all other unconfigured logical interfaces together.

Hierarchical Class of Service for Network Slicing

IN THIS SECTION

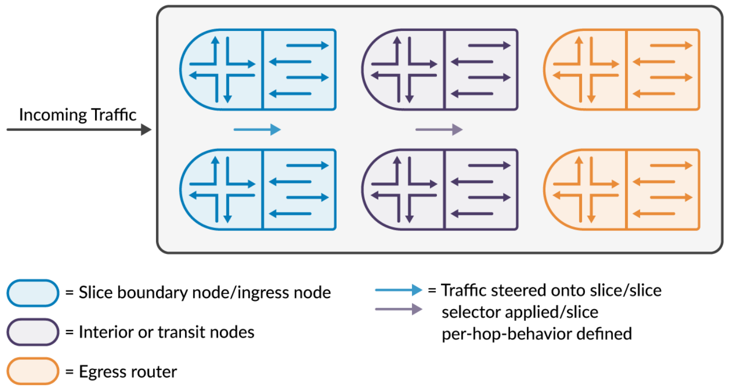
Understanding Network Slicing

Network slicing is the partitioning of a physical network into multiple logical networks. Each logical network is called a slice. On virtue of being a logical network, a slice is a designated set of network resources, such as interfaces, firewall filters, policers, virtual output queues, schedulers, shapers, traffic control profiles etc. to carry traffic.

Slice Domain

A set of connected physical nodes such as routers and switches (along with their links) that participate in network slicing is called a slice domain. The slice domain has ingress nodes, transit nodes, and egress nodes. Ingress and egress nodes are located at the borders of the slice domain. The ingress nodes receive traffic into the domain and may classify them before forwarding them to the transit nodes. The egress nodes forward traffic out of the slice domain, and before doing so, may classify the packets.

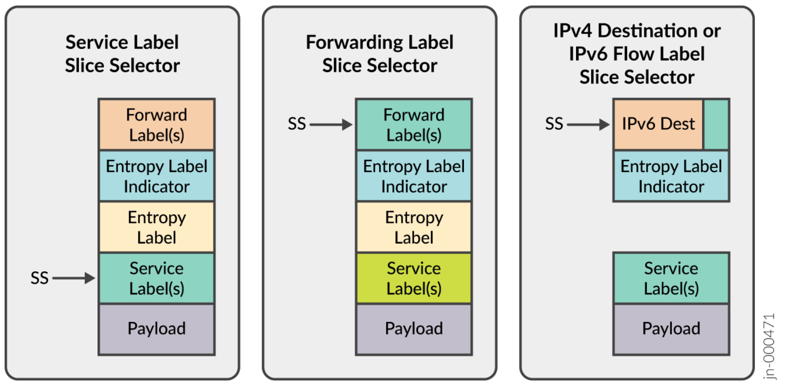
Figure 39: Slice Domain


Incoming Traffic
= Slice boundary node/ingress node
= Interior or transit nodes
= Egress router
= Traffic steered onto slice/slice selector applied/slice per-hop-behavior defined

Slice Selector

By definition, a slice selector is information in the packet header. The information is used by the boundary nodes and/or transit nodes of a slice domain to classify and/or process packets. There are various options to encapsulate/identify/designate a slice selector in the packet. As an example, a Service Label in the packet header can be used as a slice selector. If defined, this label is allotted a position in the packet header and is checked at this position by firewall filters to determine/designate slices. Similarly, there are several other options as depicted in the following figure.

Figure 40: Slice Selector Types



Workflow for Creating Slices

Slices as entities are created by specifying them under network slicing hierarchy under services. Then these slices are used to steer packets and to manage traffic destined to slices.

Hierarchical class of service for slices

You define a traffic control profile for a slice under a physical or aggregated Ethernet interface. Note that you can define traffic control profiles for multiple slices under a physical or aggregated Ethernet interface. See *slice (CoS Interfaces)*.

See "Hierarchical Class of Service (CoS) Queuing for Slice Per-hop-behavior" on page 459 to understand how slices (as part of a hierarchy) are used to control traffic.

Packet steering

Packet steering is the process of marking/matching packets to/from slices. Packets can be steered using firewall filters (firewall steering) and/or routing policy (route steering).

Firewall steering

- A firewall filter can be used at the ingress node to mark matched packets as belonging to slices using the “slice” action. See *slice (firewall filter action)*.
- A firewall filter can also be used at the transit node to match slice packets using the “slice” match condition. See *slice (firewall filter match condition)*. The packet can then be marked to another slice if required by the firewall filter or a policer applied to this packet etc.
- Packets that are not marked/matched to to/from slices are processed as non-slice traffic.

Route steering

- An export policy can be used at the ingress and/or transit node to mark matched routes as belonging to slices. See *slice (export routing policy action)*.
- The export policy can also attach a firewall filter to the route. The firewall filter is used to typically apply a policer to the packets matching the route on the ingress side. This firewall filter is not attached to any interface. Rather, it is part of the forwarding information of the route. See *filter (export routing policy action)*.
- The slice and/or firewall filter will be part of the route's next-hop forwarding information. See show route extensive expanded-nh to view slices and/or firewall filters attached to routes.
- Packets that do not match routes with slice information, are classified as non-slice traffic. Packets that match routes with no slice information, are also classified as non-slice traffic.

To summarize, the following are the configurations that are to be enabled before creating slices, can be used to create slices, or manage packets belonging to slices.

- Specify the slices under network slicing hierarchy under services- – Refer to *network-slicing*.
- Enable enhanced-ip mode – Refer *network-services*.
- Perform class of service configuration to enable a slice under an interface and also apply an output traffic control profile for the slice – See *slice (CoS Interfaces)*. See "Hierarchical Class of Service (CoS) Queuing for Slice Per-hop-behavior" on page 459 to understand how slices (as part of a hierarchy) are used to control traffic.
- Configure firewall filters to steer routes to slices and/or match routes from slices – See *slice (firewall filter match condition)* and *slice (firewall filter action)*.
- Use routing policy to steer routes to slices and/or attach firewall – See *slice (export routing policy action)* and *filter (export routing policy action)*.
- View slices and/or firewall filters attached to routes. See *show route extensive <route> expanded-nh*.
- View slices attached to the forwarding table. See *show class-of-service forwarding-table slice*.
- Show mapping of traffic control profiles to slices. See *show class-of-service forwarding-table slice mapping*.
- View traffic control profile(s) attached to a slice under an interface. See *show class-of-service slice <slice_name> interface <interface_name>*.
- View statistics for a slice. See *show cos halp flow queue-stats*.

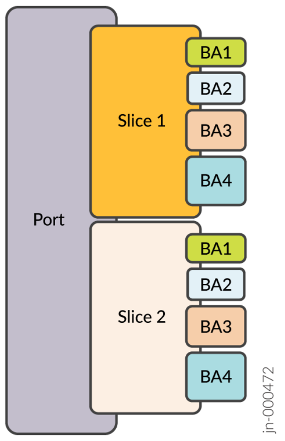
Hierarchical Class of Service (CoS) Queuing for Slice Per-hop-behavior

In hierarchical CoS, packets are classified at various levels. It could be at the port level, followed by the logical unit level, and then at the queue level. This means that packets are passing through a hierarchy. At every stage of the hierarchy, packets are being classified, shaped, scheduled etc.

In the context of network slicing, a slice also becomes part of the hierarchy. Shapers, schedulers, and traffic control profiles can be applied to the slice. Just as queues are made available to logical units, queues are made available to slices as well.

As the following figure shows, the queues (labeled BA1, BA2, BA3, BA4) are made available to the slice. The queues map to forwarding classes (FCs). Based on behavioral aggregate classifiers, packets are classified into FCs, and subsequently into a corresponding queue (BA1 or BA2 or BA3 etc.).

Figure 41: Hierarchical Queuing



As "Workflow for Creating Slices" on page 457 describes, you configure network slicing using a combination of firewall filters and Class of Service (CoS) configuration. See *slice (CoS Interfaces)* to read on how to enable slice(s) under any interface using CoS configuration.

Configuring Ingress Hierarchical CoS

You can configure ingress CoS parameters, including hierarchical schedulers, on devices with Enhanced Queuing DPCs (that is, line cards that have a queuing chip). In general, the supported configuration statements apply to per-unit schedulers or to hierarchical schedulers.

NOTE: Ingress CoS is not supported on line cards that do not contain a queuing chip.

To configure ingress CoS for per-unit schedulers, include the following statements at the `[edit class-of-service interfaces *interface-name*]` hierarchy level:

NOTE: The `input-scheduler-map` and `input-traffic-control-profile` statements are mutually exclusive at the same hierarchy level.

```
[edit class-of-service interfaces interface-name]
input-excess-bandwidth-share (proportional value | equal);
input-scheduler-map map-name;
input-shaping-rate rate;
input-traffic-control-profile profile-name shared-instance instance-name;
unit logical-unit-number;
    input-scheduler-map map-name;
    input-shaping-rate (percent percentage | rate);
    input-traffic-control-profile profile-name shared-instance instance-name;
}
```

To configure ingress CoS for hierarchical schedulers, include the `interface-set *interface-set-name*` statement at the `[edit class-of-service interfaces]` hierarchy level:

```
[edit class-of-service interfaces]
interface-set interface-set-name {
    input-excess-bandwidth-share (proportional value | equal);
    input-traffic-control-profile profile-name shared-instance instance-name;
    input-traffic-control-profile-remaining profile-name;
    interface interface-name {
        input-excess-bandwidth-share (proportional value | equal);
        input-traffic-control-profile profile-name shared-instance instance-name;
        input-traffic-control-profile-remaining profile-name;
        unit logical-unit-number;
            input-traffic-control-profile profile-name shared-instance instance-name;
        }
    }
}
```

For many platforms, CoS queuing and scheduling are enabled on the egress side but disabled on the ingress side by default. To enable ingress CoS, configure the `traffic-manager` statement with `ingress-and-egress` mode:

```
[edit chassis fpc slot-number pic pic-number]
traffic-manager mode ingress-and-egress;
```

NOTE: If you enable ingress CoS settings and inline services on the same FPC, the FPC moves to the offline state. This behavior is expected because null route detection is triggered that causes the FPC to move to the offline state.

Configured CoS features on the ingress are independent of CoS features on the egress, with the following exceptions:

- If you configure a per-unit or hierarchical scheduler at the `[edit class-of-service interfaces]` hierarchy level, the schedulers apply in both the ingress and egress directions.
- You cannot configure the same logical interface on an ingress and an egress interface set. A logical interface can only belong to one interface set.
- The DPC's frame buffer is shared between ingress and egress configurations.

The following behavior aggregate (BA) classification tables are supported on the ingress side:

- inet-precedence
- DSCP
- exp (MPLS)
- DSCP for IPv6
- IEEE 802.1p

RELATED DOCUMENTATION

Configuring Traffic Control Profiles for Shared Scheduling and Shaping | 350

Enhanced Queuing DPC CoS Properties | 916

CHAPTER 9

Controlling Congestion with Scheduler RED Drop Profiles, Buffers, PFC, and ECN

IN THIS CHAPTER

RED Drop Profiles for Congestion Management

SUMMARY

This topic describes the use and configuration of *random early detection (RED)* drop profiles for congestion management. A *drop profile* is a mechanism of RED that defines parameters that

IN THIS SECTION

allow packets to be dropped from a queue based on how full the queue is. Drop profiles define the meanings of the packet loss priorities.

Manage Congestion with RED Drop Profiles and Packet Loss Priorities

You can configure two parameters to control congestion in each output queue. One parameter, *delay-buffer bandwidth*, enables queue growth to absorb burst traffic up to the specified product of delay-buffer time and output rate. Once the specified delay buffer becomes full, packets with 100 percent drop probability are dropped from the tail of the queue. For more information, see "Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 476.

The other parameter, which this topic covers, defines the *drop probabilities* across the range of delay-buffer occupancy, supporting the *RED process*. When the number of packets queued is greater than the ability of the router or switch to empty a queue, the queue requires a method for determining which packets to drop from the network. To address this, you can enable RED on individual queues.

Depending on the drop probabilities, RED might drop many packets long before the buffer becomes full, or it might drop only a few packets even if the buffer is almost full.

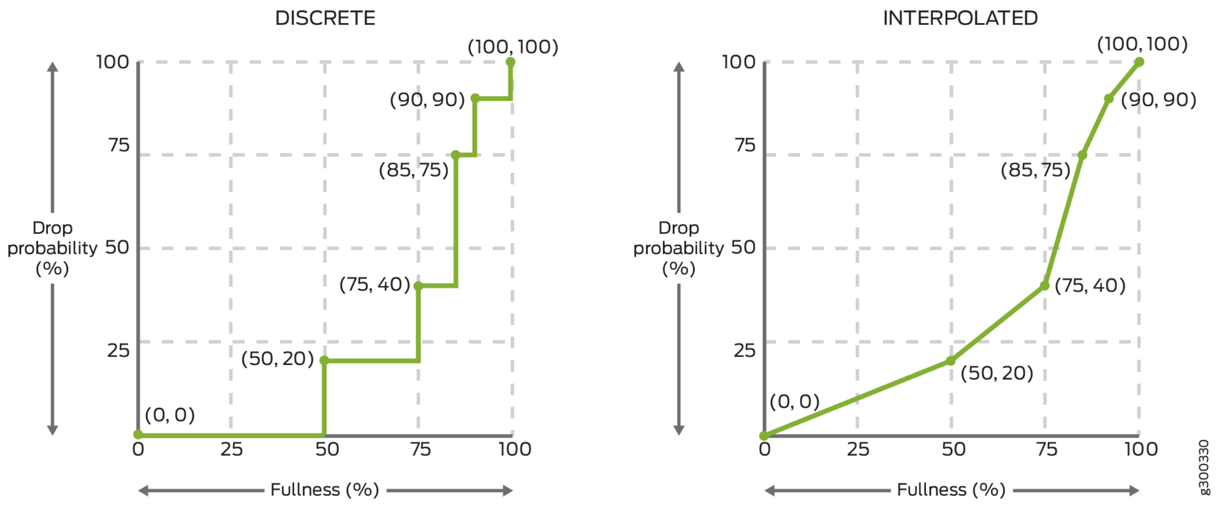
A *drop profile* is a mechanism of RED that defines parameters that allow packets to be dropped from the network. Drop profiles define the meanings of the packet loss priorities.

When you configure drop profiles, there are two important values:

- *queue fullness* represents a percentage of the memory used to store packets in relation to the total amount that has been allocated for a specific queue.
- *drop probability* is a percentage value that correlates to the likelihood that an individual packet is dropped from the network.

How these two variables function is illustrated in graph format. Figure 42 on page 464 shows both a discrete and an interpolated graph. Although the formation of these graph lines is different, the application of the profile is the same. When a packet joins the tail of the queue, a random number from 0 to 100 is calculated by the router or switch. This random number is plotted against the drop profile using the current queue fullness of that particular queue. When the random number falls above the graph line, the packet is transmitted onto the physical media. When the number falls below the graph line, the packet is dropped from the network.

Figure 42: Discrete and Interpolated Drop Profiles



You create drop profiles by defining multiple fill levels and drop probabilities.

To create the discrete profile graph as shown in Figure 42 on page 464 on the left, the software begins at the bottom-left corner, representing a 0-percent fill level and a 0-percent drop probability. This configuration creates a line horizontally to the right on the fullness level (l) x-axis until it reaches the first defined fill level, 50-percent for this configuration, which is designated to have a drop probability (p) of 20-percent. The software then continues the line horizontally along the fill level until the next drop probability is reached at the designated data point of 75-percent fill level, which has a designated drop-probability of 40-percent. The line is then continued horizontally to the next fill level of 85-percent and the designated drop probability of 75-percent. The line continues horizontally to the next designated fill level of 90-percent, which has a designated drop probability of 90-percent, and a line is created to data point 90-percent (l), 90-percent (p) (l90 p90). From the l90 p90 point, the line continues horizontally to the 100-percent fill level, which has a drop probability of 100 percent, at which the line rises to the end-point of 100-100, which is 100 percent fill level with a 100 percent drop probability.

If you specify an interpolated drop profile, in the first quadrant the initial line segment spans from the origin (0,0) to the next defined point. From that defined fill-level/drop-probability point, a second line runs to the next point, and so forth, until a final line segment connects (100, 100). The software automatically constructs a drop profile containing 64 fill levels at drop probabilities that approximate the calculated line segments.

NOTE: For consistent behavior across router families, include the pair (100, 100) in the drop profile configuration.

You can create a smoother graph line by configuring the profile with the `interpolate` statement. This enables the software to automatically generate 64 data points on the graph beginning at (0, 0) and ending at (100, 100). Along the way, the graph line intersects specific data points that you have defined.

NOTE: If you configure the `interpolate` statement, you can specify more than 64 pairs, but the system generates only 64 discrete entries.

Loss priorities allow you to set the priority of dropping a packet. Loss priority affects the scheduling of a packet without affecting the packet's relative ordering. You can use the packet loss priority (PLP) bit as part of a congestion control strategy. You can use the loss priority setting to identify packets that have experienced congestion. Typically you mark packets exceeding some service level with a high loss priority. You set loss priority by configuring a classifier or a policer. The loss priority is used later in the workflow to select one of the drop profiles used by RED.

You specify drop probabilities in the drop profile section of the class-of-service (CoS) configuration hierarchy and map them to corresponding loss priorities in each scheduler configuration. For each scheduler, you can configure multiple separate drop profiles, one for each combination of loss priority.

You can configure a maximum of 32 different drop profiles.

Use Feature Explorer to confirm platform and release support for specific features.

Review the "Platform-Specific RED Drop Profile Behavior" on page 467 section for notes related to your platform.

Configure RED Drop Profiles to Define Packet Drop or ECN Behaviors

You enable RED by applying a drop profile to a scheduler. When RED is operational on an interface, the queue no longer drops all excess packets at the tail of the queue. Rather, a controlled fraction of packets are dropped, or marked with ECN (if enabled). Some output-buffered routers perform RED drops of oldest packets at the head of the queue. Other routers perform RED as packets enter a queue. When a queue becomes full, tail-drops (100%) supersede random dropping.

To configure RED drop profiles, include the following statements at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
drop-profiles {
    profile-name {
        fill-level

               percentage

               drop-probability

               percentage;
        interpolate {
                drop-probability [ values ];
                fill-level [ values ];
            }
        }
    }
```

To configure a drop profile, include either the `interpolate` statement and its options, or the fill-level and drop-probability *percentage* values.

For example, the following shows a discrete configuration and an interpolated configuration that correspond to the graphs in Figure 42 on page 464. The values defined in the configurations are matched to represent the data points in the graph lines.

Create a Discrete Configuration

```
class-of-service {
    drop-profiles {
        discrete-style-profile {
            fill-level 0 drop-probability 0;
            fill-level 50 drop-probability 20;
            fill-level 75 drop-probability 40;
            fill-level 85 drop-probability 75;
            fill-level 90 drop-probability 90;
            fill-level 100 drop-probability 100;
        }
    }
}
```

Create an Interpolated Configuration

```
class-of-service {
    drop-profiles {
        interpolated-style-profile {
            interpolate {
                fill-level [ 0 50 75 85 90 100 ];
                drop-probability [ 0 20 40 75 90 100 ];
            }
        }
    }
}
```

To configure a drop profile:

1. Create the drop profile by specifying a name for it.

   ```
   [edit]
   user@host# edit class-of-service drop-profiles profile-name
   ```

2. (Optional) Specify the fill-level and drop-probability values for the drop profile.

   ```
   [edit class-of-service drop-profiles profile-name]
   user@host# set fill-level percentage drop-probability percentage
   ```

   Repeat this step for each fill-level and drop-probability.

3. (Optional) Specify values for interpolating the relationship between queue fill level and drop probability.

   ```
   [edit class-of-service drop-profiles profile-name]
   user@host#  set interpolate drop-probability percentage
   drop-probabilitypercentage
   ```

4. Verify your configuration.

   ```
   [edit class-of-service drop-profiles]
   user@host#  show
   ```

5. Save your configuration.

   ```
   [edit class-of-service drop-profiles]
   user@host#  commit
   ```

After you configure a drop profile, you must assign the drop profile to a drop-profile map, and assign the drop-profile map to a scheduler, as discussed in "Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers" on page 469.

Platform-Specific RED Drop Profile Behavior

Use Feature Explorer to confirm platform and release support for RED drop profiles.

Use the following table to review platform-specific behaviors for your platform:

Platform	Difference
ACX5448	• The ACX5448 router does not support interpolate drop-probability and supports only the discrete method. • The ACX5448 router supports configuring drop profiles for `loss-priority` *low*, *medium-high* and *high*. • You can specify two fill levels in each drop profile map. The drop probability you associate with the lowest fill level must be 0.
ACX7000 Series	• ACX7000 Series routers do not support interpolate drop-probability and support only the discrete method. • ACX7000 Series routers support configuring drop profiles for `loss-priority` *low*, *medium-high* and *high* . • You can specify two fill levels in each drop profile map. The drop probability you associate with the lowest fill level must be 0.
MX Series	• Physical or logical interfaces hosted on MICs in Queuing or Enhanced Queuing MPCs for MX Series routers support up to 64 (fill level, drop probability) pairs per discrete or interpolated drop profile. • Physical or logical interfaces hosted on Enhanced Queuing DPCs for MX Series routers support up to 64 (fill level, drop probability) pairs per discrete drop profile or 2 pairs per interpolated drop profile. For more information, see "Configuring WRED on Enhanced Queuing DPCs" on page 921. • Physical or logical interfaces hosted on IQ2 PICs or IQE PICs support up to two (fill level, drop probability) pairs per discrete or interpolated drop profile.

(Continued)

Platform	Difference
PTX Series	• PTX Series routers running Junos OS and PTX10003 routers running Junos OS Evolved support up to 64 (fill level, drop probability) pairs per discrete or interpolated drop profile. • Other PTX Series routers running Junos OS Evolved support up to two (fill level, drop probability) pairs per interpolated drop profile. A single-pair drop profile is discrete. Two pairs are interpolated.

RELATED DOCUMENTATION

drop-probability (Interpolated Value)

drop-probability (Percentage)

Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers

RED drop profiles take action on outgoing packets. When tricolor marking is enabled, Junos devices support four drop-profile map PLP designations: `low`, `medium-low`, `medium-high`, and `high`.

Drop-profile maps associate RED drop profiles with a scheduler. The map examines the current loss priority setting of the packet (`low`, `medium-low`, `medium-high`, or `high`) and assigns a drop profile according to these values. For example, you can specify that all TCP packets with `low` loss priority are assigned a drop profile that you name `low-drop`. You can associate multiple drop-profile maps with a single queue.

The scheduler drop profile defines the drop probabilities across the range of delay-buffer occupancy, thereby supporting the RED process. Depending on the drop probabilities, RED might drop packets aggressively long before the buffer becomes full, or it might drop only a few packets even if the buffer is almost full. For information on how to configure drop profiles, see "RED Drop Profiles for Congestion Management" on page 462.

By default, the drop profile is mapped to packets with low PLP and any protocol type.

When you configure TCM, the drop-profile map's protocol type must be `any`.

The map sets the drop profile for a specific PLP and protocol type. The inputs for the map are the PLP and the protocol type. The output is the drop profile. In other words, the map sets the drop profile for each packet with a specific PLP and protocol type exiting the interface. For more information about how CoS maps work, see "Mapping CoS Component Inputs to Outputs" on page 9.

NOTE: You can configure only the `any` option for the `protocol` statement.
For each scheduler, you can configure separate drop profile maps for each loss priority.

You can configure a maximum of 32 different drop profiles.

In the following sample configuration, the `dp` drop profile is assigned to all packets exiting the interface with a medium-low PLP and belonging to any protocol:To configure this drop profile map:

1. Specify the name of the scheduler.

```
[edit]
user@host# edit class-of-service schedulers af
```

2. Define the loss-priority value for a drop profile, the protocol type, and the name of the drop profile..

```
[[edit class-of-service schedulers af]
user@host#  set drop-profile-map loss-priority medium-low protocol any drop-profile dp
```

3. Verify your configuration.

```
[edit class-of-service]
user@host#  show schedulers af
```

```
drop-profile-map loss-priority medium-low protocol any drop-profile dp;
```

4. Save your configuration.

```
[edit class-of-service]
user@host#  commit
```

NOTE: To use this drop-profile map, you must configure the settings for the `dp` drop profile at the `[edit class-of-service drop-profiles dp]` hierarchy level..

RELATED DOCUMENTATION

RED Drop Profiles for Congestion Management | 462

Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows

By default, the least significant bit of the CoS value sets the packet loss priority (PLP) value. For example, CoS value 000 is associated with PLP `low`, and CoS value 001 is associated with PLP `high`. In general, you can change the PLP by configuring a behavior aggregate (BA) or multifield classifier, as discussed in "Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic" on page 60 and "Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields" on page 148.

However, on devices that do not have tricolor marking enabled, the loss priority can be configured by setting the PLP within a multifield classifier or by behavior aggregate (BA) classifier. This setting can then be used by the appropriate drop profile map and rewrite rule.

The following example shows a two-step procedure to override the default PLP settings.

The first part of this example specifies that while the DSCP code points are 110, the loss priority is set to `high`; however, overriding the default PLP this way has no effect.

1. Configure the classifier name and specify it as type as DSCP.

```
[edit]
user@host# edit class-of-service classifiers dscp ba-classifier
```

2. Specify the forwarding class

```
[edit class-of-service classifiers dscp ba-classifier]
user@host# set forwarding-class expedited-forwarding loss-priority high code-points 110
```

Use the following procedure to configure a multifield classifier that sets the PLP.

1. Under the `firewall` statement, specify a name for the filter.

   ```
   edit
   user@host# edit firewall filter ef-filter
   ```

2. Specify the term name and match criteria you want to look for in incoming packets.

   ```
   [edit firewall filter ef-filter]
   user@host# set term ef-multifield from precedence 6
   ```

3. Specify the action you want to take when a packet matches the conditions.

   ```
   [edit firewall filter ef-filter]
   user@host# set term ef-multifield then loss-priority high forwarding-class expedited-
   forwarding
   ```

4. Verify your configuration.

   ```
   [edit firewall]
   user@host# show
   ```

   ```
   filter ef-filter {
       term ef-multifield {
           from {
               precedence 6;
           }
           then {
               loss-priority high;
               forwarding-class expeditd-forwarding;
           }
       }
   }
   ```

5. Save your configuration.

```
[edit firewall]
user@host# commit
```

RELATED DOCUMENTATION

Mapping PLP to RED Drop Profiles | 473

RED Drop Profiles for Congestion Management | 462

Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers | 469

Configuring Schedulers | 348

Mapping PLP to RED Drop Profiles

Loss priority settings help determine which packets are dropped from the network during periods of congestion. The software supports multiple packet loss priority (PLP) designations: `low` and `high`. (In addition, `medium-low` and `medium-high` PLPs are supported when you configure tricolor marking.) You can set PLP by configuring a behavior aggregate or multifield classifier.

A drop-profile map examines the loss priority setting of an outgoing packet: `high`, `medium-high`, `medium-low`, `low`, or any.

Obviously, *low*, *medium-low*, *medium-high*, and *high* are relative terms, which by themselves have no meaning. Drop profiles define the meanings of the loss priorities. In the following example, the `low-drop` drop profile defines the meaning of `low` PLP as a 10 percent drop probability when the fill level is 75 percent and a 40 percent drop probability when the fill level is 95 percent. The `high-drop` drop profile defines the meaning of `high` PLP as a 50 percent drop probability when the fill level is 25 percent and a 90 percent drop probability when the fill level is 50 percent.

The following example procedure, configures a scheduler that includes two drop-profile maps, which specify that packets are evaluated by the `low-drop` drop profile if they have a `low` loss priority and are from any protocol. Packets are evaluated by the `high-drop` drop profile if they have a `high` loss priority and are from any protocol.

1. Create the low drop profile.

```
[edit]
user@host# edit class-of-service drop-profiles low-drop
```

2. Specify values for interpolating the relationship between the queue fill level and drop probability for the low drop profile.

```
[edit class-of-service drop-profiles low-drop]
user@host# edit interpolate
user@host# set drop-probability [10 40]
user@host# set fill-level [75 95]
```

3. Crate the high drop profile.

```
[edit class-of-service drop-profiles]
user@host# edit high-drop
```

4. Specify values for interpolating the relationship between the queue fill level and drop probability for the high drop profile.

```
[edit class-of-service drop-profiles high-drop]
user@host# edit interpolate
user@host# set drop-probability [50 90]
user@host# set fill-level [25 50]
```

5. Specify the scheduler name.

```
[edit class-of-service]
user@host# edit schedulers best effort
```

6. Define the loss-priority for each low drop profile.

```
[edit class-of-service schedulers best-effort]
user@host# set drop-profile-map loss-priority low protocol any drop-profile low-drop
```

7. Define the loss-priority for each high drop profile.

```
[edit class-of-service schedulers best-effort]
user@host# set drop-profile-map loss-priority high protocol any drop-profile high-drop
```

8. Verify your configuration.

```
[edit class-of-service]
user@host# show
```

```
drop-profiles {
    low-drop {
        interpolate {
            fill-level [ 75 95 ];
            drop-probability [ 10 40 ];
        }
    }
    high-drop {
        interpolate {
            fill-level [ 25 50 ];
            drop-probability [ 50 90 ];
        }
    }
}

schedulers {
    best-effort {
        drop-profile-map loss-priority low protocol any drop-profile low-drop;
        drop-profile-map loss-priority high protocol any drop-profile high-drop;
    }
}
```

9. Save your configuration.

```
[edit class-of-service]
user@host# commit
```

RELATED DOCUMENTATION

Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows | 471

RED Drop Profiles for Congestion Management | 462

Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers | 469

Configuring Schedulers | 348

Configure the Scheduler Buffer Size to Manage Egress Congestion

IN THIS SECTION

Traffic bursts are common and can oversubscribe a queue or port for a few milliseconds or less, leading to the drop of critical packets. Configurable deep buffer enables queues to handle bursty traffic and buffer critical packets for short periods of time, mitigating the loss of critical packets.
To control congestion at the output stage, you can configure the delay-buffer bandwidth. The delay-buffer bandwidth provides packet buffer space to absorb burst traffic up to the specified duration of delay. Once the specified delay buffer becomes full, packets with 100 percent drop probability are dropped from the head of the buffer.

The default scheduler transmission rate for queues 0 through 7 are 95, 0, 0, 0, 0, 0, 0, and 5 percent of the total available bandwidth.

The default buffer size percentages for queues 0 through 7 are 95, 0, 0, 0, 0, 0, 0, and 5 percent of the total available buffer. The total available buffer per queue differs by PIC type.

To configure the buffer size, include the `buffer-size` statement at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
buffer-size (percent percentage | remainder | shared | temporal microseconds);
```

For each scheduler, you can configure the buffer size as one of the following:

- A percentage of the total buffer. The total buffer per queue is based on microseconds and differs by routing device type.
- The remaining buffer available. The remainder is the buffer percentage that is not assigned to other queues.
- Shared from the interface's buffer pool. On PTX Series routers, set a queue's buffer to be up to 100 percent of the interface's buffer. This option allows the queue's buffer to grow as large as 100 percent of the interface's buffer if and only if it is the only active queue for the interface.
- A temporal value, in microseconds. For the temporal setting, the queuing algorithm starts dropping packets when it queues more than a computed number of bytes. This maximum is computed by multiplying the transmission rate of the queue by the configured temporal value.

NOTE: In general, the default temporal buffer value is inversely related to the speed, or shaping rate, of the interface. As the speed of the interface increases, the interface needs less and less buffer to hold data, as it is possible for the interface to send more and more data.

Because available hardware resources to configure temporal buffer settings are limited, we recommend having a common temporal buffer configuration as much as possible.

In general, you can configure the scheduler buffer size for both port schedulers and hierarchical schedulers. For port schedulers, Junos calculates the shared buffer size based on the following:

- If a port shaper is configured, Junos calculates the buffer size based on the port shaper.
- If no port shaper is configured, Junos calculates the buffer size based on the port speed.

For hierarchical schedulers, Junos calculates the shared buffer size based on the following:

- If a shaper is configured on the logical interface, Junos calculates the buffer size based on the logical interface port shaper.
- If a port shaper is configure, but no logical interface shaper, Junos calculates the buffer size based on the port shaper.
- If no shaper is configured, Junos calculates the buffer size based on the port speed.

Example: Configure the Delay Buffer Value for a Scheduler

You can assign to a physical or logical interface a scheduler map that consists of different schedulers (or queues). The physical interface's large delay buffer can be distributed to the different schedulers (or queues) using the `transmit-rate` and `buffer-size` statements at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level.

This example shows two schedulers, `sched-best` and `sched-exped`, with the delay buffer size configured as a percentage (20 percent) and temporal value (300,000 microseconds), respectively. The `sched-best` scheduler has a transmit rate of 10 percent. The `sched-exped` scheduler has a transmit rate of 20 percent.

The `sched-best` scheduler's delay buffer is twice the specified transmit rate of 10 percent. Assuming that the `sched-best` scheduler is assigned to an interface with 500,000 microseconds of delay buffer, this scheduler receives 20 percent of the total 500,000 microseconds of the interface's delay buffer. Therefore, the scheduler receives 18,750 bytes of delay buffer:

```
available interface bandwidth * configured percentage buffer-size * maximum buffer = queue buffer

1.5 Mbps * 0.2 * 500,000 microseconds = 150,000 bits = 18,750 bytes
```

Assuming that the `sched-exped` scheduler is assigned to an interface with 500,000 microseconds of delay buffer, this scheduler receives 300,000 microseconds of the interface's 500,000-microsecond delay buffer with the traffic rate at 20 percent. Therefore, the scheduler receives 11,250 bytes of delay buffer:

```
available interface bandwidth * configured percentage transmit-rate
 * configured temporal buffer-size = queue buffer

1.5 Mbps * 0.2 * 300,000 microseconds = 90,000 bits = 11,250 bytes
```

To configure this example:

1. Configure the `sched-best` scheduler.

   ```
   [edit]
   user@host#  edit class-of-service schedulers sched-best
   ```

2. Specify the transmit-rate of 10 percent.

   ```
   [edit class-of-service schedulers sched-best]
   user@host# set transmit-rate percent 10
   ```

3. Specify the buffer size as 20 percent.

   ```
   [edit class-of-service schedulers sched-best]
   user@host#  set buffer-size percent 20
   ```

4. Configure the `sched-exped` scheduler.

```
[edit]
user@host# up
[edit class-of-service schedulers]
user@host# edit sched-exped
```

5. Specify the transmit-rate of 20 percent.

```
[edit class-of-service schedulers sched-exped]
user@host# set transmit-rate percent 20
```

6. Specify the buffer size temporal value (300,000 microseconds).

```
[edit class-of-service schedulers sched-exped]
user@host#  set buffer-size temporal 300000
```

7. Verify the configuration.

```
[edit]
user@host# show class-of-service
```

```
schedulers {
    sched-best {
        transmit-rate percent 10;
        buffer-size percent 20;
    }
    sched-exped {
        transmit-rate percent 20;
        buffer-size temporal 300k;
    }
}
```

8. Save the configuration.

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

Example: Configure the Physical Interface Shaping Rate

In general, the physical interface speed is the basis for calculating the delay buffer size. However, when you include the `shaping-rate` statement, the shaping rate becomes the basis for calculating the delay buffer size.

This example configures the shaping rate on an interface to 200 Kbps, which means that the interface bandwidth is set to 200 Kbps instead of the full line rate. In this example, this interface receives 4 seconds of delay buffer, or 800 Kbps of traffic, which is 800 KB for a full second.

1. Specify the interface on which you want to configure the shaping rate.

```
[edit]
user@host#  edit class-of-service interfaces xe-0/0/0:1:1
```

2. Specify the shaping rate.

```
[edit class-of-service interfaces xe-0/0/0:1:1]
user@host# set shaping-rate 200k
```

3. Verify the configuration.

```
[edit class-of-service]
user@host# show
```

```
interfaces {
    xe-0/0/0:1:1 {
        shaping-rate 200k;
    }
}
```

4. Save the configuration.

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

Enabling and Disabling the Memory Allocation Dynamic per Queue

In the Junos OS, the memory allocation dynamic (MAD) is a mechanism that dynamically provisions extra delay buffer when a queue is using more bandwidth than it is allocated in the transmit rate setting. With this extra buffer, queues absorb traffic bursts more easily, thus avoiding packet drops. The MAD mechanism can provision extra delay buffer only when extra transmission bandwidth is being used by a queue. This means that the queue might have packet drops if there is no surplus transmission bandwidth available.

The MAD mechanism is particularly useful for forwarding classes carrying latency-immune traffic for which the primary requirement is maximum bandwidth utilization. In contrast, for latency-sensitive traffic, you might wish to disable the MAD mechanism because large delay buffers are not optimum.

MAD support is dependent on the FPC and Packet Forwarding Engine, not the PIC.

To enable the MAD mechanism on supported hardware:

Include the `buffer-size percent` statement at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
user@host# set buffer-size percent percentage
```

The minimum buffer allocated to any queue is 18,432 bytes. If a queue is configured to have a buffer size less than 18,000, the queue retains a buffer size of 18,432 bytes.

If desired, you can configure a buffer size that is greater than the configured transmission rate. The buffer can accommodate packet bursts that exceed the configured transmission rate, if sufficient excess bandwidth is available. For example:

```
class-of-service {
    schedulers {
        sched-best {
            transmit-rate percent 20;
            buffer-size percent 30;
        }
    }
}
```

As stated previously, you can use a temporal delay buffer configuration to disable the MAD mechanism on a queue, thus limiting the size of the delay buffer. However, the effective buffer latency for a temporal queue is bounded not only by the buffer size value but also by the associated drop profile. If a drop profile specifies a drop probability of 100 percent at a fill-level less than 100 percent, the effective maximum buffer latency is smaller than the buffer size setting. This is because the drop profile specifies that the queue drop packets before the queue's delay buffer is 100 percent full.

Such a configuration might look like the following example:

```
class-of-service {
    drop-profiles {
        plp-high {
            fill-level 70 drop-probability 100;
        }
        plp-low {
            fill-level 80 drop-probability 100;
        }
    }
    schedulers {
        sched {
            buffer-size temporal 500000;
            drop-profile-map loss-priority low protocol any drop-profile plp-low;
            drop-profile-map loss-priority high protocol any drop-profile plp-high;
            transmit-rate percent 20;
        }
    }
}
```

Platform-Specific Buffer Size Configuration Behavior

Use Feature Explorer to confirm platform and release support for buffer size configuration.

Use the following table to review platform-specific behaviors for your platform:

Platform	Difference
ACX7000 Series routers	For the `buffer-size`, use the `temporal` option to configure the shared buffer. Use the `percent` and `remainder` options to configure the guaranteed buffer.
EX Series switches	The MAD mechanism is enabled unless the delay buffer is configured with a temporal setting for a given queue.

(Continued)

Platform	Difference
MX Series routers	The MAD mechanism is enabled unless the delay buffer is configured with a temporal setting for a given queue.
PTX Series routers	PTX Series routers support the `shared` option for `buffer-size`, which enables you to set a queue's buffer to be up to 100 percent of the interface's buffer. This option allows the queue's buffer to grow as large as 100 percent of the interface's buffer if and only if it is the only active queue for the interface.

RELATED DOCUMENTATION

buffer-size (Schedulers)

schedulers (CoS)

q-pic-large-buffer

schedulers (CoS)

Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy

By default, RED is performed based on instantaneous buffer occupancy information. However, PICs can be configured to use *weighted average* buffer occupancy information. This option is configured on a per-PIC basis. If you configure this feature on an unsupported PIC, you see an error message.

When weighted average buffer occupancy is configured, you configure a weight value for averaged buffer occupancy calculations. This weight value is expressed as a negative exponential value of 2 in a fractional expression. For example, a configured weight value of 2 would be expressed as $1/(2^2) = 1/4$. If a configured weight value was configured as 1 (the default), the value would be expressed as $1/(2^1) = 1/2$.

This calculated weight value is applied to the instantaneous buffer occupancy value to determine the new value of the weighted average buffer occupancy. The formula to derive the new weighted average buffer occupancy is:

```
new average buffer occupancy = weight value * instantaneous buffer occupancy + (1 - weight value) * current
average buffer occupancy
```

For example, if the weight exponent value is configured as 3 (giving a weight value of $1/2^3 = 1/8$), the formula used to determine the new average buffer occupancy based on the instant buffer usage is:

```
new average buffer occupancy = 1/8 * instantaneous buffer occupancy + (7/8) * current average buffer occupancy
```

The valid operational range for the weight value on PICs is 0 through 31. A value of 0 results in the average buffer occupancy being the same as the instantaneous buffer occupancy calculations. Values higher than 31 can be configured, but in these cases the current maximum *operational* value of 31 is used for buffer occupancy calculations.

NOTE: The `show interfaces` command with the `extensive` option displays the *configured* value for the `RED buffer occupancy` weight exponent. However, in all such cases, the current *operational* maximum value of 31 is used internally.

To configure weighted average buffer occupancy:

1. Specify the FPC slot number and PIC number on which you want to configure RED weighted average buffer occupancy calculations:

```
[edit]
user@host# edit chassis fpc slot-number pic pic-number red-buffer-occupancy weighted-averaged
```

2. Specify the weight exponent value.

```
[edit chassis fpc slot-number pic pic-number red-buffer-occupancy weighted-averaged]
user@host# set instant-usage-weight-exponent exponent-value
```

3. Verify your configuration.

```
[edit chassis]
user@host#  show
```

   For example:

```
[edit chassis]
user@host# show
fpc 1 {
    pic 1 {
        red-buffer-occupancy {
            weighted-averaged {
                instant-usage-weight-exponent 3;
            }
        }
    }
}
```

4. Save your configuration.

```
[edit chassis]
user@host#  commit
```

RELATED DOCUMENTATION

Example: Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy | 485

red-buffer-occupancy

Example: Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy

IN THIS SECTION

This topic provides two examples for configuring the weighted RED buffer occupancy feature to manage traffic bursts.

Requirements

Weighted RED buffer occupancy is configured on a per-PIC basis. If you configure this feature on an unsupported PIC, you see an error message.

Overview

To manage traffic bursts on PICs, you can base RED queue management on weighted average buffer occupancy values. This topic provides two examples for configuring weighted RED buffer occupancy feature to manage traffic bursts.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, and then copy and paste the commands into the CLI.

To configure a PIC to use a weight value of 1/2 in average buffer occupancy calculations:

```
[edit]
edit chassis fpc 0 pic 1
set red-buffer-occupancy weighted-averaged instant-usage-weight-exponent 1
```

To configure a PIC to use a weight value of 1/4 in average buffer occupancy calculations:

```
[edit]
edit chassis fpc 0 pic 1
set red-buffer-occupancy weighted-averaged instant-usage-weight-exponent 2
```

Example: Configuring a PIC to Use a Weight Value of 1/2 in Average Buffer Occupancy Calculations

Step-by-Step Procedure

To configure a PIC to use a weight value of 1/2 in average buffer occupancy calculations:

1. Specify the PIC.

```
[edit]
user@host# edit chassis fpc 1 pic 0
```

2. Configure the RED queue management values.

```
[edit chassis fpc 1 pic 0]
user@host# set red-buffer-occupancy weighted-averaged instant-usage-weight-exponent 1
```

Example: Configuring a PIC to Use a Weight Value of 1/4 in Average Buffer Occupancy Calculations

Step-by-Step Procedure

To configure a PIC to use a weight value of 1/4 in average buffer occupancy calculations:

1. Specify the PIC.

```
[edit]
user@host# edit chassis fpc 1 pic 1
```

2. Configure the RED queue management values.

```
[edit chassis fpc 1 pic 1]
user@host# set red-buffer-occupancy weighted-averaged instant-usage-weight-exponent 2
```

Results

From configuration mode, confirm your configuration by entering the `show` command at the `[edit chassis fpc 1]` hierarchy level. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit chassis fpc 1]
user@host# show
```

```


red-buffer-occupancy {
    weighted-averaged {
        instant-usage-weight-exponent 1;
    }
}
red-buffer-occupancy {
    weighted-averaged {
        instant-usage-weight-exponent 2;
    }
}
```

Enter `commit` from configuration mode.

RELATED DOCUMENTATION

Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy | 483

red-buffer-occupancy

PFC Functionality Across L3 Interfaces

Priority-based flow control (PFC) allows you to select traffic flows within a link and pause these flows, so that the output queues associated with the flows do not overflow and drop packets. PFC is more granular than Ethernet PAUSE, which pauses all traffic on a physical link. PFC helps you configure lossless transport for traffic flows across a data center bridging (DCB) network.

However, you might want to create a traffic flow that losslessly traverses the L2 DCB network *and* also losslessly traverses an L3 network that connects Ethernet hosts in different L2 networks. In addition to configuring PFC on L2 (bridging) interfaces, you can configure PFC on VLAN-tagged traffic that traverses L3 interfaces. This enables you to preserve the lossless characteristics that PFC provides on VLAN-tagged traffic, even when the traffic crosses L3 interfaces that connect two L2 networks.

NOTE: This topic is applicable for VLAN-tagged traffic only. On supported platforms, you can configure DSCP-based PFC for *untagged* traffic on L3 interfaces and L2 access interfaces. DSCP-based PFC uses a DSCP classifier to classify the traffic based on a 6-bit DSCP value that is mapped to a 3-bit PFC priority value. For details on using DSCP-based PFC on supporting switches, see "Understanding PFC Using DSCP at Layer 3 for Untagged Traffic" on page 518.

Video: Preserving Lossless Behavior on an SDN or Overlay Network

PFC works the same way across L3 interfaces as it works across L2 interfaces. When an output queue buffer reaches a certain fill level threshold, the switch sends a PFC pause message to the connected peer to pause transmission of the traffic on which PFC is enabled. Pausing the incoming traffic prevents the queue buffer from overflowing and dropping packets, just as on L2 interfaces. When the queue buffer fill level decreases below a certain threshold, the interface sends a message to the connected peer to restart traffic transmission.

Although PFC is a DCB technology, PFC also works on L3 interfaces because PFC operates at the queue level. When you use an IEEE 802.1p code point to classify incoming traffic and you enable PFC on the appropriate priority (IEEE 802.1p code point), PFC works on L2 and L3 interfaces.

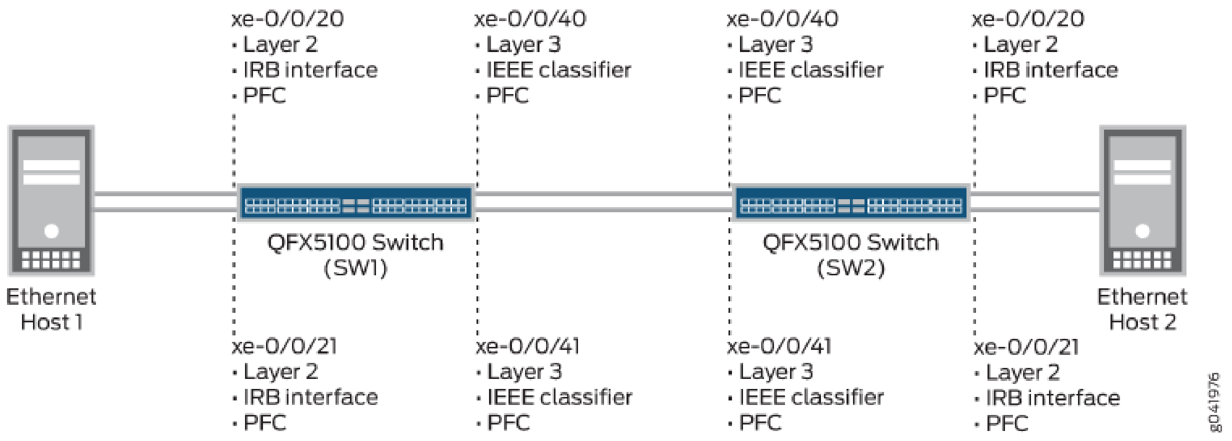
NOTE: Lossless VLAN-tagged traffic on L3 interfaces *must* use an IEEE 802.1p classifier to classify incoming traffic. PFC does not use DSCP or DSCP IPv6 code points to identify VLAN-tagged traffic for flow control. PFC cannot pause traffic flows unless the incoming traffic is classified by an IEEE 802.1p classifier. Do not apply a DSCP (or a DSCP IPv6) classifier to L3 VLAN-tagged traffic on which you want to enable PFC.

Because PFC functionality relies on the mapping (classifying) of incoming traffic to IEEE 802.1p code points and on enabling PFC on the correct code point(s) at each interface, you must ensure that incoming traffic has the correct 3-bit IEEE 802.1p code point (priority) in the priority code point (PCP) field of the Ethernet frame header.

NOTE: L3 interfaces do not support FCoE traffic. FCoE traffic must use L2 interfaces and cannot use L3 interfaces. Therefore, you cannot enable PFC on FCoE traffic across L3 interfaces.

Figure 43 on page 490 shows a topology in which two Ethernet hosts in Layer 2 networks communicate across a Layer 3 network, with PFC enabled on all of the Layer 2 and Layer 3 switch interfaces.

Figure 43: Enabling PFC Across Layer 3 Interface Hops



The Ethernet host-facing interfaces (xe-0/0/20 and xe-0/0/21 on both switches) and the L3 network-facing interfaces (interfaces xe-0/0/40 and xe-0/0/41 on both switches) require different interface configurations to enable PFC on the L3 interfaces. In addition, you must configure CoS for each interface correctly, including enabling PFC on the traffic that you want to treat as lossless traffic:

Ethernet-host facing interfaces (xe-0/0/20 and xe-0/0/21) require the following configuration:

- Set interfaces as family ethernet-switching
- Set the interface mode as trunk mode
- Create VLANs to carry the traffic
- Create IRB interfaces to place the L2 VLAN traffic on L3 for transport between IP networks
- Create an IEEE 802.1p classifier to classify incoming traffic into the correct forwarding class, based on the IEEE 802.1p code point
- Create a CNP to configure PFC on the IEEE 802.1p code point of the traffic that you want treat as lossless traffic
- Apply the classifier and the CNP to the L2 interfaces
- Configure CoS: lossless forwarding classes, hierarchical port scheduling (also known as enhanced transmission selection), or direct port scheduling, depending on your switch, and apply it to the L2 interfaces

L3 IP network-facing interfaces (xe-0/0/40 and xe-0/0/41) require the following configuration:

- Set interfaces as family inet
- Set VLAN tagging on the interfaces
- Create VLANs to carry the traffic

- Create an IEEE 802.1p classifier to classify incoming traffic into the correct forwarding class, based on the IEEE 802.1p code point (do not use a DSCP or DSCP IPv6 classifier)
- Create a CNP to configure PFC on the IEEE 802.1p code point of the traffic that you want to treat as lossless traffic on the L3 interfaces
- Apply the IEEE 802.1p classifier and the CNP to the L3 interfaces
- Configure CoS: lossless forwarding classes, hierarchical port scheduling (enhanced transmission selection), or direct port scheduling, depending on your switch, and apply it to the L3 interfaces

NOTE: Configuring or changing PFC on an interface blocks the entire port until the PFC change is completed. After a PFC change is complete, the port is unblocked and traffic resumes. Blocking the port stops ingress and egress traffic and causes packet loss on all queues on the port until the port is unblocked.

When you configure the Layer 2 and Layer 3 interfaces correctly, the switch enables PFC on the traffic between Ethernet Host 1 and Ethernet Host 2 across the entire path between the two hosts. If any output queue in the path on which PFC is enabled experiences congestion, PFC pauses the traffic and prevents packet loss for the flow.

RELATED DOCUMENTATION

Example: Configuring PFC Across Layer 3 Interfaces

Understanding CoS Flow Control (Ethernet PAUSE and PFC)

Understanding Integrated Routing and Bridging

Example: Configure PFC Across L3 Interfaces

IN THIS SECTION

Priority-based flow control (PFC) helps ensure lossless transport across DCB interfaces by pausing incoming traffic when output queue buffers fill to a certain threshold. In addition to configuring PFC on L2 (bridging) interfaces, you can configure PFC on VLAN-tagged traffic that traverses L3 interfaces. This enables you to preserve the lossless characteristics that PFC provides on VLAN-tagged traffic, even when the traffic crosses L3 interfaces that connect two L2 networks.

NOTE: This topic is applicable for VLAN-tagged traffic only. On supported platforms, you can also configure DSCP-based PFC for *untagged* traffic on L3 interfaces and L2 access interfaces. DSCP-based PFC uses a DSCP classifier to classify the traffic based on a 6-bit DSCP value that is mapped to a 3-bit PFC priority value. For details on configuring DSCP-based PFC on supporting switches, see "Configuring DSCP-based PFC for Layer 3 Untagged Traffic" on page 521.

Requirements

This example uses the following hardware and software components:

- Two switches
- Any supported Junos release
- Two Ethernet hosts

Overview

IN THIS SECTION

On a network that uses two QFX Series switches to connect hosts on two different Ethernet networks across an L3 network, to configure PFC across the L2 and L3 interfaces, you must:

- Configure the L2 and L3 interfaces on the switches
- Configure VLANs to carry the traffic across the L2 and L3 networks
- Configure integrated routing and bridging (IRB) interfaces on the L2 interfaces to move the L2 VLAN traffic to L3
- Configure and apply the appropriate classifiers to the interfaces
- Configure and apply CNPs on the interfaces to enable PFC on the traffic that you want to be lossless

NOTE: Configuring or changing PFC on an interface blocks the entire port until the PFC change is completed. After a PFC change is complete, the port is unblocked and traffic resumes. Blocking the port stops ingress and egress traffic, and causes packet loss on all queues on the port until the port is unblocked.

- Configure lossless forwarding classes and either hierarchical port scheduling (also known as enhanced transmission selection) or direct port scheduling, depending on your switch, on the interfaces

NOTE: PFC operates at the queue level, based on the IEEE 802.1p code point in the priority code point (PCP) field of the Ethernet frame header (sometimes known as the CoS bits). For this reason, VLAN-tagged traffic on L3 interfaces on which you want to enable PFC must use an IEEE 802.1p classifier to map incoming traffic to forwarding classes and loss priorities. You cannot use a DSCP or DSCP IPv6 classifier to classify L3 traffic if you want to enable PFC on VLAN-tagged traffic flows.

Topology

Figure 44 on page 493 shows the topology for this example.

Figure 44: Enabling PFC Across Layer 3 Interface Hops

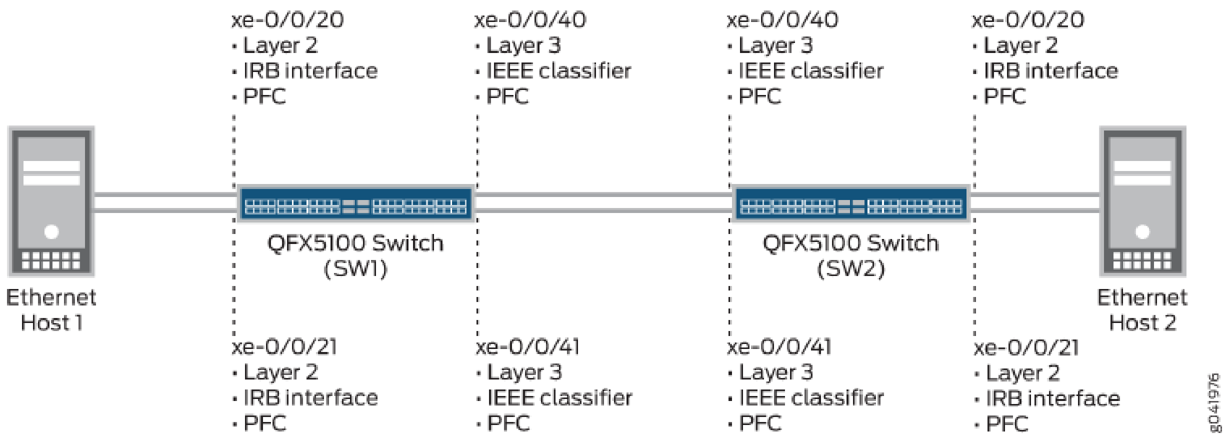


Table 47 on page 494 shows the configuration components for this example. On the two switches, the Ethernet host-facing interfaces use the same interface names and configuration, and the Layer 3 network-facing interfaces use the same interface names and configuration.

Table 47: Components of the PFC Across Layer 3 Interfaces Topology

Component	Settings
Hardware	Two switches, Switch SW1 and Switch SW2. Two Ethernet hosts
L3 interfaces (xe-0/0/40 and xe-0/0/41) and VLANs	Interface xe-0/0/40: • Interface family—inet • Interface IP address—100.103.1.2/24 • VLAN tagging—enabled • Interface VLAN ID—103 Interface xe-0/0/41: • Interface family—inet • Interface IP address—100.104.1.2/24 • VLAN tagging—enabled • Interface VLAN ID—104
L2 interfaces (xe-0/0/20 and xe-0/0/21) and VLAN membership	Family: Ethernet switching Interface mode—trunk Interface xe-0/0/20 VLAN membership—vlan105 Interface xe-0/0/21 VLAN membership—vlan106
VLANs for the IRB interfaces	VLAN unit 105—family inet, IP address 100.105.1.1/24 VLAN unit 106—family inet, IP address 100.106.1.1/24

Table 47: Components of the PFC Across Layer 3 Interfaces Topology *(Continued)*

Component	Settings
L2 IRB interfaces	Interface xe-0/0/20: • IRB interface unit—105 • IRB interface family—inet • IRB interface IP address—100.105.1.1/24 • IRB interface VLAN ID—105 • L3 interface name—irb.105 Interface xe-0/0/21: • IRB interface unit—106 • IRB interface family—inet • IRB interface IP address—100.106.1.1/24 • IRB interface VLAN ID—106 • L3 interface name—irb.106
Forwarding classes (both switches)	Name—lossless-3 Queue mapping—queue 3 Packet drop attribute—no-loss Name—lossless-4 Queue mapping—queue 4 Packet drop attribute—no-loss **NOTE**: Matching the forwarding class names (lossless-*3* and lossless-*4*) to the queue number and to the classified IEEE 802.1p code point (priority) creates a configuration that is logical and easy to map because the forwarding class, queue, and priority all use the same number. Name—all-others Queue mapping—queue 0 Packet drop attribute—none **NOTE**: The forwarding class *all-others* is for best-effort traffic that traverses the interfaces.

Table 47: Components of the PFC Across Layer 3 Interfaces Topology *(Continued)*

Component	Settings
L2 interface BA classifier	Name—lossless-3-4-ieee Forwarding class lossless-3—mapped to code point 011 (IEEE 802.1p priority 3) and a PLP of low Forwarding class lossless-4—mapped to code point 100 (IEEE 802.1p priority 4) and a PLP of low Apply the L2 IEEE 802.1p classifier to both the L2 and the L3 interfaces (xe-0/0/20, xe-0/0/21, xe-0/040, and xe-0/0/41).
CNP (PFC, both switches)	Name—lossless-cnp PFC enabled on IEEE 802.1p code points—011 (lossless-3 forwarding class and priority), 100 (lossless-4 forwarding class and priority) Apply the CNP to both the L2 and the L3 interfaces (xe-0/0/20, xe-0/0/21, xe-0/040, and xe-0/0/41) to enable PFC on IEEE 802.1p code points 011 and 100.
Enhanced transmission selection (ETS) hierarchical port scheduling (only if using ETS)	Hierarchical port scheduling (ETS) includes configuring: • Schedulers to assign bandwidth to traffic • Scheduler mapping to forwarding classes • Grouping of the forwarding classes (priorities) in forwarding class sets (priority groups) • A TCP to assign bandwidth to the forwarding class set and to associate the forwarding class set with the scheduler mapping Hierarchical port scheduling also includes applying the hierarchical scheduler (defined in the TCP) to the interfaces. This example focuses on configuring PFC across the L2 and L3 interfaces. To maintain this focus, this example includes the CLI statements needed to configure hierarchical port scheduling, but does not include descriptive explanations of the configuration. The *Related Documentation* section provides links to example documents that show how to configure hierarchical port scheduling. Apply the scheduling configuration to both the L2 and the L3 interfaces (xe-0/0/20, xe-0/0/21, xe-0/040, and xe-0/0/41).

Table 47: Components of the PFC Across Layer 3 Interfaces Topology *(Continued)*

Component	Settings
Direct port scheduling (only if using port scheduling instead of ETS)	Direct port scheduling includes configuring: • Schedulers to assign bandwidth to traffic • Scheduler mapping to forwarding classes Port scheduling also includes applying the scheduler map to the interfaces. This example focuses on configuring PFC across the L2 and L3 interfaces. To maintain this focus, this example includes the CLI statements needed to configure direct port scheduling, but does not include descriptive explanations of the configuration. The *Related Documentation* section provides links to example documents that show how to configure port scheduling. Apply the scheduling configuration to both the L2 and the L3 interfaces (xe-0/0/20, xe-0/0/21, xe-0/040, and xe-0/0/41).

Configuration

IN THIS SECTION

CLI Quick Configuration

To configure PFC across L3 interfaces, copy the following commands, paste them in a text file, remove the line breaks, change variables and details to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level. The same configuration applies to both Switch SW1 and Switch SW2. The configuration is separated into the configuration common to ETS and direct port scheduling, and the portions of the configuration that apply only to ETS and only to port scheduling.

Common Configuration (Applies to ETS Hierarchical Scheduling and to Port Scheduling)

```
set interfaces xe-0/0/40 vlan-tagging
set interfaces xe-0/0/40 unit 0 vlan-id 103
set interfaces xe-0/0/40 unit 0 family inet address 100.103.1.2/24
set interfaces xe-0/0/41 vlan-tagging
set interfaces xe-0/0/41 unit 0 vlan-id 104
set interfaces xe-0/0/41 unit 0 family inet address 100.104.1.2/24
set interfaces xe-0/0/20 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/20 unit 0 family ethernet-switching vlan members vlan105
set interfaces xe-0/0/21 unit 0 family ethernet-switching interface-mode trunk
set interfaces xe-0/0/21 unit 0 family ethernet-switching vlan members vlan106
set interfaces irb unit 105 family inet address 100.105.1.1/24
set interfaces irb unit 106 family inet address 100.106.1.1/24
set vlans vlan105 vlan-id 105
set vlans vlan106 vlan-id 106
set vlans vlan105 l3-interface irb.105
set vlans vlan106 l3-interface irb.106
set class-of-service forwarding-classes class lossless-3 queue-num 3 no-loss
set class-of-service forwarding-classes class lossless-4 queue-num 4 no-loss
set class-of-service forwarding-classes class all-others queue-num 0
set class-of-service classifiers ieee-802.1 lossless-3-4-ieee forwarding-class lossless-3 loss-
priority low code-points 011
set class-of-service classifiers ieee-802.1 lossless-3-4-ieee forwarding-class lossless-4 loss-
priority low code-points 100
set class-of-service congestion-notification-profile lossless-cnp input ieee-802.1 code-point
011 pfc
set class-of-service congestion-notification-profile lossless-cnp input ieee-802.1 code-point
100 pfc
set class-of-service schedulers lossless_sch transmit-rate 6g
set class-of-service schedulers lossless_sch shaping-rate percent 100
set class-of-service schedulers all-others_sch transmit-rate 4g
set class-of-service scheduler-maps lossless_map forwarding-class lossless-3 scheduler
lossless_sch
set class-of-service scheduler-maps lossless_map forwarding-class lossless-4 scheduler
lossless_sch
set class-of-service scheduler-maps all-others_map forwarding-class all-others scheduler all-
others_sch
set class-of-service interfaces xe-0/0/20 congestion-notification-profile lossless-cnp
set class-of-service interfaces xe-0/0/20 unit 0 classifiers ieee-802.1 lossless-3-4-ieee
set class-of-service interfaces xe-0/0/21 congestion-notification-profile lossless-cnp
set class-of-service interfaces xe-0/0/21 unit 0 classifiers ieee-802.1 lossless-3-4-ieee
set class-of-service interfaces xe-0/0/40 congestion-notification-profile lossless-cnp
set class-of-service interfaces xe-0/0/40 classifiers ieee-802.1 lossless-3-4-ieee
set class-of-service interfaces xe-0/0/41 congestion-notification-profile lossless-cnp
set class-of-service interfaces xe-0/0/41 classifiers ieee-802.1 lossless-3-4-ieee
```

Configuration for ETS Hierarchical Scheduling

The ETS-specific portion of this example configures forwarding class set (priority group) membership and priority group CoS settings (TCP), and assigns the priority group and its CoS configuration to the interfaces.

```
set class-of-service forwarding-class-sets lossless_fc_set class lossless-3
set class-of-service forwarding-class-sets lossless_fc_set class lossless-4
set class-of-service forwarding-class-sets all-others_fc_set class all-others
set class-of-service traffic-control-profiles lossless_tcp scheduler-map lossless_map
set class-of-service traffic-control-profiles lossless_tcp guaranteed-rate percent 60
set class-of-service traffic-control-profiles lossless_tcp shaping-rate percent 100
set class-of-service traffic-control-profiles all-others_tcp scheduler-map all-others_map
set class-of-service traffic-control-profiles all-others_tcp guaranteed-rate percent 40
set class-of-service interfaces xe-0/0/20 forwarding-class-set lossless_fc_set output-traffic-
control-profile lossless_tcp
set class-of-service interfaces xe-0/0/20 forwarding-class-set all-others_fc_set output-traffic-
control-profile all-others_tcp
set class-of-service interfaces xe-0/0/21 forwarding-class-set lossless_fc_set output-traffic-
control-profile lossless_tcp
set class-of-service interfaces xe-0/0/21 forwarding-class-set all-others_fc_set output-traffic-
control-profile all-others_tcp
set class-of-service interfaces xe-0/0/40 forwarding-class-set lossless_fc_set output-traffic-
control-profile lossless_tcp
set class-of-service interfaces xe-0/0/40 forwarding-class-set all-others_fc_set output-traffic-
control-profile all-others_tcp
set class-of-service interfaces xe-0/0/41 forwarding-class-set lossless_fc_set output-traffic-
control-profile lossless_tcp
set class-of-service interfaces xe-0/0/41 forwarding-class-set all-others_fc_set output-traffic-
control-profile all-others_tcp
```

Configuration for Port Scheduling

The port-scheduling-specific portion of this example assigns the scheduler maps (which set the CoS treatment of the forwarding classes in the scheduler map) to the interfaces.

```
[edit class-of-service]
set interfaces xe-0/0/20 scheduler-map lossless_map
set interfaces xe-0/0/20 scheduler-map all-others_map
set interfaces xe-0/0/21 scheduler-map lossless_map
set interfaces xe-0/0/21 scheduler-map all-others_map
set interfaces xe-0/0/40 scheduler-map lossless_map
set interfaces xe-0/0/40 scheduler-map all-others_map
set interfaces xe-0/0/41 scheduler-map lossless_map
set interfaces xe-0/0/41 scheduler-map all-others_map
```

Common Configuration (Applies to ETS Hierarchical Scheduling and to Port Scheduling)

Step-by-Step Procedure

The following procedure shows you how to configure the VLANs, IRB interfaces, lossless forwarding classes, classifiers, PFC settings to enable PFC across L3 interfaces, and the queue scheduling configuration common to ETS and direct port scheduling. For completeness, the ETS hierarchical port scheduling and direct port scheduling configurations are included separately, in the following procedures, but without explanatory text. See the *Related Documentation* links for detailed examples of the scheduling elements of the configuration.

1. Configure the L3 interface VLANs and IP addresses:

```
[edit interfaces]
user@switch# set xe-0/0/40 vlan-tagging
user@switch# set xe-0/0/40 unit 0 vlan-id 103
user@switch# set xe-0/0/40 unit 0 family inet address 100.103.1.2/24
user@switch# set xe-0/0/41 vlan-tagging
user@switch# set xe-0/0/41 unit 0 vlan-id 104
user@switch# set xe-0/0/41 unit 0 family inet address 100.104.1.2/24
```

2. Configure the L2 interface VLAN membership and interface mode:

```
[edit interfaces]
user@switch# set xe-0/0/20 unit 0 family ethernet-switching interface-mode trunk
user@switch# set xe-0/0/20 unit 0 family ethernet-switching vlan members vlan105
```

```
user@switch# set xe-0/0/21 unit 0 family ethernet-switching interface-mode trunk
user@switch# set xe-0/0/21 unit 0 family ethernet-switching vlan members vlan106
```

3. Configure the IRB interfaces and VLANs to transport incoming L2 traffic assigned to VLANs vlan105 (of which interface xe-0/0/20 is a member) and vlan106 (of which interface xe-0/0/21 is a member) across L3:

```
[edit]
user@switch# set interfaces irb unit 105 family inet address 100.105.1.1/24
user@switch# set interfaces irb unit 106 family inet address 100.106.1.1/24
user@switch# set vlans vlan105 vlan-id 105
user@switch# set vlans vlan106 vlan-id 106
user@switch# set vlans vlan105 l3-interface irb.105
user@switch# set vlans vlan106 l3-interface irb.106
```

4. Configure the lossless forwarding classes and a BE forwarding class for any other traffic that might use the interfaces:

```
[edit class-of-service]
user@switch# set forwarding-classes class lossless-3 queue-num 3 no-loss
user@switch# set forwarding-classes class lossless-4 queue-num 4 no-loss
user@switch# set forwarding-classes class all-others queue-num 0
```

5. Configure the IEEE classifier for the L2 and L3 interfaces to classify incoming traffic into the lossless forwarding classes based on the IEEE 802.1p code point of the traffic:

```
[edit class-of-service classifiers]
user@switch# set ieee-802.1 lossless-3-4-ieee forwarding-class lossless-3 loss-priority low
code-points 011
user@switch# set ieee-802.1 lossless-3-4-ieee forwarding-class lossless-4 loss-priority low
code-points 100
```

6. Configure the CNP to enable PFC on the lossless priorities (the lossless forwarding classes mapped to IEEE 802.1p code points 3 and 4):

```
[edit class-of-service congestion-notification-profile]
user@switch# set lossless-cnp input ieee-802.1 code-point 011 pfc
user@switch# set lossless-cnp input ieee-802.1 code-point 100 pfc
```

7. Apply the L2 IEEE 802.1p classifier and the CNP to the L3 interfaces:

```
[edit class-of-service interfaces]
user@switch# set xe-0/0/40 classifiers ieee-802.1 lossless-3-4-ieee
user@switch# set xe-0/0/40 congestion-notification-profile lossless-cnp
user@switch# set xe-0/0/41 classifiers ieee-802.1 lossless-3-4-ieee
user@switch# set xe-0/0/41 congestion-notification-profile lossless-cnp
```

8. Apply the L2 IEEE 802.1p classifier and the CNP to the L2 interfaces:

```
[edit class-of-service interfaces]
user@switch# xe-0/0/20 unit 0 classifiers ieee-802.1 lossless-3-4-ieee
user@switch# xe-0/0/20 congestion-notification-profile lossless-cnp
user@switch# xe-0/0/21 unit 0 classifiers ieee-802.1 lossless-3-4-ieee
user@switch# xe-0/0/21 congestion-notification-profile lossless-cnp
```

9. Configure queue scheduling to support the lossless configuration and map the schedulers to the forwarding classes:

```
[edit class-of-service]
user@switch# set schedulers lossless_sch transmit-rate 6g
user@switch# set schedulers lossless_sch shaping-rate percent 100
user@switch# set schedulers all-others_sch transmit-rate 4g
user@switch# set scheduler-maps lossless_map forwarding-class lossless-3 scheduler
lossless_sch
user@switch# set scheduler-maps lossless_map forwarding-class lossless-4 scheduler
lossless_sch
user@switch# set scheduler-maps all-others_map forwarding-class all-others scheduler all-
others_sch
```

ETS Hierarchical Scheduling Configuration

Step-by-Step Procedure

1. Configure hierarchical scheduling to support the lossless configuration and apply it to the L2 and L3 interfaces:

```
[edit class-of-service interfaces]
user@switch# set forwarding-class-sets lossless_fc_set class lossless-3
user@switch# set forwarding-class-sets lossless_fc_set class lossless-4
user@switch# set forwarding-class-sets all-others_fc_set class all-others
user@switch# set traffic-control-profiles lossless_tcp scheduler-map lossless_map
user@switch# set traffic-control-profiles lossless_tcp guaranteed-rate percent 60
user@switch# set traffic-control-profiles lossless_tcp shaping-rate percent 100
user@switch# set traffic-control-profiles all-others_tcp scheduler-map all-others_map
user@switch# set traffic-control-profiles all-others_tcp guaranteed-rate percent 40
user@switch# set interfaces xe-0/0/20 forwarding-class-set lossless_fc_set output-traffic-
control-profile lossless_tcp
user@switch# set interfaces xe-0/0/20 forwarding-class-set all-others_fc_set output-traffic-
control-profile all-others_tcp
user@switch# set interfaces xe-0/0/21 forwarding-class-set lossless_fc_set output-traffic-
control-profile lossless_tcp
user@switch# set interfaces xe-0/0/21 forwarding-class-set all-others_fc_set output-traffic-
control-profile all-others_tcp
user@switch# set interfaces xe-0/0/40 forwarding-class-set lossless_fc_set output-traffic-
control-profile lossless_tcp
user@switch# set interfaces xe-0/0/40 forwarding-class-set all-others_fc_set output-traffic-
control-profile all-others_tcp
user@switch# set interfaces xe-0/0/41 forwarding-class-set lossless_fc_set output-traffic-
control-profile lossless_tcp
user@switch# set interfaces xe-0/0/41 forwarding-class-set all-others_fc_set output-traffic-
control-profile all-others_tcp
```

Port Scheduling Configuration

Step-by-Step Procedure

1. Apply port scheduling to support the lossless configuration on interfaces:

```
[edit class-of-service]
user@switch# set interfaces xe-0/0/20 scheduler-map lossless_map
user@switch# set interfaces xe-0/0/20 scheduler-map all-others_map
user@switch# set interfaces xe-0/0/21 scheduler-map lossless_map
user@switch# set interfaces xe-0/0/21 scheduler-map all-others_map
user@switch# set interfaces xe-0/0/40 scheduler-map lossless_map
user@switch# set interfaces xe-0/0/40 scheduler-map all-others_map
user@switch# set interfaces xe-0/0/41 scheduler-map lossless_map
user@switch# set interfaces xe-0/0/41 scheduler-map all-others_map
```

Results

Display the results of the interface, VLAN, and CoS configurations. The system shows only the explicitly configured parameters; it does not show default parameters. The results are valid for both Switch SW1 and Switch SW2 because the same configuration is used on both switches. The results are from the ETS hierarchical scheduling configuration, which show the more complex configuration. Direct port scheduling results would not show the TCP or forwarding class set portions of the configuration, but would display the name of the scheduler map under each interface. Other than that, the results are the same.

Display the results of the interface configuration:

```
user@switch# show configuration interfaces
xe-0/0/20 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members vlan105;
            }
        }
    }
}
xe-0/0/21 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members vlan106;
            }
        }
    }
}
xe-0/0/40 {
    vlan-tagging;
    unit 0 {
        vlan-id 103;
        family inet {
            address 100.103.1.2/24;
        }
    }
}
xe-0/0/41 {
    vlan-tagging;
    unit 0 {
        vlan-id 104;
        family inet {
            address 100.104.1.2/24;
        }
    }
}
irb {
    unit 105 {
        family inet {
            address 100.105.1.1/24;
        }
    }
    unit 106 {
        family inet {
            address 100.106.1.1/24;
        }
    }
}
vlan {
    unit 105 {
        family inet {
            address 100.105.1.1/24;
        }
    }
    unit 106 {
        family inet {
            address 100.106.1.1/24;
        }
    }
}
```

Display the results of the vlan configuration:

```
user@switch# show configuration vlans
vlan105 {
    vlan-id 105;
    l3-interface irb.105;
}
vlan106 {
    vlan-id 106;
    l3-interface irb.106;
}
```

Display the results of the CoS configuration:

```
user@switch# show configuration class-of-service
classifiers {
    ieee-802.1 lossless-3-4-ieee {
        forwarding-class lossless-3 {
            loss-priority low code-points 011;
        }
        forwarding-class lossless-4 {
            loss-priority low code-points 100;
        }
    }
}
forwarding-classes {
    class lossless-3 queue-num 3 no-loss;
    class lossless-4 queue-num 4 no-loss;
    class all-others queue-num 0;
}
traffic-control-profiles {
    lossless_tcp {
        scheduler-map lossless_map;
        shaping-rate percent 100;
        guaranteed-rate percent 60;
    }
    all-others_tcp {
        scheduler-map all-others_map;
        guaranteed-rate percent 40;
    }
}
forwarding-class-sets {
    lossless_fc_set {
        class lossless-3;
        class lossless-4;
    }
    all-others_fc_set {
        class all-others;
    }
}
congestion-notification-profile {
    lossless-cnp {
        input {
            ieee-802.1 {
                code-point 011 {
                    pfc;
                }
                code-point 100 {
                    pfc;
                }
            }
        }
    }
}
interfaces {
    xe-0/0/20 {
        forwarding-class-set {
            lossless_fc_set {
                output-traffic-control-profile lossless_tcp;
            }
            all-others_fc_set {
                output-traffic-control-profile all-others_tcp;
            }
        }
        congestion-notification-profile lossless-cnp;
        unit 0 {
            classifiers {
                ieee-802.1 lossless-3-4-ieee;
            }
        }
    }
    xe-0/0/21 {
        forwarding-class-set {
            all-others_fc_set {
                output-traffic-control-profile all-others_tcp;
            }
            lossless_fc_set {
                output-traffic-control-profile lossless_tcp;
            }
        }
        congestion-notification-profile lossless-cnp;
        unit 0 {
            classifiers {
                ieee-802.1 lossless-3-4-ieee;
            }
        }
    }
    xe-0/0/40 {
        forwarding-class-set {
            lossless_fc_set {
                output-traffic-control-profile lossless_tcp;
            }
            all-others_fc_set {
                output-traffic-control-profile all-others_tcp;
            }
        }
        congestion-notification-profile lossless-cnp;
        classifiers {
            ieee-802.1 lossless-3-4-ieee;
        }
    }
    xe-0/0/41 {
        forwarding-class-set {
            lossless_fc_set {
                output-traffic-control-profile lossless_tcp;
            }
            all-others_fc_set {
                output-traffic-control-profile all-others_tcp;
            }
        }
        congestion-notification-profile lossless-cnp;
        classifiers {
            ieee-802.1 lossless-3-4-ieee;
        }
    }
}
scheduler-maps {
    lossless_map {
        forwarding-class lossless-3 scheduler lossless_sch;
        forwarding-class lossless-4 scheduler lossless_sch;
    }
    all-others_map {
        forwarding-class all-others scheduler all-others_sch;
    }
}
schedulers {
    lossless_sch {
        transmit-rate 6g;
        shaping-rate percent 100;
    }
    all-others_sch {
        transmit-rate 4g;
    }
}
```

TIP: To quickly configure the switch, issue the `load merge terminal` command, and then copy the hierarchies and paste them into the switch terminal window.

Verification

IN THIS SECTION

To verify that the PFC across L3 interfaces configuration has been created and is operating properly, perform these tasks:

Verifying the Interface Configuration

Purpose

Verify that the L2 Ethernet interfaces, L3 IP interfaces, IRB interfaces, and VLAN interfaces have been created on the switch and are correctly configured.

Action

Display the switch interface configuration using the `show configuration interfaces` command:

```
user@switch> show configuration interfaces
xe-0/0/20 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members vlan105;
            }
        }
    }
}
xe-0/0/21 {
    unit 0 {
        family ethernet-switching {
            interface-mode trunk;
            vlan {
                members vlan106;
            }
        }
    }
}
```

```
xe-0/0/40 {
    vlan-tagging;
    unit 0 {
        vlan-id 103;
        family inet {
            address 100.103.1.2/24;
        }
    }
}
xe-0/0/41 {
    vlan-tagging;
    unit 0 {
        vlan-id 104;
        family inet {
            address 100.104.1.2/24;
        }
    }
}
irb {
    unit 105 {
        family inet {
            address 100.105.1.1/24;
        }
    }
    unit 106 {
        family inet {
            address 100.106.1.1/24;
        }
    }
}
vlan {
    unit 105 {
        family inet {
            address 100.105.1.1/24;
        }
    }
    unit 106 {
        family inet {
            address 100.106.1.1/24;
        }
    }
}
```

Meaning

The `show configuration interfaces` command displays all of the interfaces configured on the switch. The command output shows that:

- Interfaces xe-0/0/20 and xe-0/0/21 are Ethernet interfaces (family ethernet-switching) in trunk interface mode. Interface xe-0/0/20 is a member of VLAN vlan105, and interface xe-0/0/21 is a member of VLAN vlan106.
- Interfaces xe-0/0/40 and xe-0/0/41 are IP interfaces (family inet) with VLAN tagging enabled. Interface xe-0/0/40 has an IP address of 100.103.1.2/24 and a VLAN ID of 103. Interface xe-0/0/41 has an IP address of 100.104.1.2/24 and a VLAN ID of 104.
- Two IRB interfaces are configured, IRB unit 105 with an IP address of 100.105.1.1/24 and IRB unit 106 with an IP address of 100.106.1.1/24.
- Two VLAN interfaces are configured: VLAN unit 105 with an IP address of 100.105.1.1/24 (for IRB interface unit 105) and VLAN unit 106 with an IP address of 100.106.1.1/24 (for IRB interface unit 106).

Verifying the VLAN Configuration

Purpose

Verify that VLANs have been created on the switch and are correctly configured.

Action

Display the VLAN configuration using the `show configuration vlans` command:

```
user@switch> show configuration vlans
vlan105 {
    vlan-id 105;
    l3-interface irb.105;
}
vlan106 {
    vlan-id 106;
    l3-interface irb.106;
}
```

Meaning

The `show configuration vlans` command displays all of the VLANs configured on the switch. The command output shows that:

- VLAN vlan105 has been configured with VLAN ID 105 on IRB interface irb.105.
- VLAN vlan106 has been configured with VLAN ID 106 on IRB interface irb.106.

Verifying the PFC Configuration (CNP)

Purpose

Verify that PFC has been enabled on the correct IEEE 802.1p code points (priorities) in the CNP.

Action

Display the PFC configuration using the `show configuration class-of-service congestion-notification-profile` command:

```
user@switch> show configuration class-of-service congestion-notification-profile
lossless-cnp {
    input {
        ieee-802.1 {
            code-point 011 {
                pfc;
            }
            code-point 100 {
                pfc;
            }
        }
    }
}
```

Meaning

The `show configuration class-of-service congestion-notification-profile` command displays all of the CNPs configured on the switch. The command output shows that:

- The CNP named `lossless-cnp` is configured on the switch.
- The CNP `lossless-cnp` enables PFC on IEEE 802.1p code points 100 and 100.

Verify the Forwarding Class Configuration

Purpose

Verify that the two lossless forwarding classes and the BE forwarding class have been configured on the switch.

Action

Display the forwarding class configuration using the `show configuration class-of-service forwarding-classes` command:

```
user@switch> show configuration class-of-service forwarding-classes
class lossless-3 queue-num 3 no-loss;
class lossless-4 queue-num 4 no-loss;
class all-others queue-num 0;
```

Meaning

The `show configuration class-of-service forwarding-classes` command displays all of the forwarding classes configured on the switch. Default forwarding classes are not displayed. The command output shows that:

- Forwarding class `lossless-3` is mapped to queue 3 and is configured as a lossless forwarding class (the `no-loss` attribute is applied)
- Forwarding class `lossless-4` is mapped to queue 4 and is configured as a lossless forwarding class (the `no-loss` attribute is applied)
- Forwarding class `all-others` is mapped to queue 0. It is not a lossless forwarding class (the `no-loss` attribute is not applied).

Verifying the Classifier Configuration

Purpose

Verify that the IEEE 802.1p classifier has been configured on the switch.

Action

Display the classifier configuration using the `show configuration class-of-service classifiers` command:

```
user@switch> show configuration class-of-service classifiers
ieee-802.1 lossless-3-4-ieee {
    forwarding-class lossless-3 {
        loss-priority low code-points 011;
    }
    forwarding-class lossless-4 {
        loss-priority low code-points 100;
    }
}
```

Meaning

The `show configuration class-of-service classifiers` command displays all of the classifiers configured on the switch. The command output shows that the L2 IEEE 802.1p classifier `lossless-3-4-ieee` classifies traffic with the code point 011 into the `lossless-3` forwarding class with a loss priority of `low`, and classifies traffic with the code point 100 into the `lossless-4` forwarding class with a loss priority of `low`.

Verifying the Interface CoS Configuration (Hierarchical Scheduling, PFC, and Classifier Mapping to Interfaces)

Purpose

Verify that the interfaces have the correct hierarchical scheduling, PFC, and classifier configurations.

NOTE: The results are from the ETS hierarchical scheduling configuration, which shows the more complex configuration. Direct port scheduling results do not show the TCP or forwarding class set portions of the interface configuration, but rather display the name of the scheduler map under each interface.

Action

Display the interface CoS configuration using the `show configuration class-of-service interfaces` command:

```
user@switch> show configuration class-of-service interfaces
xe-0/0/20 {
    forwarding-class-set {
        lossless_fc_set {
            output-traffic-control-profile lossless_tcp;
        }
        all-others_fc_set {
            output-traffic-control-profile all-others_tcp;
        }
    }
    congestion-notification-profile lossless-cnp;
    unit 0 {
        classifiers {
            ieee-802.1 lossless-3-4-ieee;
        }
    }
}
xe-0/0/21 {
    forwarding-class-set {
        all-others_fc_set {
            output-traffic-control-profile all-others_tcp;
        }
        lossless_fc_set {
            output-traffic-control-profile lossless_tcp;
        }
    }
    congestion-notification-profile lossless-cnp;
    unit 0 {
        classifiers {
            ieee-802.1 lossless-3-4-ieee;
        }
    }
}
xe-0/0/40 {
    forwarding-class-set {
        lossless_fc_set {
            output-traffic-control-profile lossless_tcp;
        }
        all-others_fc_set {
            output-traffic-control-profile all-others_tcp;
        }
    }
    congestion-notification-profile lossless-cnp;
    classifiers {
        ieee-802.1 lossless-3-4-ieee;
    }
}
xe-0/0/41 {
    forwarding-class-set {
        lossless_fc_set {
            output-traffic-control-profile lossless_tcp;
        }
        all-others_fc_set {
            output-traffic-control-profile all-others_tcp;
        }
    }
    congestion-notification-profile lossless-cnp;
    classifiers {
        ieee-802.1 lossless-3-4-ieee;
    }
}
```

Meaning

The `show configuration class-of-service interfaces` command displays all of the CoS components configured on the switch interfaces. The command output shows that:

- The configuration on L2 Ethernet interfaces xe-0/0/20 and xe-0/0/21 includes:
  - Hierarchical scheduling—The forwarding class set `lossless_fc_set` with the TCP `lossless_tcp` for the lossless traffic, and the forwarding class set `all-others_fc_set` with the TCP `all-others_tcp` for the best-effort traffic are applied to both interfaces.
  - PFC—The `lossless-cnp` CNP is applied to both interfaces.
  - Classifiers—The L2 IEEE 802.1p classifier `lossless-3-4-ieee` is applied to both interfaces.
- The configuration on L3 IP interfaces xe-0/0/40 and xe-0/0/41 includes:
  - Hierarchical scheduling—The forwarding class set `lossless_fc_set` with the TCP `lossless_tcp` for the lossless traffic, and the forwarding class set `all-others_fc_set` with the TCP `all-others_tcp` for the best-effort traffic are applied to both interfaces.
  - PFC—The `lossless-cnp` CNP is applied to both interfaces.
  - Classifiers—The L2 IEEE 802.1p classifier `lossless-3-4-ieee` is applied to both interfaces. Traffic that would use a DSCP or a DSCP IPv6 classifier if it were configured uses the IEEE 802.1p classifier instead. Using the IEEE 802.1p classifier allows the interface to use PFC to pause traffic during periods of congestion to prevent packet loss.

RELATED DOCUMENTATION

Understanding PFC Functionality Across Layer 3 Interfaces

Understanding PFC Using DSCP at Layer 3 for Untagged Traffic

IN THIS SECTION

Protocols such as Remote Direct Memory Access (RDMA) over converged Ethernet version 2 (RoCEv2) require lossless behavior for traffic across Layer 3 connections to Layer 2 Ethernet subnetworks. Traditionally, priority-based flow control (PFC) can be used to prevent traffic loss when congestion occurs on Layer 2 or Layer 3 interfaces for VLAN-tagged traffic by selectively pausing traffic on any of eight priorities corresponding to IEEE 802.1p code points in the VLAN headers of incoming traffic on an interface. However, *untagged* traffic—traffic without VLAN tagging—cannot be examined for IEEE 802.1p code points on which to pause traffic.

To support lossless traffic flow at Layer 3 for untagged traffic, we support enabling PFC for Layer 3 interfaces and Layer 2 access interfaces using Distributed Services code point (DSCP) values in the Layer 3 IP header of incoming traffic, rather than IEEE 802.1p code point values in a Layer 2 VLAN header.

Overview of DSCP-based PFC

PFC is a data center bridging technology operating at Layer 2, and DSCP information is exchanged in IP headers at Layer 3. However, you can configure DSCP-based PFC, which preserves lossless behavior across Layer 3 network connections for untagged traffic.

PFC operates by generating pause frames for traffic identified on configured code points in incoming traffic to notify the peer to pause transmission when the link is congested. With DSCP-based PFC enabled, pause frames are triggered based on a configured 6-bit DSCP value (corresponding to decimal values 0-63) in the Layer 3 IP header of incoming traffic.

However, PFC can only send pause frames with a 3-bit PFC priority—one of 8 code points corresponding to decimal values 0-7—which, for VLAN-tagged traffic, usually corresponds to the IEEE

802.1p code points in the incoming traffic VLAN headers. Untagged traffic provides no reference for IEEE 802.1p code point values, so to trigger PFC on a DSCP value, the DSCP value must be mapped explicitly in the configuration to a PFC priority to use in the PFC pause frames sent to the peer when congestion occurs for that code point. You can map traffic on a DSCP value to a PFC priority when you define the no-loss forwarding class with which you want to classify DSCP-based PFC traffic. The forwarding class must also be mapped to an output queue with no-loss behavior.

NOTE: You cannot assign the same PFC priority to more than one forwarding class because the mapped PFC priority value is used as the forwarding class ID when DSCP-based PFC is configured.

A DSCP classifier (instead of an IEEE 802.1p classifier) is also required to specify that incoming traffic with the above-configured DSCP value belongs to the no-loss forwarding class. Any DSCP values for which DSCP-based PFC is enabled on a interface must be specified in either the default DSCP classifier or in a user-defined DSCP classifier associated with the interface.

To enable DSCP-based PFC on an interface, define an input congestion notification profile with the same DSCP value (and desired buffering parameters), and associate it with the interface.

The peer device should have a matching PFC configuration for the mapped PFC priority code points.

Limitations of DSCP-based PFC

The following are limitations of DSCP-based PFC:

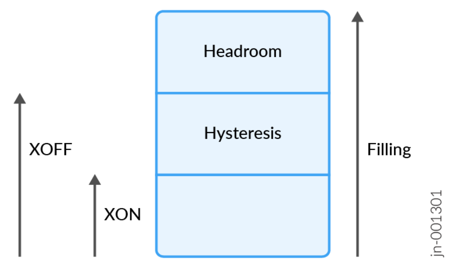
- You cannot configure both DSCP-based PFC and IEEE 802.1p PFC under the same congestion notification profile, or associate both a DSCP-based congestion notification profile and an IEEE 802.1p congestion notification profile with the same interface.
- DSCP-based PFC is supported on Layer 3 interfaces and Layer 2 access interfaces for untagged traffic only. PFC behavior is unpredictable if VLAN-tagged packets are received on an interface with DSCP-based PFC enabled.
- Each no-loss forwarding class can only be associated with a unique 3-bit PFC priority value from 0 through 7.

Configurable PFC Accounting Thresholds

On supported platforms, there are virtual PFC pause buffers called PFC accounts that you define within a congestion notification profile (CNP). Each ingress port can have two such PFC accounts, You can independently set the PFC priority to transmit pause frames and the thresholds of `XOFF` and `XON` for each PFC account.

Consider Figure 45 on page 520, which shows a typical pause buffer. In this diagram, the buffer starts to fill from the bottom up due to congestion on the egress port. When the buffer fill reaches `XOFF`, a PFC Pause frame is sent upstream to pause traffic associated with the PFC class. The headroom space allows for in-flight packets and processing delays so that the upstream device can pause traffic before the buffer fills completely and begins dropping packets. The system uses the cable length and the maximum receive unit (MRU) to calculate the amount of buffer headroom reserved to support PFC. The the shorter the cable length and lower the MRU, the less headroom buffer space is required for PFC.

Figure 45: Typical Pause Buffer



When congestion reduces and the buffer fill falls under the `XON` threshold level, a resume frame is sent upstream to restart the data traffic.

For PFC to work effectively you must correctly set `XOFF`, `XON`, and the headroom buffer for each PFC account. Junos calculates the headroom space based on the defined cable length and other internally calculated factors.

You define a PFC account for input traffic in a CNP:

1. Define one or two PFC accounts. Set a PFC priority for each account, and if necessary, set `XOFF` and `XON` for each account.

2. Set the code-points that you are using for PFC and assign a PFC account to each code-point.

3. Set the correct `cable-length` for the CNP. The cable length is the distance between the interface and its peer interfaces in meters.

Platform-Specific PFC Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behaviors for your platform.

Platform	Difference
PTX10000 Series	• You can configure up to two queues as `no-loss` when defining forwarding classes. • PTX10000 Series routers support up to 100KM of cable length. • PTX10000 Series routers have virtual PFC Pause buffers called PFC-Accounts. • All PFC pause buffer accounting happens with respect to ingress ports and not egress ports. • If both PFC and ECN are enabled, when the occupancy of a PFC account is above XON, by default ECN-capable packets are marked as congestion experienced (CE).

RELATED DOCUMENTATION

DSCP-based PFC for Layer 3 Untagged Traffic | **521**

Understanding CoS Classifiers

Understanding CoS Flow Control (Ethernet PAUSE and PFC)

Understanding CoS Forwarding Classes

Understanding CoS IEEE 802.1p Priorities for Lossless Traffic Flows

DSCP-based PFC for Layer 3 Untagged Traffic

IN THIS SECTION

You can configure DSCP-based PFC to support lossless behavior for untagged traffic across Layer 3 connections to Layer 2 subnetworks for protocols such as Remote Direct Memory Access (RDMA) over converged Ethernet version 2 (RoCEv2).

Overview

With DSCP-based PFC, pause frames are generated to notify the peer that the link is congested based on a configured 6-bit Distributed Services code point (DSCP) value in the Layer 3 IP header of incoming traffic, rather than a 3-bit IEEE 802.1p code point in the Layer 2 VLAN header.

Because PFC can only send pause frames corresponding to PFC priority code points, the 6-bit configured DSCP value must be mapped to a 3-bit PFC priority to use in pause frames when DSCP-based PFC is triggered. Configuring the mapping involves mapping the PFC priority value to a no-loss forwarding class when you map the forwarding class to a queue, defining a congestion notification profile to enable PFC on traffic with the desired DSCP value, and configuring a DSCP classifier to associate the PFC priority-mapped forwarding class (along with the loss priority) with the configured DSCP value on which to trigger PFC pause frames.

The peer device should have output PFC and a corresponding flow control queue configured to match the PFC priority configuration on the device.

Use Feature Explorer to confirm platform and release support for specific features.

DSCP-based PFC for Layer 3 Untagged Traffic in AI-ML Data Centers

AI and ML applications are rapidly expanding in data centers. When dealing with AI and ML workloads and large data sets, one critical challenge is handling the size of the data. Offloading the computation to graphics processing units (GPUs) can significantly speed up this task. However, the data size and the model, especially with large language models (LLMs), often exceed the memory capacity of a single GPU. As a result, you commonly require multiple GPUs to achieve reasonable job completion times, especially for training.

The performance of an AI data center depends on the number of GPUs that are used and the efficiency of the network that connects them. Slowdowns in the network can lead to underutilization of GPUs and longer job completion times. Ethernet-based networks are becoming more popular as an alternative to InfiniBand for AI data center networking. One solution is the Remote Direct Memory Access (RDMA) over Converged Ethernet version 2 (RoCEv2) network.

RoCEv2 involves encapsulating RDMA protocol packets within UDP packets for transport over Ethernet networks. The RoCEv2 protocol utilizes priority-based flow control (PFC) to establish a drop-free network, while *data center quantized congestion notification* (DCQCN) provides end-to-end congestion control for RoCEv2. Junos OS Evolved supports DCQCN by combining explicit congestion notification (ECN) and PFC to enable end-to-end lossless AI Ethernet networking.

To support lossless IPv6 traffic across Layer 3 (L3) connections to Layer 2 (L2) subnetworks, you can configure PFC to operate using 6-bit Differentiated Services code point (DSCP) values from L3 headers of untagged VLAN traffic. You can use PFC with DSCP as an alternative to IEEE 802.1p priority values in L2 VLAN-tagged packet headers. You need DSCP-based PFC to support RoCEv2.

Benefits

- Utilize Ethernet-based networks for AI-ML data center networking.
- Improve network efficiency for large data sets.
- Enable end-to-end lossless AI-ML Ethernet networking.

Configuration

To configure DSCP-based PFC:

1. Map a lossless forwarding class to a PFC priority—a 3-bit value represented in decimal form (0-7)—to use in the PFC pause frames.

   You must also assign an output queue to the forwarding class with the `queue-num` option. The `no-loss` option is required in this case to support lossless behavior for DSCP-based PFC, and the `pfc-priority` statement specifies the priority value mapping, as follows:

   ```
   [edit class-of-service]
   user@device# set forwarding-classes class class-name queue-num queue-number no-loss
   user@device# set forwarding-classes class class-name pfc-priority pfc-priority
   ```

2. Define an input congestion notification profile to enable PFC on traffic specified by the desired 6-bit DSCP value. Optionally configure the maximum receive unit (MRU) and cable length (used to determine PFC buffer headroom space reserved for the link):

   NOTE: You cannot configure both DSCP-based PFC and IEEE 802.1p PFC under the same congestion notification profile.

   ```
   [edit class-of-service]
   user@device# set congestion-notification-profile name input dscp code-point code-point-bits
   pfc mru mru-value
   user@device# set congestion-notification-profile name cable-length cable-length-value
   ```

3. Set up a DSCP classifier for the configured DSCP value and no-loss forwarding class mapped in the previous steps:

```
[edit class-of-service]
user@device# set classifiers dscp classifier-name forwarding-class class-name loss-priority
level code-points code-point-bits
```

4. Assign the classifier and congestion notification profile set up in the previous steps to an interface on which you are enabling DSCP-based PFC:

```
[edit class-of-service]
user@device# set interfaces interface-name classifiers dscp classifier-name
user@device# set interfaces interface-name congestion-notification-profile profile-name
```

5. Review your configuration.

   For example, with the following sample commands configuring DSCP-based PFC for interface xe-0/0/1, PFC pause frames will be generated with PFC priority 3 when incoming traffic with DSCP value 110000 becomes congested:

```
set interfaces xe-0/0/1 unit 0 family inet address 10.1.1.2/24
set class-of-service forwarding-classes class fc1 queue-num 1 no-loss
set class-of-service forwarding-classes class fc1 pfc-priority 3
set class-of-service congestion-notification-profile dpfc-cnp input dscp code-point 110000 pfc
set class-of-service classifiers dscp dpfc forwarding-class fc1 loss-priority low code-points
110000
set class-of-service interfaces xe-0/0/1 congestion-notification-profile dpfc-cnp
set class-of-service interfaces xe-0/0/1 classifiers dscp dpfc
```

Configuration for PTX10000 Series Routers

1. PTX10000 Series routers have separate buffer spaces for lossy and lossless queues, with 10percent of the total buffer spaces reserved for lossless queues by default. If necessary, adjust the amount of buffer space reserved for lossless queus.

   You adjust the percent of buffer space reserved for lossless queues on a per-FPC basis:

```
[edit chassis]
user@device# set fpc fpc-slot no-loss buffer percentage percent
```

2. Map a lossless forwarding class to a PFC priority—a 3-bit value represented in decimal form (0-7)—to use in the PFC pause frames.

   You must also assign an output queue to the forwarding class with the `queue-num` option. The `no-loss` option is required in this case to support lossless behavior for DSCP-based PFC, and the `pfc-priority` statement specifies the priority value mapping, as follows:

   ```
   [edit class-of-service]
   user@device# set forwarding-classes class class-name queue-num queue-number no-loss
   user@device# set forwarding-classes class class-name pfc-priority pfc-priority
   ```

3. Define an input congestion notification profile to enable PFC on traffic specified by the desired 6-bit DSCP value. Optionally configure the maximum receive unit (MRU) and cable length (used to determine PFC buffer headroom space reserved for the link):

   NOTE: You cannot configure both DSCP-based PFC and IEEE 802.1p PFC under the same congestion notification profile.

   ```
   [edit class-of-service]
   user@device# set congestion-notification-profile name input dscp code-point code-point-bits
   pfc mru mru-value
   user@device# set congestion-notification-profile name cable-length cable-length-value
   ```

   Include the PFC account(s) and assign a PFC account to each code point.

   ```
   [edit class-of-service]
   user@device# set congestion-notification-profile name input pfc-account account-name pfc-
   priority priority
   user@device# set congestion-notification-profile name input pfc-account account-name xoff
   value
   user@device# set congestion-notification-profile name input pfc-account account-name xon value
   user@device# set congestion-notification-profile name input dscp code-point code-point-bits
   pfc-account account-name
   ```

4. Set up a DSCP classifier for the configured DSCP value and no-loss forwarding class mapped in the previous steps:

```
[edit class-of-service]
user@device# set classifiers dscp classifier-name forwarding-class class-name loss-priority
level code-points code-point-bits
```

5. Assign the classifier and congestion notification profile set up in the previous steps to an interface on which you are enabling DSCP-based PFC:

```
[edit class-of-service]
user@device# set interfaces interface-name classifiers dscp classifier-name
user@device# set interfaces interface-name congestion-notification-profile profile-name
```

6. Review your configuration.

   For example, with the following sample commands configuring DSCP-based PFC for interface xe-0/0/1, PFC pause frames will be generated with PFC priority 3 when incoming traffic with DSCP value 110000 reaches a delay equal to XOFF, which is set to 5000 microseconds, and a resume frame is sent with the delay falls back below XON, which is set to 2500 microseconds:

```
set chassis 0 no-loss buffer percentage 25
set interfaces xe-0/0/1 unit 0 family inet address 10.1.1.2/24
set class-of-service forwarding-classes class fc1 queue-num 1 no-loss
set class-of-service forwarding-classes class fc1 pfc-priority 3
set class-of-service congestion-notification-profile dpfc-cnp input cable-length 1000
set class-of-service congestion-notification-profile dpfc-cnp input pfc-account pfca1 pfc-priority 3
set class-of-service congestion-notification-profile dpfc-cnp input pfc-account pfca1 xoff 5000
set class-of-service congestion-notification-profile dpfc-cnp input pfc-account pfca1 xon 2500
set class-of-service congestion-notification-profile dpfc-cnp input dscp code-point 110000 pfc
set class-of-service congestion-notification-profile dpfc-cnp input dscp code-point 110000 pfc-account
pfca1
set class-of-service classifiers dscp dpfc forwarding-class fc1 loss-priority low code-points 110000
set class-of-service interfaces xe-0/0/1 congestion-notification-profile dpfc-cnp
set class-of-service interfaces xe-0/0/1 classifiers dscp dpfc
```

Verify the configuration.

1. Check the ingress port.

```
show interfaces interface-name extensive | match Priority
```

```
show interfaces queue interface-name
```

2. Display the DSCP-based input congestion notification profile.

```
show class-of-service congestion-notification-profile cnp name
```

3. Display which forwarding classes are mapped to each PFC priority.

```
show class-of-service forwarding-classes
```

RELATED DOCUMENTATION

Understanding PFC Using DSCP at Layer 3 for Untagged Traffic

Configuring CoS PFC (Congestion Notification Profiles)

Defining CoS BA Classifiers (DSCP, DSCP IPv6, IEEE 802.1p)

Defining CoS Forwarding Classes

PFC Watchdog

IN THIS SECTION

PFC Watchdog Overview

IN THIS SECTION

Priority-based flow control (PFC) allows independent flow control for each class of service to ensure that congestion does not result in frame loss. PFC pause frames instruct the link partner to halt packet transmission. These frames can propagate through the network, causing traffic on the PFC streams to stop in what is known as a *PFC pause storm*. Use the *PFC watchdog* to detect and to resolve PFC pause storms.

The PFC watchdog monitors PFC-enabled ports for PFC pause storms. The PFC watchdog intervenes when a PFC-enabled port receives PFC pause frames for an extended period of time and is unable to schedule any of the data packets on PFC-enabled queues. The PFC watchdog mitigates the situation by disabling the queue where the PFC pause storm was detected for a set length of time. This length of time, called the recovery time, is configurable. After the recovery time passes, the PFC watchdog reenables the affected queue.

You can monitor the number of PFC pause storms that have been detected and recovered, as well as the number of packets that have been dropped, on a particular interface.

In lossless Ethernet fabrics such as those used in AI-ML data centers, the PFC watchdog plays a critical role in keeping the fabric lossless even during congestion.

Use Feature Explorer to confirm platform and release support for specific features.

Benefits

- Quickly detect and resolve PFC pause storms.
- Maintain lossless traffic links.
- Improve link quality.

Understanding PFC Watchdog

IN THIS SECTION

The PFC watchdog has three key functions: detection, mitigation, and restoration.

Detection

The PFC watchdog checks the status of PFC queues at regular intervals called *polling intervals*. If the PFC watchdog finds a PFC queue with a non-zero pause timer, it compares the queue's current transmit counter register to the last recorded value. If the PFC queue has not transmitted any packets since the last polling interval, the PFC watchdog checks if there are any packets in the queue. If there are packets on the queue that are not being transmitted and there are no flow control frames on that port, the PFC watchdog detects a stall condition.

The PFC watchdog monitors the PFC-enabled queues periodically for continuous PFC pause assertion by the downstream device when the queue is empty. If this occurs, PFC watchdog detects a stall condition. The system must detect this stall condition within a specified amount of time. This length of time is determined by how you configure two statements: `poll-interval` and `detection`.

The PFC watchdog checks the status of PFC queues at regular intervals. Configure this interval in milliseconds using the `poll-interval` statement. The PFC watchdog checks the status of the queues once per polling interval. The default interval is 100 ms. The minimum interval is 100 ms and the maximum is 1000 ms.

The PFC watchdog must detect stall conditions for at least two consecutive polling intervals before it determines that a PFC queue has stalled. Configure the `detection` statement to control how many polling intervals the PFC watchdog waits before it mitigates the stalled traffic. The default is two polling intervals. The maximum number is 10 polling intervals.

The total detection time is the length of the polling interval multiplied by the number of polling intervals.

Mitigation

When the PFC watchdog detects that a PFC queue has stalled, it moves the queue to the mitigation state. First it disables the queue where it detected the PFC pause storm for a period of time called the recovery time.

Configure the `pfc-watchdog-action` statement to specify the action that the PFC watchdog takes to mitigate the traffic congestion. The only option is the drop action. It drops all queued packets and all newly arriving packets for the stalled PFC queue. The system monitors all packet drops on the PFC queue during the recovery time.

Restoration

When the recovery time ends, the PFC watchdog collects the ingress drop counters and any other drop counters associated with disabling the PFC queue. The PFC watchdog maintains a count of the packets lost during the last recovery and the total number of lost packets due to PFC mitigation since the device was started. The PFC watchdog then restores the queue and re-enables PFC.

Use the `recovery` statement to configure how long the PFC watchdog disables the affected queue. The minimum recovery period is 200 ms and the maximum is 10,000 ms. After the recovery time passes, the PFC watchdog re-enables PFC on the affected queues.

Configure PFC Watchdog

You can enable the PFC watchdog on all PFC-enabled queues. The PFC watchdog recovery is a global setting, so it requires the same action on all ports to function. When you configure the PFC watchdog on multiple ports, make sure all ports are configured with the same type of action (drop or forward). By default, all ports use the drop action.

Enabling PFC watchdog on the congestion notification profile without configuring other options enables the PFC watchdog with the default values. By default, the polling interval is 100 ms, the detection period is set to 2 (that is, two polling intervals, or 200 ms), and the recovery time is 200 ms.

PFC watchdog only works for PFC queues. To designate a queue as a PFC queue, use the `flow-control-queue` statement with the queue number. For example:

```
set class-of-service congestion-notification-profile cnp output ieee-802.1 code-point 011 flow-
control-queue 3
set class-of-service congestion-notification-profile cnp output ieee-802.1 code-point 100 flow-
control-queue 4
```

1. Enable PFC watchdog. Use the `pfc-watchdog` statement at the `[edit class-of-service congestion-notification-profile` *`profile-name`*`]` hierarchy level:

   ```
   set class-of-service congestion-notification-profile profile-name pfc-watchdog
   ```

2. Configure the polling interval in milliseconds. The polling interval is how often the PFC watchdog checks the status of PFC queues.

   ```
   set class-of-service congestion-notification-profile profile-name pfc-watchdog poll-interval
   time
   ```

3. Configure the detection interval number. The detection interval number is how many polling intervals the PFC watchdog waits before it mitigates the stalled traffic.

```
set class-of-service congestion-notification-profile profile-name pfc-watchdog detection
polling-interval-number
```

4. Specify the action that the PFC watchdog takes to mitigate the traffic congestion.

```
set class-of-service congestion-notification-profile profile-name pfc-watchdog watchdog-
action drop
```

5. Configure the recovery time in milliseconds. The recovery time is how long the PFC watchdog disables the affected queue for before it restores PFC.

```
set class-of-service congestion-notification-profile cnp-name pfc-watchdog recovery time
```

6. Verify your configuration with the `show class-of-service congestion-notification-profile` *`profile-name`* command.

   The detection time shown is the polling interval multiplied by the detection interval number. In this case, the polling interval is 100 milliseconds, so the configured number of detection intervals was two.

```
user@device> show class-of-service congestion-notification-profile cnp-profile

Name: cnp, Index: 0
Type: Input
Cable Length: 100
Type: Output
  Priority     Flow-Control-Queues
  011
               3
  Priority     Flow-Control-Queues
  100
               4
PFC Watchdog : enabled
PFC-action : drop
Polling Interval : 100 ms
Detection Time : 200 ms
Recovery Time : 200 ms
```

Use the PFC Watchdog for Monitoring

You can track PFC watchdog events in the system log. The device logs PFC watchdog detection and recovery events in the system log with a timestamp. You can identify these logs from the following messages:

- `CDA PfcWd: PFC Watchdog detection enabled on ifd: et-0/0/16 Poll Interval:100ms Detection Period:200ms Recovery Interval:200ms`—PFC watchdog was enabled on a new port.
- `CDA PfcWd: PFC Storm Detected! on ifd:et-0/0/16 Queue: 3 Priority: 3 BLOCKED for AutoRecovery Recovery Time: 200ms`—PFC watchdog detected a stall condition.
- `CDA PfcWd: PFC Storm Recovered on Port ifd:et-0/0/16 Queue: 3 Priority: 3 UNBLOCKED after AutoRecovery Recovery Time: 200ms`—PFC watchdog restores the PFC queue and the queue recovers from the PFC pause storm.

You can also monitor the PFC watchdog statistics on a particular interface. Use the following command to view the number of PFC pause storms that have been detected and recovered, as well as the number of packets that have been dropped, on the PFC queues on an interface:

```
user@device> show interfaces interface extensive
...

Priority Flow Control Watchdog Statistics:
             Detected     Recovered  LastPacketDropCount  TotalPacketDropCount
Queue :  0      0              0       0                      0
Queue :  1      0              0       0                      0
Queue :  2      0              0       0                      0
Queue :  3      0              0       0                      0
Queue :  4      0              0       0                      0
Queue :  5      0              0       0                      0
Queue :  6      0              0       0                      0
Queue :  7      0              0       0                      0

...
```

RELATED DOCUMENTATION

Configuring CoS PFC (Congestion Notification Profiles)

congestion-notification-profile

CoS Explicit Congestion Notification (ECN)

SUMMARY

Use ECN to improve transport efficiency between two IP endpoints by marking packets to signal congestion instead of dropping them.

IN THIS SECTION

Overview

Explicit congestion notification (ECN) enables end-to-end congestion notification between two endpoints on IP based networks. ECN improves transport efficiency between two IP endpoints by marking packets to signal congestion instead of dropping them for RED (random early detection). The two endpoints are an ECN-enabled sender and an ECN-enabled receiver. ECN must be enabled on both endpoints and on all of the intermediate devices between the endpoints for ECN to work properly. Any device in the transmission path that does not support ECN breaks the end-to-end ECN functionality.

NOTE: ECN marking happens at the IP header level. ECN works well with both TCP and UDP/ROCEv2 traffic.

ECN notifies networks about congestion with the goal of reducing packet loss and delay by making the sending device decrease the transmission rate until the congestion clears, without dropping packets. RFC 3168, *The Addition of Explicit Congestion Notification (ECN) to IP*, defines ECN.

ECN is disabled by default. You can enable ECN on any queue and the transport protocol chooses whether to use it. Normally, you enable ECN only on queues that handle best-effort traffic because other traffic types use different methods of congestion notification—lossless traffic uses priority-based flow control (PFC) and strict-high priority traffic receives all of the port bandwidth it requires up to the point of a configured maximum rate.

You enable ECN on individual output queues (as represented by forwarding classes) by enabling ECN in the queue scheduler configuration, mapping the scheduler to forwarding classes (queues), and then applying the scheduler to interfaces.

There are two types of ECN: Static ECN and Dynamic ECN (D-ECN). Static ECN requires you to manually configure the thresholds that trigger congestion notifications, and the thresholds stay the same until you change the setting. Dynamic ECN adjusts the thresholds automatically based on real-time conditions such as queue length and traffic patterns.

On supported devices, both ECN versions can be enabled on your device at the same time, but only one version can be assigned to a particular queue at a time.

NOTE: For ECN to work on a queue, you must also apply a weighted random early detection (WRED) packet drop profile to the queue.

How ECN Works

IN THIS SECTION

Without ECN, devices respond to network congestion by dropping IP packets. Dropped packets signal the network that congestion is occurring. Devices on the IP network respond to packet drops by reducing the packet transmission rate to allow the congestion to clear. However, the packet drop method of congestion notification and management has some disadvantages. For example, packets are dropped and must be retransmitted. Also, bursty traffic can cause the network to reduce the transmission rate too much, resulting in inefficient bandwidth utilization.

Instead of dropping packets to signal network congestion, ECN marks packets to signal network congestion, without dropping the packets. For ECN to work, all of the devices in the path between two ECN-enabled endpoints must have ECN enabled. ECN is negotiated during the establishment of the connection between the endpoints.

ECN-enabled devices determine the queue congestion state based on the WRED packet drop profile configuration applied to the queue, so each ECN-enabled queue must also have a WRED drop profile. If a queue fills to the level at which the WRED drop profile has a packet drop probability greater than zero

(0), the device might mark a packet as experiencing congestion. Whether or not a device marks a packet as experiencing congestion is the same probability as the drop probability of the queue at that fill level.

ECN communicates whether or not congestion is experienced by marking the two least-significant bits in the differentiated services (DiffServ) field in the IP header. The most significant six bits in the DiffServ field contain the Differentiated Services Code Point (DSCP) bits. The state of the two ECN bits signals whether or not the packet is an ECN-capable packet and whether or not congestion has been experienced.

ECN-capable senders mark packets as ECN-capable. If a sender is not ECN-capable, it marks packets as not ECN-capable. If an ECN-capable packet experiences congestion at the egress queue of a device, the device marks the packet as experiencing congestion. When the packet reaches the ECN-capable receiver (destination endpoint), the receiver echoes the congestion indicator to the sender (source endpoint) by sending a packet marked to indicate congestion.

After receiving the congestion indicator from the receiver, the source endpoint reduces the transmission rate to relieve the congestion. This is similar to the result of TCP congestion notification and management, but instead of dropping the packet to signal network congestion, ECN marks the packet and the receiver echoes the congestion notification to the sender. Because the packet is not dropped, the packet does not need to be retransmitted.

ECN Bits in the DiffServ Field

The two ECN bits in the DiffServ field provide four codes that determine if a packet is marked as an ECN-capable transport (ECT) packet, meaning that both endpoints of the transport protocol are ECN-capable, and if there is congestion experienced (CE), as shown in the table below:

Table 48: ECN Bit Codes

ECN Bits (Code)	Meaning
00	Non-ECT—Packet is marked as not ECN-capable
01	ECT(1)—Endpoints of the transport protocol are ECN-capable
10	ECT(0)—Endpoints of the transport protocol are ECN-capable
11	CE—Congestion experienced

Codes 01 and 10 have the same meaning: the sending and receiving endpoints of the transport protocol are ECN-capable. There is no difference between these codes.

End-to-End ECN Behavior

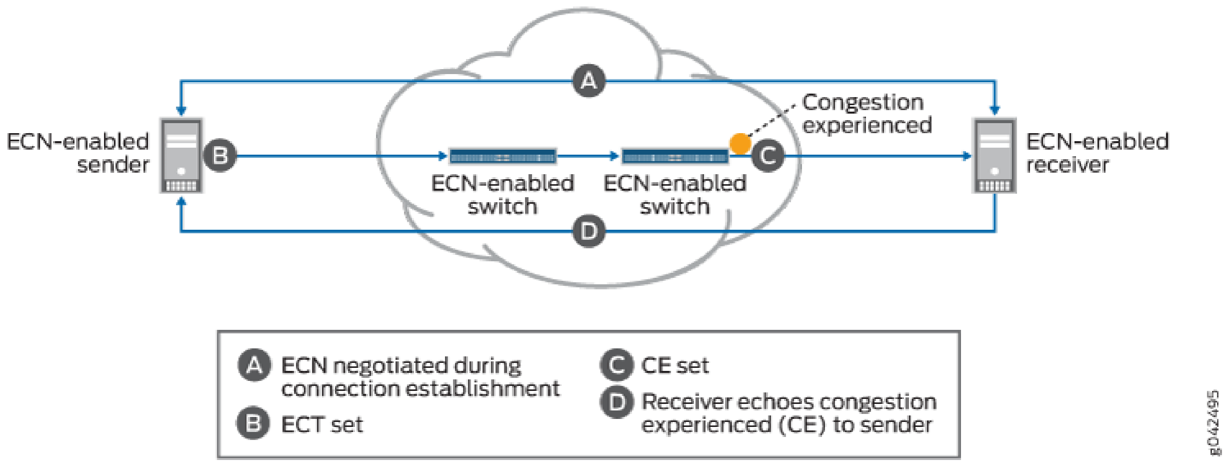
After the sending and receiving endpoints negotiate ECN, the sending endpoint marks packets as ECN-capable by setting the DiffServ ECN field to ECT(1) (01) or ECT(0) (10). Every intermediate device between the endpoints must have ECN enabled or it does not work.

When a packet traverses a device and experiences congestion at an output queue that uses the WRED packet drop mechanism, the device marks the packet as experiencing congestion by setting the DiffServ ECN field to CE (11). Instead of dropping the packet (as with TCP congestion notification), the device forwards the packet.

NOTE: At the egress queue, the WRED algorithm determines whether or not a packet is drop eligible based on the queue fill level (how full the queue is). If a packet is drop eligible and marked as ECN-capable, the packet can be marked CE and forwarded. If a packet is drop eligible and is not marked as ECN-capable, it might be dropped. See "WRED Drop Profile Control of ECN Thresholds" on page 539 for more information about the WRED algorithm.

When the packet reaches the receiver endpoint, the CE mark tells the receiver that there is network congestion. The receiver then sends (echoes) a message to the sender that indicates there is congestion on the network. The sender acknowledges the congestion notification message and reduces its transmission rate. The diagram below summarizes how ECN works to mitigate network congestion:

Figure 46: Explicit Congestion Notification



End-to-end ECN Behavior

End-to-end ECN behavior includes:

1. The ECN-capable sender and receiver negotiate ECN capability during the establishment of their connection.

2. After successful negotiation of ECN capability, the ECN-capable sender sends IP packets with the ECT field set to the receiver.

   NOTE: You must enable ECN on *all* of the intermediate devices in the path between the sender and the receiver.

3. If the WRED algorithm on a device egress queue determines that the queue is experiencing congestion and the packet is drop eligible, the device can mark the packet as "congestion experienced" (CE) to indicate to the receiver that there is congestion on the network. If the packet has already been marked CE (congestion has already been experienced at the egress of another device), the device forwards the packet with CE marked.

   If there is no congestion at the egress queue, the device forwards the packet and does not change the ECT-enabled marking of the ECN bits, so the packet is still marked as ECN-capable but not as experiencing congestion.

4. The receiver receives a packet marked CE to indicate that congestion was experienced along the congestion path.

5. The receiver echoes (sends) a packet back to the sender with the ECE bit (bit 9) marked in the flag field of the TCP/UDP header. The ECE bit is the ECN echo flag bit, which notifies the sender that there is congestion on the network.

6. The sender reduces the data transmission rate and sends a packet to the receiver with the CWR bit (bit 8) marked in the flag field of the TCP/UDP header. The CWR bit is the congestion window reduced flag bit, which acknowledges to the receiver that the congestion experienced notification was received.

7. When the receiver receives the CWR flag, the receiver stops setting the ECE bit in replies to the sender.

The table below summarizes the behavior of traffic on ECN-enabled queues.

Table 49: Traffic Behavior on ECN-Enabled Queues

Incoming IP Packet Marking of ECN Bits	ECN Configuration on the Output Queue	Action if WRED Algorithm Determines Packet is Drop Eligible	Outgoing Packet Marking of ECN Bits
Non-ECT (00)	Does not matter	Drop.	No ECN bits marked
ECT (10 or 01)	ECN disabled	Drop	Packet dropped—no ECN bits marked
ECT (10 or 01)	ECN enabled	Do not drop. Mark packet as experiencing congestion (CE, bits 11).	Packet marked ECT (11) to indicate congestion
CE (11)	ECN disabled	Drop	Packet dropped—no ECN bits marked
CE (11)	ECN enabled	Do not drop. Packet is already marked as experiencing congestion, forward packet without changing the ECN marking.	Packet marked ECT (11) to indicate congestion

When an output queue is not experiencing congestion as defined by the WRED drop profile mapped to the queue, all packets are forwarded, and no packets are dropped.

ECN Compared to PFC and Ethernet PAUSE

ECN is an end-to-end network congestion notification mechanism for IP traffic. Priority-based flow control (PFC) (IEEE 802.1Qbb) and Ethernet PAUSE (IEEE 802.3X) are different types of congestion management mechanisms.

ECN requires that an output queue must also have an associated WRED packet drop profile. Output queues used for traffic on which PFC is enabled should not have an associated WRED drop profile. Interfaces on which Ethernet PAUSE is enabled should not have an associated WRED drop profile.

PFC is a peer-to-peer flow control mechanism to support lossless traffic. PFC enables connected peer devices to pause flow transmission during periods of congestion. PFC enables you to pause traffic on a specified type of flow on a link instead of on all traffic on a link. For example, you can (and should) enable PFC on lossless traffic classes such as the `fcoe` forwarding class. Ethernet PAUSE is also a peer-to-peer flow control mechanism, but instead of pausing only specified traffic flows, Ethernet PAUSE pauses all traffic on a physical link.

With PFC and Ethernet PAUSE, the sending and receiving endpoints of a flow do not communicate congestion information to each other across the intermediate devices. Instead, PFC controls flows between two PFC-enabled peer devices (for example, devices) that support data center bridging (DCB) standards. PFC works by sending a pause message to the connected peer when the flow output queue becomes congested. Ethernet PAUSE simply pauses all traffic on a link during periods of congestion and does not require DCB.

PFC works this way: if a device output queue fills to a certain threshold, the device sends a PFC pause message to the connected peer device that is transmitting data. The pause message tells the transmitting device to pause transmission of the flow. When the congestion clears, the device sends another PFC message to tell the connected peer to resume transmission. (If the output queue of the transmitting device also reaches a certain threshold, that device can in turn send a PFC pause message to the connected peer that is transmitting to it. In this way, PFC can propagate a transmission pause back through the network.)

See *Understanding CoS Flow Control (Ethernet PAUSE and PFC)* for more information. You can also refer to *Understanding PFC Functionality Across Layer 3 Interfaces*.

WRED Drop Profile Control of ECN Thresholds

IN THIS SECTION

You apply WRED drop profiles to forwarding classes (which are mapped to output queues) to control how the device marks ECN-capable packets. A scheduler map associates a drop profile with a scheduler and a forwarding class, and then you apply the scheduler map to interfaces to implement the scheduling properties for the forwarding class on those interfaces.

Drop profiles define queue fill level (the percentage of queue fullness) and drop probability (the percentage probability that a packet is dropped) pairs. When a queue fills to a specified level, traffic that matches the drop profile has the drop probability paired with that fill level. When you configure a drop profile, you configure pairs of fill levels and drop probabilities to control how packets drop at different levels of queue fullness.

The first fill level and drop probability pair is the drop start point. Until the queue reaches the first fill level, packets are not dropped. When the queue reaches the first fill level, packets that exceed the fill level have a probability of being dropped that equals the drop probability paired with the fill level.

The last fill level and drop probability pair is the drop end point. When the queue reaches the last fill level, all packets are dropped unless they are configured for ECN.

NOTE: Lossless queues (forwarding class configured with the `no-loss` packet drop attribute) and strict-high priority queues do not use drop profiles. Lossless queues use PFC to control the flow of traffic. Strict-high priority queues receive all of the port bandwidth they require up to the configured maximum bandwidth limit.

Different devices support different amounts of fill level/drop probability pairs in drop profiles.

NOTE: Do not configure the last fill level as 100 percent.

The drop profile configuration affects ECN packets as follows:

- Drop start point—ECN-capable packets might be marked as congestion experienced (CE).
- Drop end point—ECN-capable packets are always marked CE.

As a queue fills from the drop start point to the drop end point, the probability that an ECN packet is marked CE is the same as the probability that a non-ECN packet is dropped if you apply the drop profile to best-effort traffic. As the queue fills, the probability of an ECN packet being marked CE increases, just as the probability of a non-ECN packet being dropped increases when you apply the drop profile to best-effort traffic.

At the drop end point, all ECN packets are marked CE, but the ECN packets are not dropped. When the queue fill level exceeds the drop end point, all ECN packets are marked CE. At this point, all non-ECN packets are dropped. ECN packets (and all other packets) are tail-dropped if the queue fills completely.

How to Apply a WRED Drop Profile to an Output Queue: ETS Hierarchical Scheduling

To configure a WRED packet drop profile and apply it to an output queue (using hierarchical scheduling on devices that support ETS):

1. Configure a drop profile using the statement `set class-of-service drop-profiles` *`profile-name`* `interpolate fill-level` *`drop-start-point`* `fill-level` *`drop-end-point`* `drop-probability 0 drop-probability` *`percentage`*.

2. Map the drop profile to a queue scheduler using the statement `set class-of-service schedulers` *`scheduler-name`* `drop-profile-map loss-priority (low | medium-high | high) protocol any drop-profile` *`profile-name`*. The name of the drop-profile is the name of the WRED profile configured in Step 1.

3. Map the scheduler, which Step 2 associates with the drop profile, to the output queue using the statement `set class-of-service scheduler-maps` *`map-name`* `forwarding-class` *`forwarding-class-name`* `scheduler` *`scheduler-name`*. The forwarding class identifies the output queue. Forwarding classes are mapped to

output queues by default, and can be remapped to different queues by explicit user configuration. The scheduler name is the scheduler configured in Step 2.

4. Associate the scheduler map with a traffic control profile using the statement `set class-of-service traffic-control-profiles` *`tcp-name`* `scheduler-map` *`map-name`*. The scheduler map name is the name configured in Step 3.

5. Associate the traffic control profile with an interface using the statement `set class-of-service interface` *`interface-name`* `forwarding-class-set` *`forwarding-class-set-name`* `output-traffic-control-profile` *`tcp-name`*. The output traffic control profile name is the name of the traffic control profile configured in Step 4.

   The interface uses the scheduler map in the traffic control profile to apply the drop profile (and other attributes, including the enable ECN attribute) to the output queue (forwarding class) on that interface. Because you can use different traffic control profiles to map different schedulers to different interfaces, the same queue number on different interfaces can handle traffic in different ways.

How to Apply a WRED Drop Profile to an Output Queue: Port Scheduling

You can configure a WRED packet drop profile and apply it to an output queue on devices that support port scheduling (ETS hierarchical scheduling is either not supported or not used). To configure a WRED packet drop profile and apply it to an output queue on devices that support port scheduling (ETS hierarchical scheduling is either not supported or not used):

1. Configure a drop profile using the statement `set class-of-service drop-profiles` *`profile-name`* `interpolate fill-level` *`level1 level2 ... level32`* `drop-probability` *`probability1 probability2 ... probability32`*. You can specify as few as two fill level/drop probability pairs or as many as 32 pairs.

2. Map the drop profile to a queue scheduler using the statement `set class-of-service schedulers` *`scheduler-name`* `drop-profile-map loss-priority (low | medium-high | high) drop-profile` *`profile-name`*. The name of the drop-profile is the name of the WRED profile configured in Step 1.

3. Map the scheduler, which Step 2 associates with the drop profile, to the output queue using the statement `set class-of-service scheduler-maps` *`map-name`* `forwarding-class` *`forwarding-class-name`* `scheduler` *`scheduler-name`*. The forwarding class identifies the output queue. Forwarding classes are mapped to output queues by default, and can be remapped to different queues by explicit user configuration. The scheduler name is the scheduler configured in Step 2.

4. Associate the scheduler map with an interface using the statement `set class-of-service interfaces` *`interface-name`* `scheduler-map` *`scheduler-map-name`*.

   The interface uses the scheduler map to apply the drop profile (and other attributes) to the output queue mapped to the forwarding class on that interface. Because you can use different scheduler maps on different interfaces, the same queue number on different interfaces can handle traffic in different ways.

Dynamic ECN

IN THIS SECTION

Dynamic ECN enhances the ECN feature set by providing a way to automate the thresholds triggering a congestion notification event. Junos OS Evolved monitors real-time conditions like queue length and traffic patterns to evaluate whether or not a threshold should be adjusted. This results in a faster response to congestion events than static ECN, and improves congestion control efficiency.

D-ECN is more difficult to implement than static ECN, and requires active monitoring. You should evaluate your network conditions and configuration to decide if D-ECN is the best fit for your network.

You can check Feature Explorer to see if your device supports D-ECN.

Support, Limitations, and Notes

If the WRED algorithm that is mapped to a queue does not find a packet drop eligible, then the ECN configuration and ECN bits marking does not matter. The packet transport behavior is the same as when ECN is not enabled.

ECN is disabled by default. Normally, you enable ECN only on queues that handle best-effort traffic, and you do not enable ECN on queues that handle lossless traffic or strict-high priority traffic.

ECN supports the following:

- IPv4 and IPv6 packets
- Untagged, single-tagged, and double-tagged packets
- The outer IP header of IP tunneled packets (but not the inner IP header)

ECN does not support the following:

- IP packets with MPLS encapsulation
- The inner IP header of IP tunneled packets (however, ECN works on the outer IP header)
- Multicast, broadcast, and destination lookup fail (DLF) traffic
- Non-IP traffic

NOTE: To apply a WRED drop profile to non-ECT traffic, configure a multifield (MF) classifier to assign non-ECT traffic to a different output queue that is not ECN-enabled, and then apply the WRED drop profile to that queue.

ECN Packets per Queue

IN THIS SECTION

Overview

IN THIS SECTION

Explicit congestion notification (ECN) enables two endpoint devices on TCP/IP-based networks to send end-to-end congestion notifications to each other. Without ECN, devices respond to network congestion by dropping TCP/IP packets. The dropped packets signal the occurrence of network congestion. In contrast, ECN marks packets to signal network congestion without dropping the packets. ECN reduces packet loss by making the sending device decrease the transmission rate until the congestion clears.

Packets may be delayed as the device decreases the transmission rate until congestion clears. To account for how many packets are delayed, you can use the `show interfaces queue` command to view the amount of ECN congestion experienced (CE) traffic in the queue.

Benefits

- Identify the packets that have experienced congestion.
- Helps in identifying if traffic is going to reach the queue buffer limits.
- Enables quick troubleshooting of network congestion points.

Configuration

1. Configure ECN.

   ECN is disabled by default. For how to configure ECN, see Example: Configuring Static and Dynamic ECN.
2. Enable ECN on both endpoints and on all of the intermediate devices between the endpoints.

   ECN must be enabled this way for ECN to work properly. Any device in the transmission path that does not support ECN breaks the end-to-end ECN functionality.
3. Use the `show interfaces queue` command to view the amount of traffic that has experienced congestion.

   The `ECN-CE packets` field shows the number of packets that have experienced congestion, while the `ECN-CE bytes` field shows the number of total bytes in those packets.

   The per-queue ECN counters `ECN-CE packets` and `ECN-CE bytes` only count packets that experienced congestion on the local switch.

   For example:

```
show interfaces queue et-0/0/5 forwarding-class network-control

Physical interface: et-0/0/5, up, Physical link is Up
  Interface index: 1262, SNMP ifIndex: 974
Forwarding classes: 12 supported, 9 in use
Egress queues: 12 supported, 9 in use
Queue: 3, Forwarding classes: network-control1
  Queued:
    Packets              :             15239998              856158 pps
    Bytes                :           2225039708           999992904 bps
  Transmitted:
    Packets              :             15239998              856158 pps
    Bytes                :           2225039708           999992904 bps
    Tail-dropped packets :                    0                   0 pps
    Tail-dropped bytes   :                    0                   0 bps
    RED-dropped packets  :                    0                   0 pps
    RED-dropped bytes    :                    0                   0 bps
    ECN-CE packets       :              8577686              482043 pps
    ECN-CE bytes         :           1252342156            70378315 bps
```

On-Chip Buffer with ECN on PTX10002-36QDD Routers

PTX10002-36QDD routers support fine-grained control over buffer allocation and congestion management. This allows you to configure buffers in such a way that they remain "on-chip"; that is, buffered packets remain on the main traffic chip rather than be stored on a separate buffering chip. On-chip buffering is optimized for low-latency, high-throughput scenarios.

VOQs with a buffer size of 40µs, at the configured target rate, will be always on-chip. On-chip VOQs do not participate in on/off chip decisions.

NOTE: The default buffer size value for unconfigured queues is 1 byte, which makes the corresponding VOQ also on-chip.

When you have a VOQ with an OCB and then configure an ECN profile for the queue, after 100% ECN marking probability is reached at the defined queue occupancy, the queue stays on-chip.

As an example, shown below, say you want to mark traffic after a queue grows to 8µs. Assuming the RTT over the router fabric is 6µs, configure an ECN profile with 100% drop probability at a fill level of 35% (35% of 40µs is 14us = 8µs buffer size + 6µs RTT). The queue stays on chip at 14µs occupancy, and the network will have 36µs to react before the queue fills up and starts tail-dropping.

```
[edit class-of-service]
set drop-profile dp1 fill-level 35 drop-probability 100
set schedulers s1 explicit-congestion-notification
set schedulers s1 drop-profile-map loss-priority low protocol any drop-profile dp1
set schedulers s1 buffer-size temporal 40
set schedulers s1 priority low
set scheduler-maps map1 forwarding-class fc1 scheduler s1
set interfaces xe-0/0/1 scheduler-map map1
```

Platform-Specific Behavior

Use Feature Explorer to confirm platform and release support for ECN.

Use the following table to review platform-specific behaviors for this feature.

Platform	Difference
PTX10000 Series	• You can implement low-threshold ECN (begin ECN marking as soon as the buffer starts filling up) by defining a buffer rate for a scheduler and a low percentage drop profile. • The buffer rate acts as the base rate for buffer size calculation. The buffer rate is the target rate of a VOQ, which is the intended egress queue rate during typical congestion. Set the `buffer-rate` at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level. • You can also define more granular (to the tenths of a percent) and smaller fill-level percentages for drop profiles. That is, you can set a fill-level as low as 0.7 percent. **NOTE**: PTX routers support only Static ECN. • If both PFC and ECN are enabled, when the occupancy of a PFC account is above XON, by default ECN-capable packets are marked as congestion experienced (CE). • On PTX10002-36QDD routers, virtual output queues (VOQs) with a buffer size equal to or less than 40 microseconds, at the configured buffer rate, are always on-chip. When you have a VOQ with an OCB and then configure an ECN profile for the queue, the queue remains on-chip after the system reaches 100% ECN marking probability at the defined queue occupancy. Keeping the queue on-chip eliminates the need to access slower external memory. This approach is ideal for low-latency, high-throughput applications.

(Continued)

Platform	Difference
QFX5000 Series	On QFX5K platforms, ECN functionality is tightly integrated with WRED thresholds. WRED thresholds are static, so ECN also works based on static calculations of buffer thresholds. However, actual shared buffer usage of queues is dynamic. Following are the formulas used for ECN marking threshold calculations at the time of ECN configuration. • `max buffer access eligibility for ECN enabled queue = (( shared pool size * hardware_alpha)/(1 + hardware_alpha)) + egress queue dedicated buffer` • `ECN marking start threshold = WRED start fill level percent * Max buffer access eligibility for ECN enabled queue` • `ECN 100% marking threshold = WRED end fill level percent * Max buffer access eligibility for ECN enabled queue` During congestion for ECN capable packets, ECN CE marking starts after `ECN marking start threshold` is reached. ECN capable packets are probabilistically ECN CE marked until `ECN 100% marking threshold` is reached. After this threshold, all ECN capable packets are ECN CE marked until the queue reaches `max buffer access eligibility for ECN enabled queue`. After this threshold, tail drops occur. In the above calculation of `max buffer access eligibility for ECN enabled queue`, the best case scenario of a single queue alone competing for shared buffer space is assumed. However, the actual shared buffer use for a congested queue can decrease dynamically based on the number of competing queues for the shared buffer at any point of time. Following is the formula to calculate the `actual dynamic max buffer usage per queue`. • `actual max buffer usage for ECN enabled queue = (shared pool size * hw_alpha) / (1 + (hw_alpha * number of competing queues)) + egress queue dedicated buffer + ingress Pg dedicated buffer by the traffic flow` Two parameters, `number of competing queues` and `ingress Pg dedicated buffer by the traffic flow`, used in `actual dynamic max buffer usage per queue` calculation cannot be considered while

(Continued)

Platform	Difference
	calculating the ECN threshold as both these parameters are dynamic in nature. This creates a possibility that actual max buffer usage for an ECN enabled queue can be below the calculated static ECN marking thresholds. Therefore, with certain shared buffer and WRED fill level configurations, there is a possibility of packet tail drops due to shared buffer exhaustion occurring even before ECN marking on ECN enabled lossy queues. For lossless queues, due to the above limitation, PFC can start from an ingress port before ECN marking, as the PFC XOFF threshold is dynamic, unlike the static ECN threshold. You can determine proper ECN marking thresholds by monitoring the peak buffer usage of congested queues and fine tuning the ECN/WRED thresholds accordingly.
QFX10000 Series	• On QFX10000 switches, when you enable a queue for ECN and apply a WRED drop profile to the queue, the WRED drop profile only sets the thresholds for marking ECN traffic as experiencing congestion (CE, 11). On ECN-enabled queues, the WRED drop profile does not set drop thresholds for non-ECT (00) traffic (traffic that is not ECN-capable). Instead, the switch uses the tail-drop algorithm on traffic is that is marked non-ECT on ECN-enabled queues during periods of congestion.
SRX Series	• Explicit congestion notification (ECN) enables end-to-end congestion notification between two endpoints on TCP/IP based networks. The two endpoints are an ECN-enabled sender and an ECN-enabled receiver. ECN must be enabled on both endpoints. However, in the case of an unsupported peer, an SRX Series Firewall that supports ECN bootstraps the incoming packets from the unsupported peer and marks the packets to signal network congestion when it occurs. • If the client is not ECN capable, then the SRX firewall negotiates ECN on behalf of client during the connection establishment. The SRX firewall sets the ECE and CWR bits in the TCP header of the SYN packet.

RELATED DOCUMENTATION

Example: Configuring Static and Dynamic ECN | 549

Example: Configuring Static and Dynamic ECN

IN THIS SECTION

This example shows how to enable explicit congestion notification (ECN) on an output queue.

Requirements

This example uses the following hardware and software components:

- One device that supports ECN.

NOTE: Use Feature Explorer to confirm platform and release support for static and dynamic ECN.

Overview

ECN enables end-to-end congestion notification between two endpoints on TCP/IP based networks. The two endpoints are an ECN-enabled sender and an ECN-enabled receiver. ECN must be enabled on both endpoints and on all of the intermediate devices between the endpoints for ECN to work properly. Any device in the transmission path that does not support ECN breaks the end-to-end ECN functionality

A weighted random early detection (WRED) packet drop profile must be applied to the output queues on which ECN is enabled. ECN uses the WRED drop profile thresholds to mark packets when the output queue experiences congestion.

ECN reduces packet loss by forwarding ECN-capable packets during periods of network congestion instead of dropping those packets. (TCP notifies the network about congestion by dropping packets.) During periods of congestion, ECN marks ECN-capable packets that egress from congested queues.

When the receiver receives an ECN packet that is marked as experiencing congestion, the receiver echoes the congestion state back to the sender. The sender then reduces its transmission rate to clear the congestion.

ECN is disabled by default. You can enable ECN on best-effort traffic. ECN should not be enabled on lossless traffic queues, which uses priority-based flow control (PFC) for congestion notification, and ECN should not be enabled on strict-high priority traffic queues.

There are two types of ECN available: Static ECN and Dynamic ECN. Static ECN requires you to manually set the threshold levels that trigger a notification event. Dynamic ECN automatically adjusts the thresholds based on real-time conditions like queue length and traffic patterns.

To enable ECN on an output queue, you not only need to enable ECN in the queue scheduler, you also need to:

- Configure a WRED packet drop profile.
- Configure a queue scheduler that includes the WRED drop profile and enables ECN. (This example shows only ECN and drop profile configuration; you can also configure bandwidth, priority, and buffer settings in a scheduler.)
- Map the queue scheduler to a forwarding class (output queue) in a scheduler map.
- If you are using enhanced transmission selection (ETS) hierarchical scheduling, add the forwarding class to a forwarding class set (priority group).
- If you are using ETS, associate the queue scheduler map with a traffic control profile (priority group scheduler for hierarchical scheduling).
- If you are using ETS, apply the traffic control profile and the forwarding class set to an interface. On that interface, the output queue uses the scheduler mapped to the forwarding class, as specified by the scheduler map attached to the traffic control profile. This enables ECN on the queue and applies the WRED drop profile to the queue.

  If you are using port scheduling, apply the scheduler map to an interface. On that interface, the output queue uses the scheduler mapped to the forwarding class in the scheduler map, which enables ECN on the queue and applies the WRED drop profile to the queue.

Table 50 on page 551 shows the configuration components for this example.

Table 50: Components of the ECN Configuration Example

Component	Settings
Drop profile (with two fill level/drop probability pairs)	Name: `be-dp` Drop start fill level: `30` percent Drop end fill level: `75` percent Drop probability at drop start (minimum drop rate): `0` percent Drop probability at drop end (maximum drop rate): `80` percent
Scheduler	Name: `be-sched` ECN: enabled Drop profile: `be-dp` Transmit rate: `25%` Buffer size: `25%` Priority: `low`
Scheduler map	Name: `be-map` Forwarding class: `best-effort` Scheduler: `be-sched` **NOTE**: By default, the `best-effort` forwarding class is mapped to output queue `0`.
Forwarding class set (ETS only)	Name: `be-pg` Forwarding class: `best-effort` (queue 0)
Traffic control profile (ETS only)	Name: `be-tcp` Scheduler map: `be-map`
Interface (ETS only)	Name: `xe-0/0/20` Forwarding class set: `be-pg` (Output) traffic control profile: `be-tcp`
Interface (port scheduling only)	Name: `xe-0/0/20` Scheduler map: `be-map`

NOTE: Only devices that support ETS hierarchical scheduling support forwarding class set and traffic control profile configuration. Direct port scheduling does not use the hierarchical scheduling structure.

Configuration

IN THIS SECTION

CLI Quick Configuration

To quickly configure the drop profile, scheduler with ECN enabled, and to map the scheduler to an output queue on an interface, copy the following commands, paste them in a text file, remove line breaks, change variables and details to match your network configuration, and then copy and paste the commands into the CLI at the `[edit]` hierarchy level.

ETS Quick Configuration

```
[edit class-of-service]
set drop-profile be-dp interpolate fill-level 30 fill-level 75 drop-probability 0 drop-
probability 80
set schedulers be-sched explicit-congestion-notification
set schedulers be-sched drop-profile-map loss-priority low protocol any drop-profile be-dp
set schedulers be-sched transmit-rate percent 25
set schedulers be-sched buffer-size percent 25
set schedulers be-sched priority low
set scheduler-maps be-map forwarding-class best-effort scheduler be-sched
set forwarding-class-sets be-pg class best-effort
set traffic-control-profiles be-tcp scheduler-map be-map
set interfaces xe-0/0/20 forwarding-class-set be-pg output-traffic-control-profile be-tcp
```

Port Scheduling Quick Configuration

```
[edit class-of-service]
set drop-profile be-dp interpolate fill-level 30 fill-level 75 drop-probability 0 drop-
probability 80
set schedulers be-sched explicit-congestion-notification
set schedulers be-sched drop-profile-map loss-priority low protocol any drop-profile be-dp
set schedulers be-sched transmit-rate percent 25
set schedulers be-sched buffer-size percent 25
set schedulers be-sched priority low
set scheduler-maps be-map forwarding-class best-effort scheduler be-sched
set interfaces xe-0/0/20 scheduler-map be-map
```

Configuring Static ECN

Step-by-Step Procedure

To configure Static ECN:

1. Configure the WRED packet drop profile `be-dp`. This example uses a drop start point of `30` percent, a drop end point of `75` percent, a minimum drop rate of `0` percent, and a maximum drop rate of `80` percent:

   ```
   [edit class-of-service]
   user@switch# set drop-profile be-dp interpolate fill-level 30 fill-level 75 drop-probability
   0 drop-probability 80
   ```

2. Create the scheduler `be-sched` with ECN enabled and associate the drop profile `be-dp` with the scheduler:

   ```
   [edit class-of-service]
   user@switch# set schedulers be-sched explicit-congestion-notification
   user@switch# set schedulers be-sched drop-profile-map loss-priority low protocol any drop-
   profile be-dp
   user@switch# set be-sched transmit-rate percent 25
   user be-sched transmit-rate percent 25
   user@switch# set be-sched buffer-size percent 25
   user@switch# set be-sched buffer-size percent 25
   user@switch# set be-sched priority low
   ```

3. Map the scheduler `be-sched` to the `best-effort` forwarding class (output queue 0) using scheduler map `be-map`:

   ```
   [edit class-of-service]
   user@switch# set scheduler-maps be-map forwarding-class best-effort scheduler be-sched
   ```

4. If you are using ETS, add the forwarding class `best-effort` to the forwarding class set `be-pg`; if you are using direct port scheduling, skip this step:

   ```
   [edit class-of-service]
   user@switch# set forwarding-class-sets be-pg class best-effort
   ```

5. If you are using ETS, associate the scheduler map `be-map`with the traffic control profile `be-tcp`; if you are using direct port scheduling, skip this step:

   ```
   [edit class-of-service]
   user@switch# set traffic-control-profiles be-tcp scheduler-map be-map
   ```

6. If you are using ETS, associate the traffic control profile `be-tcp` and the forwarding class set `be-pg` with the interface on which you want to enable ECN on the best-effort queue:

   ```
   [edit class-of-service]
   user@switch# set interfaces xe-0/0/20 forwarding-class-set be-pg output-traffic-control-
   profile be-tcp
   ```

   If you are using direct port scheduling, associate the scheduler map `be-map` with the interface on which you want to enable ECN on the best-effort queue:

   ```
   [edit class-of-service]
   user@switch# set interfaces xe-0/0/20 scheduler-map be-map
   ```

Configuring Dynamic ECN

Step-by-Step Procedure

To configure Dynamic ECN:

1. Follow the steps to configure Static ECN.

2. Enable the Dynamic ECN profile:

```
[edit class-of-service]
user@switch# set decn-profile profile-name queue queue number floor floor value offset offset
value
```

Floor and offset values range from 0 to 524287.

Floor and offset values do not need to be configured. If no value is selected, the maximum value of 524287 will be selected by default.

3. Assign the Dynamic ECN profile to an interface:

```
[edit class-of-service]
user@switch# set interfaces interface decn-profile profile-name
```

Verification

IN THIS SECTION

Verifying That ECN Is Enabled

Purpose

Verify that ECN is enabled in the scheduler `be-sched` by showing the configuration for the scheduler map `be-map`.

Action

Display the scheduler map configuration using the operational mode command `show class-of-service scheduler-map be-map`:

```
user@switch> show class-of-service scheduler-map be-map
Scheduler map: be-map, Index: 12240
```

```
Scheduler:be-sched, Forwarding class: best-effort, Index: 115
  Transmit rate: 25 percent, Rate Limit: none, Buffer size: 25 percent,
  Buffer Limit: none, Priority: low
  Excess Priority: unspecified, Explicit Congestion Notification: enable
  ECN Type: static/dynamic Offset offset-value Floor floor-value
  Drop profiles:
    Loss priority   Protocol    Index    Name
    Low             any          3312    be-dp
    Medium-high     any          1       <default-drop-profile>
    High            any          1       <default-drop-profile>
```

Meaning

The `show class-of-service scheduler-map` operational command shows the configuration of the scheduler associated with the scheduler map and the forwarding class mapped to that scheduler. The output shows that:

- The scheduler associated with the scheduler map is `be-sched`.
- The scheduler map applies to the forwarding class `best-effort` (output queue 0).
- The scheduler `be-sched` has a transmit rate of `25` percent, a queue buffer size of `25` percent, and a drop priority of `low`.
- Explicit congestion notification state is `enable`.
- The WRED drop profile used for low drop priority traffic is `be-dp`.

RELATED DOCUMENTATION

CoS Explicit Congestion Notification (ECN) | **533**

CHAPTER 10

Altering Outgoing Packet Headers Using Rewrite Rules

IN THIS CHAPTER

Rewriting Packet Headers to Ensure Forwarding Behavior

As packets enter or exit a network, edge routers might be required to alter the class-of-service (CoS) settings of the packets. *Rewrite rules* set the value of the CoS bits within the packet's header. Each rewrite rule reads the current forwarding class and loss priority information associated with the packet, locates the chosen CoS value from a table, and writes this CoS value into the packet header.

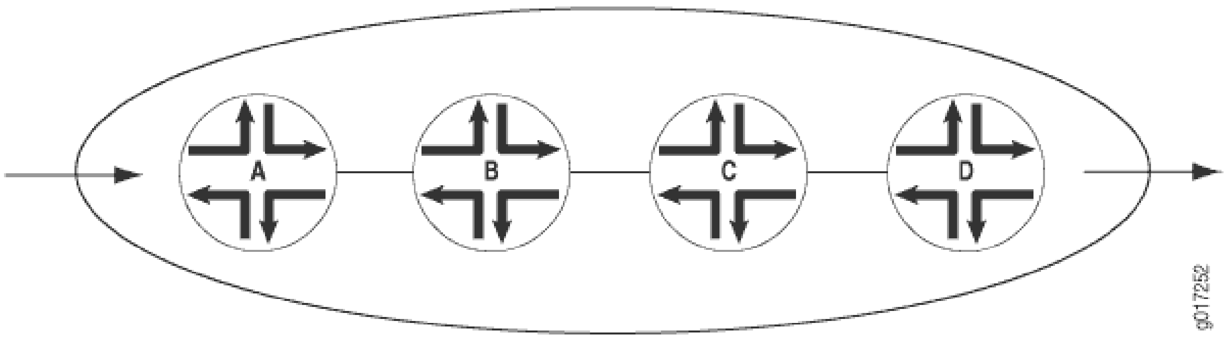
In effect, the rewrite rule performs the opposite function of the behavior aggregate (BA) classifier used when the packet enters the routing device. As the packet leaves the routing platform, the final CoS action is generally the application of a rewrite rule.

You configure rewrite rules to alter CoS values in outgoing packets on the outbound interfaces of an edge router to meet the policies of a targeted peer. This allows the downstream routing device in a neighboring network to classify each packet into the appropriate service group.

In addition, you often need to rewrite a given marker (IP precedence, Differentiated Services code point [DSCP], IEEE 802.1p, or MPLS EXP settings) at the inbound interfaces of an edge router to accommodate BA classification by core devices.

Figure 47 on page 559 shows a flow of packets through four routing devices. Router A rewrites the CoS bits in incoming packet to accommodate the BA classification performed by Routers B and C. Router D alters the CoS bits of the packets before transmitting them to the neighboring network.

Figure 47: Packet Flow Across the Network



For every incoming packet, the ingress classifier decodes the ingress CoS bits into a forwarding class and packet loss priority (PLP) combination. The egress CoS information depends on which type of rewrite marker is active, as follows:

- For Multiprotocol Label Switching (MPLS) EXP and IEEE 802.1 rewrite markers, values are derived from the forwarding class and PLP values in *rewrite rules*. MPLS EXP and IEEE 802.1 markers are not preserved because they are part of the Layer 2 encapsulation.
- For IP precedence and DiffServ code point (DSCP) rewrite markers, the marker alters the first three bits on the type-of-service (ToS) byte while leaving the last three bits unchanged.

To configure CoS rewrite rules, you define the rewrite rule and apply it to an interface. Include the following statements at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
interfaces {
    interface-name {
        unit logical-unit-number {
            rewrite-rules {
                dscp (rewrite-name | default)protocol protocol-types;
                dscp-ipv6 (rewrite-name | default);
                exp (rewrite-name | default)protocol protocol-types;
                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 protocol-types;
            }
        }
    }
}
rewrite-rules {
    (dscp | dscp-ipv6 | exp | frame-relay-de | ieee-802.1 | inet-precedence) rewrite-name {
        import (rewrite-name | default);
        forwarding-class class-name {
            loss-priority level code-point (alias | bits);
        }
    }
}
```

NOTE: You cannot apply a rewrite rule and output forwarding class map to the same logical interface (unit). Although a warning is issued, there is nothing in the CLI that prevents this configuration. An error message appears when you attempt to commit the configuration.

Applying Default Rewrite Rules

By default, rewrite rules are not usually applied to interfaces. If you want to apply a rewrite rule, you can either design your own rule and apply it to an interface, or you can apply a default rewrite rule.

NOTE: The lone exception is that non-MPC MPLS-enabled interfaces use the default EXP rewrite rule, even if not configured.

To apply default rewrite rules, include one or more of the following statements at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules]
dscp default;
dscp-ipv6 default;
exp default;
ieee-802.1 default vlan-tag (outer | outer-and-inner);
inet-precedence default;
```

Table 51 on page 562 shows the default rewrite rule mappings. These are based on the default bit definitions of DSCP, DSCP IPv6, EXP, IEEE, and IP CoS values, as shown in "Default Aliases for CoS Value Bit Patterns Overview" on page 71, and the default forwarding classes shown in "Default Forwarding Classes" on page 288.

When the software detects packets whose CoS values match the forwarding class and PLP values in the first two columns in Table 51 on page 562, the software maps the header bits of those packets to the code-point aliases in the last column in Table 51 on page 562. The code-point aliases in the last column map to the CoS bits shown in "Default Aliases for CoS Value Bit Patterns Overview" on page 71.

Table 51: Default Packet Header Rewrite Mappings

Map from Forwarding Class	Loss Priority	Map to DSCP/DSCP IPv6/ EXP/IEEE/IP
`expedited-forwarding`	`low`	`ef`
`expedited-forwarding`	`high`	`ef`
`assured-forwarding`	`low`	`af11`
`assured-forwarding`	`high`	`af12` (DSCP/DSCP IPv6/EXP)
`best-effort`	`low`	`be`
`best-effort`	`high`	`be`
`network-control`	`low`	`nc1/cs6`
`network-control`	`high`	`nc2/cs7`

Starting with Junos OS Release 21.1, PTX routers support two forwarding classes and four loss priorities for MPLS EXP default rewrite rules, as Table 52 on page 562 shows.

Table 52: Default MPLS EXP Rewrite Mapping for PTX Routers

Map from Forwarding Class	Loss Priority	Map to MPLS EXP Bits
`best-effort`	`low`	`000`
`best-effort`	`high`	`001`
`best-effort`	`medium-low`	`010`
`best-effort`	`medium-high`	`011`

Table 52: Default MPLS EXP Rewrite Mapping for PTX Routers *(Continued)*

Map from Forwarding Class	Loss Priority	Map to MPLS EXP Bits
network-control	medium-low	100
network-control	medium-high	101
network-control	low	110
network-control	high	111

In the following example, the so-1/2/3.0 interface is assigned the default DSCP rewrite rule. One result of this configuration is that each packet exiting the interface with the expedited-forwarding forwarding class and the high or low loss priority has its DSCP bits rewritten to the DSCP ef code-point alias. "Default Aliases for CoS Value Bit Patterns Overview" on page 71 shows that this code-point alias maps to the 101110 bits.

Another result of this configuration is that all packets exiting the interface with the best-effort forwarding class and the high or low loss priority have their EXP bits rewritten to the EXP be code-point alias."Default Aliases for CoS Value Bit Patterns Overview" on page 71 shows that this code-point alias maps to the 000 bits.

To evaluate all the implications of this example, see "Default Aliases for CoS Value Bit Patterns Overview" on page 71 and Table 51 on page 562.

```
class-of-service {
    interfaces {
        so-1/2/3 {
            unit 0 {
                rewrite-rules {
                    dscp default;
                }
            }
        }
    }
}
```

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 60

Configuring Rewrite Rules

IN THIS SECTION

You define markers in the rewrite rules section of the CoS configuration hierarchy and reference them in the logical interface configuration. This model supports marking on the DSCP, DSCP IPv6, IP precedence, IPv6 precedence, IEEE 802.1, and MPLS EXP CoS values.

To configure a rewrite-rules mapping and associate it with the appropriate forwarding class and code-point alias or bit set, include the `rewrite-rules` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
rewrite-rules {
    (dscp | dscp-ipv6 | exp | ieee-802.1 |ieee-802.1ad | inet-precedence | inet6-precedence)
rewrite-name {
        import (rewrite-name | default);
        forwarding-class class-name {
            loss-priority level code-point (alias | bits);
        }
    }
}
```

The rewrite rule sets the code-point aliases and bit patterns for a specific forwarding class and packet loss priority (PLP). The inputs for the map are the forwarding class and the PLP. The output of the map is the code-point alias or bit pattern. For more information about how CoS maps work, see "Mapping CoS Component Inputs to Outputs" on page 9.

By default, IP precedence rewrite rules alter the first three bits on the type-of-service (ToS) byte while leaving the last three bits unchanged. This default behavior is not configurable. The default behavior applies to rules you configure by including the `inet-precedence` statement at the `[edit class-of-service rewrite-rules]` hierarchy level. The default behavior also applies to rewrite rules you configure for MPLS packets with IPv4 payloads. You configure these types of rewrite rules by including the `mpls-inet-both` or `mpls-inet-both-non-vpn` option at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules exp` *`rewrite-rule-name`* `protocol]` hierarchy level.

NOTE: The IRB classifiers and rewrite rules are used only for *routed* packets; in other words, it is for traffic that originated in the Layer 2 domain and is then routed through IRB into the Layer 3 domain, or vice versa. Only IEEE classifiers and IEEE rewrite rules are allowed for pure Layer 2 interfaces within a bridge domain.

NOTE: The forwarding class and loss priority are determined by ingress classification.

Platform-Specific Rewrite Rules Behavior

Use the following table to review platform-specific behaviors for your platforms.

Table 53: Platform-Specific Rewrite Rules Behavior

Platform	Difference
EX Series	• On EX Series switches, if you configure `vlan-vpls` encapsulation and add an IEEE 802.1 header on a Gigabit Ethernet or 10 Gigabit Ethernet interface to output traffic, but do not apply an IEEE 802.1 rewrite rule, then the default IEEE 802.1 rewrite rule is ignored and the IEEE 802.1p bits are set to match the forwarding class queue.

Table 53: Platform-Specific Rewrite Rules Behavior *(Continued)*

Platform	Difference
MX Series	• MX Series routers with MPCs support rewrite rules that rewrite the first three bits of the IPv6 DSCP value through the `inet6-precedence` statement at the `[edit class-of-service rewrite-rules]` hierarchy level. This allows you to set a 3-bit code point for a particular forwarding class and loss priority for IPv6 traffic. This rewrite rule option can also be applied to packets entering an MPLS LSP by including the protocol `mpls statement` when applying the rewrite rule to a logical interface. • Integrated Bridging and Routing (IRB) interfaces are used to tie together Layer 2 switched and Layer 3 routed domains on MX routers. MX routers support classifiers and rewrite rules on the IRB interface at the `[edit class-of-service interfaces irb unit logical-unit-number]` level of the hierarchy. All types of classifiers and rewrite rules are allowed, including IEEE 802.1p. • On MX960 routers, if you configure `vlan-vpls` encapsulation and add an IEEE 802.1 header on a Gigabit Ethernet or 10 Gigabit Ethernet interface to output traffic, but do not apply an IEEE 802.1 rewrite rule, then the default IEEE 802.1 rewrite rule is ignored and the IEEE 802.1p bits are set to match the forwarding class queue.
PTX Series	• The `inet-precedence` statement is supported on PTX Series routers only when network services is set to `enhanced-mode`. For more information, see `enhanced-mode (Network Services)`.

RELATED DOCUMENTATION

Applying Rewrite Rules to Output Logical Interfaces | 577

Configuring Rewrite Rules Based on PLP

Rewrite rules take action on outgoing packets. When tricolor marking (TCM) is enabled, routers support four rewrite packet loss priority (PLP) designations: `low`, `medium-low`, `medium-high`, and `high`. To include the PLP for a rewrite rule, include the following statements at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
rewrite-rules {
    (dscp | dscp-ipv6 | exp | ieee-802.1 | inet-precedence) rewrite-name                    {
        import (rewrite-name | default);
        forwarding-class class-name {
            loss-priority (low | medium-low | medium-high | high) code-point (alias | bits);
        }
    }
}
```

In Junos OS, rewrite rules only look at the forwarding class and packet loss priority of the packet (as assigned by a behavior aggregate or multifield classifier at ingress), not at the incoming CoS value, to determine the CoS value to write to the packet header at egress. The inputs for a rewrite rule are the forwarding class and the PLP. The output for a rewrite rule are the CoS values. In other words, a rewrite rule sets the CoS value for each packet exiting the interface with a specified forwarding class and PLP.

For example, if you configure the following, the `000000` CoS value is assigned to all packets exiting the interface with the `assured-forwarding` forwarding class and `medium-high` PLP:

```
class-of-service {
    rewrite-rules {
        dscp dscp-rw {
            forwarding-class assured-forwarding {
                loss-priority medium-high code-point 000000;
            }
        }
    }
}
```

To use this classifier, you must configure the settings for the `assured-forwarding` forwarding class at the `[edit class-of-service forwarding-classes queue queue-number assured-forwarding]` hierarchy level. For more information, see "Understanding How Forwarding Classes Assign Classes to Output Queues" on page 286.

Applying IEEE 802.1p Rewrite Rules to Dual VLAN Tags

IN THIS SECTION

By default, when you apply an IEEE 802.1p rewrite rule to an output logical interface, the software rewrites the IEEE bits in the outer VLAN tag only.

For Gigabit Ethernet IQ2 PICs, 10-port 10-Gigabit OSE PICs, and 10-Gigabit Ethernet IQ2 PICs only, you can rewrite the IEEE bits in both the outer and inner VLAN tags of the tagged Ethernet frames. When you enable class of service (CoS) rewrite for both tags, the same IEEE 802.1p rewrite table is used for the inner and outer VLAN tag.

For IQ PICs, you can only configure one IEEE 802.1 rewrite rule on a physical port. All logical ports (units) on that physical port should apply the same IEEE 802.1 rewrite rule.

To rewrite both the outer and inner VLAN tags, include the `vlan-tag outer-and-inner` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules ieee-802.1 (`*`rewrite-name`* `| default)]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules
ieee-802.1 (rewrite-name | default)]
vlan-tag outer-and-inner;
```

To explicitly specify the default behavior, include the `vlan-tag outer` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules ieee-802.1 (`*`rewrite-name`* `| default)]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules
ieee-802.1 (rewrite-name | default)]
vlan-tag outer;
```

For more information about VLAN tags, see the Junos OS Network Interfaces Library for Routing Devices.

On MX routers and EX Series switches, you can perform IEEE 802.1p and DEI rewriting based on forwarding class and PLP at the VPLS ingress PE. You rewrite (mark) the IEEE 802.1p or DEI bits on frames at the VPLS ingress PE based on the value of the forwarding class and PLP established for the traffic. You can rewrite either the outer tag only or the outer and inner tag. When both tags are rewritten, both get the same value. To configure these rewrite rules, include the `ieee-802.1` statement at the `[edit class-of-services routing-instance` *`routing-instance-name`* `rewrite-rules]` hierarchy level.

NOTE: For MX80, MX240, MX480, and MX960 routers with MPC/MICs, rewrite on LSI interfaces is not supported (the routers, with DPC, do support rewrite on LSI interfaces).

On routing devices with IQ2 or IQ2-E PICs, you can perform IEEE 802.1p and DEI rewriting based on forwarding-class and packet loss priority (PLP) at the VPLS ingress provider edge (PE) router. You rewrite (mark) the IEEE 802.1p or DEI bits on frames at the VPLS ingress PE based on the value of the forwarding-class and PLP established for the traffic. You can rewrite either the outer tag only or both the outer and inner tags. When both tags are rewritten, both get the same value.

NOTE: The 10-port 10-Gigabit OSE PIC does not support DEI rewriting based on forwarding class and PLP at the VPLS ingress PE.

To configure these rewrite rules, include the `ieee-802.1` statement at the `[edit class-of-services routing-instance` *`routing-instance-name`* `rewrite-rules]` hierarchy level.

Example: Applying an IEEE 802.1p Rewrite Rule to Dual VLAN Tags

Apply the `ieee8021p-rwrule1` rewrite rule to both inner and outer VLAN tags of Ethernet-tagged frames exiting the `ge-0/0/0.0` interface:

```
class-of-service {
    interfaces {
        ge-0/0/0 {
            unit 0 {
                rewrite-rules {
                    ieee-802.1 ieee8021p-rwrule1 vlan-tag outer-and-inner;
                }
            }
        }
    }
}
```

Applying IEEE 802.1ad Rewrite Rules to Dual VLAN Tags

IN THIS SECTION

By default, when you apply an IEEE 802.1ad rewrite rule to an output logical interface, the software rewrites the IEEE bits in the outer VLAN tag only.

For MX Series routers and IQ2 PICs, you can rewrite the IEEE 802.1ad bits in both the outer and inner VLAN tags of the tagged Ethernet frames. When you enable the CoS rewrite for both tags, the same IEEE 802.1ad rewrite table is used for the inner and outer VLAN tag.

NOTE: When you apply IEEE 802.1ad rewrite rules for inner and outer VLAN tags, you cannot rewrite the Canonical Format Identifier (CFI) bit for the inner VLAN tag.

To rewrite both the outer and inner VLAN tags, include the `vlan-tag outer-and-inner` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules ieee-802.1ad (`*`rewrite-name`* `| default)]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules
ieee-802.1ad (rewrite-name | default)]
vlan-tag outer-and-inner;
```

To explicitly specify the default behavior, include the `vlan-tag outer` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules ieee-802.1ad (`*`rewrite-name`* `| default)]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules
ieee-802.1ad (rewrite-name | default)]
vlan-tag outer;
```

For more information about VLAN tags, see the Junos OS Network Interfaces Library for Routing Devices.

Example: Applying an IEEE 802.1ad Rewrite Rule to Dual VLAN Tags

Apply the `dot1p_dei_rw` rewrite rule to both inner and outer VLAN tags of Ethernet-tagged frames exiting the `ge-2/0/0.0` interface:

```
class-of-service {
    interfaces {
        ge-2/0/0 {
            unit 0 {
                rewrite-rules {
                    ieee-802.1ad dot1p_dei_rw vlan-tag outer-and-inner;
                }
            }
        }
    }
}
```

Rewriting IEEE 802.1p Packet Headers with an MPLS EXP Value

For Ethernet interfaces a Juniper device that have a peer connection to another Juniper device, you can rewrite both MPLS EXP and IEEE 802.1p bits to a configured value. This enables you to pass the configured value to the Layer 2 VLAN path.

To rewrite both the MPLS EXP and IEEE 802.1p bits, you must include EXP and IEEE 802.1p rewrite rules in the interface configuration. To configure EXP and IEEE 802.1p rewrite rules, include the `rewrite-rules` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]` hierarchy level, specifying the `exp` and `ieee-802.1` options:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
rewrite-rules {
    exp rewrite-rule-name;
    ieee-802.1 default;
}
```

When you combine these two rewrite rules, only the EXP rewrite table is used for rewriting packet headers. If you do not configure a VLAN on the interface, only the EXP rewriting is in effect. If you do not configure an LSP on the interface or if the MPLS EXP rewrite rule mapping is removed, the IEEE 802.1p default rewrite rules mapping takes effect.

NOTE: You can also combine other rewrite rules. IP, DSCP, DSCP IPv6, and MPLS EXP are associated with Layer 3 packet headers, and IEEE 802.1p is associated with Layer 2 packet headers.

If you combine IEEE 802.1p with IP rewrite rules, the Layer 3 packets and Layer 2 headers are rewritten with the IP rewrite rule.

If you combine IEEE 802.1p with DSCP or DSCP IPv6 rewrite rules, three bits of the Layer 2 header and six bits of the Layer 3 packet header are rewritten with the DSCP or DSCP IPv6 rewrite rule.

NOTE: For MPCs, default EXP rewrite rules do not exist for logical interfaces. The EXP CoS bits for MPLS labels are obtained from the IP precedence bits for IP traffic. The EXP bits for labels that are pushed or swapped are inherited from the current label of the MPLS packets. For non-IP and non-MPLS packets, the EXP bits are set to 0. If a custom EXP rewrite rule is configured on the core-facing interface, then it overrides the EXP bits.

The following example shows how to configure an EXP rewrite rule and apply it to both MPLS EXP and IEEE 802.1p bits:

```
[edit class-of-service]
rewrite-rules {
    exp exp-ieee-table {
        forwarding-class best-effort {
            loss-priority low code-point 000;
            loss-priority high code-point 001;
        }
        forwarding-class assured-forwarding {
            loss-priority low code-point 010;
            loss-priority high code-point 011;
        }
        forwarding-class expedited-forwarding {
            loss-priority low code-point 111;
            loss-priority high code-point 110;
        }
        forwarding-class network-control {
            loss-priority low code-point 100;
            loss-priority high code-point 101;
        }
    }
}
interfaces {
    xe-3/1/0 {
        unit 0 {
            rewrite-rules {
                exp exp-ieee-table;
                ieee-802.1 default;
            }
        }
    }
}
```

Setting IPv6 DSCP and MPLS EXP Values Independently

You can set the DSCP and MPLS EXP bits independently on IPv6 packets. To enable this feature, include the `protocol mpls` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules dscp-ipv6` *`rewrite-name`*`]` hierarchy level.

You can set DSCP IPv6 values only at the ingress MPLS node.

The following limitations apply to this feature:

- MPLS packets entering another MPLS tunnel at the ingress node may mark only the EXP value if EXP *rewrite rules* are configured, but not the DSCP value in the IPv6 header.
- This feature does not support MPLS packets generated by the Routing Engine.
- The IP precedence field is not applicable for IPv6, and is not supported.

RELATED DOCUMENTATION

Configuring DSCP Values for IPv6 Packets Entering the MPLS Tunnel | **573**

Configuring DSCP Values for IPv6 Packets Entering the MPLS Tunnel

The following configuration example explains in detail how to set the DSCP and MPLS EXP bits independently on IPv6 packets.

1. Configure the router device (ingress PE router) to classify (behavior aggregate or multifield) the incoming packets to a particular forwarding class.

```
[edit firewall]
family inet6 {
    filter ss-v6filt {
        term ss-vpn {
            from {
                destination-address {
                    ::ffff:192.0.2.128/120;
                }
            }
                then {
                loss-priority low;
                forwarding-class ss-fc;
            }
        }
    }
}
```

In the preceding example, the ingress FPC classifies (MF) incoming IPv6 packets destined for address "::ffff:192.0.2.128/120" to forwarding class "ss-fc" and loss priority "low."

2. Attach the preceding firewall filter to an interface. Because you are matching on inbound traffic, this would be an input filter. This classifies all traffic on the interface "ge-2/1/0" that matches the filter "ss-v6."

```
[edit interfaces]
ge-2/1/0 {
    hierarchical-scheduler;
    vlan-tagging;
    unit 300 {
        family inet6 {
            filter {
                input ss-v6filt;
            }
            address ::ffff:192.0.2.100/120;
        }
    }
}
```

3. Configure the DSCP–IPv6 rewrite rule for the forwarding class “ss-fc.” This causes the outgoing IPv6 packets belonging to the forwarding class “ss-fc” and loss priority “low” to have their DSCP value rewritten to 100000.

```
[edit class-of-service rewrite-rules]
dscp-ipv6 ss-v6dscp {
    forwarding-class ss-fc {
        loss-priority low code-point 100000;
    }
}
```

4. Configure the EXP rewrite values for the forwarding class “ss-fc.” This rewrite rule stamps an EXP value of 100 on all outgoing MPLS packets assigned to the forwarding class “ss-fc” and loss priority “low.”

```
[edit class-of-service rewrite-rules]
exp ss-exp {
    forwarding-class ss-fc {
        loss-priority low code-point 100;
    }
}
```

5. Apply the preceding rewrite rule to an egress interface. On the egress FPC, all IPv6 packets in the forwarding class “ss-fc” with loss priority “low” are marked with the DSCP value “100000” and an EXP value of “100” before they enter the MPLS tunnel.

```
[edit class-of-service interfaces]
ge-2/1/1 {
    unit 10 {
        rewrite-rules {
            dscp-ipv6 ss-v6dscp protocol mpls;
            exp ss-exp;
        }
    }
}
```

6. To support IPv4 DSCP and MPLS EXP independent rewrite at the same time, you can define and apply an IPv4 DSCP rewrite rule "ss-dscp" to the same interface.

```
[edit class-of-service interfaces]
ge-2/1/1 {
    unit 10 {
        rewrite-rules {
            dscp ss-dscp protocol mpls;
            dscp-ipv6 ss-v6dscp protocol mpls;
            exp ss-exp;
        }
    }
}
```

RELATED DOCUMENTATION

Setting IPv6 DSCP and MPLS EXP Values Independently | 573

Setting Ingress DSCP Bits for Multicast Traffic over Layer 3 VPNs

By default, the DSCP bits on outer IP headers arriving at an ingress PE router using generic routing encapsulation (GRE) are not set for multicast traffic sent over an Layer 3 virtual private network (VPN) provider network. However, you can configure a type-of-service (ToS) rewrite rule so the router sets the DSCP bits of GRE packets to be consistent with the service provider's overall core network CoS policy. The bits are set at the core-facing interface of the ingress provider edge (PE) router. For more information about rewriting IP header bits, see "Rewriting Packet Headers to Ensure Forwarding Behavior" on page 558.

This section describes this configuration from a CoS perspective. The examples are not complete multicast or VPN configurations. For more information about multicast, see the Junos OS Multicast Protocols User Guide. For more information about Layer 3 VPNs, see the Junos OS VPNs Library for Routing Devices.

To configure the rewrite rules on the core-facing interface of the ingress PE, include the `rewrite-rules` statement at the `[edit class-of-service]` hierarchy level. You apply the rule to the proper ingress interface at the `[edit class-of-service interfaces]` hierarchy level to complete the configuration. This ingress DSCP rewrite is independent of classifiers placed on ingress traffic arriving on the customer-facing interface of the PE router.

The rewrite rules are applied to all unicast packets and multicast groups. You cannot configure different rewrite rules for different multicast groups. The use of DSCPv6 bits is not supported because IPv6 multicast is not supported. You can configure another rewrite rule for the EXP bits on MPLS CE-CE unicast traffic.

This example defines a rewrite rule called `dscp-rule` that establishes a value of `000000` for best-effort traffic. The rule is applied to the outgoing, core-facing PE interface `ge-2/3/0`.

```
[edit class-of-service]
rewrite-rules {
    dscp dscp-rule {
        forwarding-class best-effort {
            loss-priority low code-point 000000;
        }
    }
}

[edit class-of-service interfaces]
ge-2/3/0 {
    unit 0 {
        rewrite-rules {
            dscp dscp-rule;
        }
    }
}
```

Applying Rewrite Rules to Output Logical Interfaces

IN THIS SECTION

To assign the rewrite-rules configuration to the output logical interface, include the `rewrite-rules` statement at the `[edit class-of-service interfaces *interface-name* unit *logical-unit-number*]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
rewrite-rules {
    dscp (rewrite-name | <default>) protocol protocol-types;
    dscp-ipv6 (rewrite-name | <default>) protocol protocol-types;
    exp (rewrite-name | <default>) protocol protocol-types;
    exp-push-push-push <default>;
    exp-swap-push-push <default>;
    ieee-802.1 (rewrite-name | <default>) inet-prec vlan-tag (outer | outer-and-inner);
    inet-precedence (rewrite-name | <default>) protocol protocol-types;
    inet6-precedence (rewrite-name | <default>) protocol protocol-types;
}
```

You can combine the `dscp` or `inet-prec` and `exp` options to set the DSCP or IP precedence bits and MPLS EXP bits independently on IP packets entering an MPLS tunnel.

Logical interfaces do not support multiple `dscp` rewrite rules or multiple `dscp-ipv6` rewrite rules for the same protocol.

In the following example, the DSCP bits specified in `ss-dscp` are applied to packets entering the MPLS tunnel on `ge-2/1/1`, and the DSCP bits specified in `ss-v6dscp` are applied to IPv6 packets. The EXP bits are set to the bit configuration specified in `ss-exp`:

```
[edit class-of-service interfaces]
ge-2/1/1
    unit 10 {
        rewrite-rules {
            dscp ss-dscp protocol mpls; # Applies to IPv4 packets entering MPLS tunnel
            dscp-ipv6 ss-v6dscp protocol mpls; # Applies to IPv6 packets entering MPLS tunnel
            exp ss-exp; # Sets label EXP bits independently
        }
    }
}
```

You can use interface wildcards for *`interface-name`* and *`logical-unit-number`*. You can also include Layer 2 and Layer 3 rewrite information in the same configuration.

Platform-Specific Behavior

Use the following table to review platform-specific behaviors for your platforms.

Table 54: Platform-Specific Behavior

Platform	Difference
ACX5448	ACX5448 supports rewrite on services such as: • L2 (VLAN, VLAN-CCC - Priority Code Point (PCP)) • L3 (IPv4 and IPv6 - Differentiated Services Code Point (DSCP)) • MPLS (EXP - Experimental Bits) For L2, rewrite information resides in the VLAN translation entry at the egress side. Without that entry, rewrite does not take place. For VLAN-CCC (local cross-connect), the rewrite is supported by enabling VLAN translation entry in the same manner as that for L2.

RELATED DOCUMENTATION

Setting IPv6 DSCP and MPLS EXP Values Independently | 573
Configuring DSCP Values for IPv6 Packets Entering the MPLS Tunnel | 573

Rewriting MPLS and IPv4 Packet Headers

IN THIS SECTION

You can apply a rewrite rule to MPLS and IPv4 packet headers simultaneously. This allows you to initialize MPLS EXP and IP precedence bits at LSP ingress. You can configure different rewrite rules depending on whether the traffic is VPN or non-VPN.

Devices running Junos OS Evolved do not support different rewrite rules for VPN and non-VPN traffic.

The default MPLS EXP rewrite rules are shown in Table 55 on page 580.

Table 55: Default MPLS EXP Rewrite Rules

Forwarding Class	Loss Priority	MPLS EXP Rewrite Value
`best-effort`	`low`	000
`best-effort`	`high`	001
`expedited-forwarding`	`low`	010
`expedited-forwarding`	`high`	011
`assured-forwarding`	`low`	100
`assured-forwarding`	`high`	101
`network-control`	`low`	110
`network-control`	`high`	111

By default, IP precedence rewrite rules alter the first three bits on the type-of-service (ToS) byte while leaving the last three bits unchanged. This default behavior applies to rewrite rules you configure for MPLS packets with IPv4 payloads on provider edge (PE) routers only. On transit routers (P), we do not alter the inner IPv4 headers and payloads while setting EXP bits in the outer MPLS header.

To override the default MPLS EXP rewrite table and rewrite MPLS and IPv4 packet headers simultaneously, include the `protocol` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules exp` *`rewrite-rule-name`*`]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules exp
rewrite-rule-name]
protocol protocol-types;
```

The `protocol` statement defines the types of MPLS packets and packet headers to which the specified rewrite rule is applied. The MPLS packet can be a standard MPLS packet or an MPLS packet with an IPv4 payload. Specify the type of MPLS packet using the following options:

- `mpls-any`—Applies the rewrite rule to MPLS packets and writes the CoS value to MPLS headers.
- `mpls-inet-both`—Applies the CoS value to the MPLS and IPv4 headers of VPN packets on provider edge (PE) routers. On core routers, this option rewrites only the MPLS header and does not rewrite CoS bits in the inner IPv4 payload.
- `mpls-inet-both-non-vpn`—Applies the rewrite rule to non-VPN MPLS packets with IPv4 payloads. Applies the CoS value to the MPLS and IPv4 headers of non-VPN packets on provider edge (PE) routers. On core routers, this option rewrites only the MPLS header and does not rewrite CoS bits in the inner IPv4 payload.

Junos OS Evolved supports only the `mpls-any` option. Devices running Junos OS Evolved do not support different rewrite rules for VPN and non-VPN traffic.

On MX Series routers, you can perform simultaneous DSCP and EXP rewrite by attaching independent DSCP or IPv4 precedence rewrite rules and EXP rewrite rules to the same core interface. Thus, you can rewrite both code points (DSCP and EXP) when the packet is received by the ingress provider edge (PE) router on the MPLS core.

An alternative to overwriting the default with a rewrite-rules mapping is to configure the default packet header rewrite mappings, as discussed in "Applying Default Rewrite Rules" on page 561.

By default, IP precedence rewrite rules alter the first three bits on the ToS byte while leaving the last three bits unchanged. This default behavior is not configurable. The default behavior applies to rules you configure by including the `inet-precedence` statement at the `[edit class-of-service rewrite-rules]` hierarchy level. The default behavior also applies to rewrite rules you configure for MPLS packets with IPv4 payloads. You configure these types of rewrite rules by including the `mpls-inet-both` or `mpls-inet-both-non-vpn` option at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules exp` *`rewrite-rule-name`* `protocol]` hierarchy level.

Example: Rewriting MPLS and IPv4 Packet Headers

Configure rewrite tables and apply them in various ways to achieve the following results:

- For interface `et-3/1/0`, the three EXP rewrite tables are applied to packets, depending on the protocol of the payload:
  - IPv4 packets (VPN) that enter the LSPs on interface `et-3/1/0` are initialized with values from rewrite table `exp-inet-table`. An identical 3-bit value is written into the IP precedence and MPLS EXP bit fields if this router acts as a provider edge router only. For a transit router, the 3-bit value is written to MPLS EXP bits only.
  - IPv4 packets (non-VPN) that enter the LSPs on interface `et-3/1/0` are initialized with values from rewrite table `rule-non-vpn`. An identical 3-bit value is written into the IP precedence and MPLS EXP bit fields if this router acts as a provider edge router only. For a transit router, the 3-bit value is written to MPLS EXP bits only.
  - Non-IPv4 packets that enter the LSPs on interface `et-3/1/0` are initialized with values from rewrite table `rule1`, and written into the MPLS EXP header field only. The statement `exp rule1` has the same result as `exp rule1 protocol mpls`.
- For interface `et-3/1/0`, IPv4 packets transmitted over a non-LSP layer are initialized with values from IP precedence rewrite table `rule2`.
- For interface `et-3/1/1`, IPv4 packets that enter the LSPs are initialized with values from EXP rewrite table `exp-inet-table`. An identical 3-bit value is written into the IP precedence and MPLS EXP bit fields if this router acts as a provider edge router only. For a transit router, the 3-bit value is written to MPLS EXP bits only.
- For interface `et-3/1/1`, MPLS packets other than IPv4 Layer 3 types are also initialized with values from table `exp-inet-table`. For VPN MPLS packets with IPv4 payloads, the CoS value is written to MPLS and IPv4 headers. For VPN MPLS packets without IPv4 payloads, the CoS value is written to MPLS headers only. Note that IPv4 headers are rewritten only when the router acts as a provider edge router.

```
[edit class-of-service]
rewrite-rules {
    exp exp-inet-table {
        forwarding-class best-effort {
            loss-priority low code-point 000;
            loss-priority high code-point 001;
        }
        forwarding-class assured-forwarding {
            loss-priority low code-point 010;
            loss-priority high code-point 011;
        }
        forwarding-class expedited-forwarding {
            loss-priority low code-point 111;
            loss-priority high code-point 110;
        }
        forwarding-class network-control {
            loss-priority low code-point 100;
            loss-priority high code-point 101;
        }
    }
    exp rule1 {
        ...
    }
    inet-precedence rule2 {
        ...
    }
}
exp rule_non_vpn {
    ...
}

interfaces {
    et-3/1/0 {
        unit 0 {
            rewrite-rules {
                exp rule1;
                inet-precedence rule2;
                exp exp-inet-table protocol mpls-inet-both; # For all VPN traffic.
                exp rule_non_vpn protocol mpls-inet-both-non-vpn; # For all non-VPN
# traffic.
            }
        }
    }
    et-3/1/1 {
        unit 0 {
            rewrite-rules {
                exp exp-inet-table protocol [mpls mpls-inet-both];
            }
        }
    }
}
```

Example: Simultaneous DSCP and EXP Rewrite

Configure the simultaneous DSCP and EXP rewrite rules as shown below:

1. Configure CoS.

```
[edit]
user@host# edit class-of-service
```

2. Configure the EXP rewrite rule on the interface.

```
[edit class-of-service]
user@host# set interfaces ge-2/0/3 unit 0 rewrite-rule exp rule1
```

3. Configure the IPv4 rewrite rule on the interface.

```
[edit class-of-service]
user@host# set interfaces ge-2/0/3 unit 0 rewrite-rule inet-precedence rule2
```

4. Configure the IPv4 rewrite rule on the interface and apply it to packets entering the MPLS tunnel.

```
[edit class-of-service]
user@host# set interfaces ge-2/0/3 unit 0 rewrite-rule inet-precedence rule3 protocol mpls
```

5. Verify the configuration by using the `show interfaces` command.

```
[edit class-of-service]
user@host# show interfaces ge-2/0/3 unit 0
rewrite-rules {
exp rule1;
inet-precedence rule2;
inet-precedence rule3 protocol mpls;
}
```

In the example above, there are two different IPv4 precedence rewrite rules: `rule2` and `rule3`. `rule2` affects the IPv4 to IPv4 traffic and `rule3` affects the IPv4 to MPLS traffic.

Rewriting the EXP Bits of All Three Labels of an Outgoing Packet

IN THIS SECTION

In interprovider, carrier-of-carrier, and complex traffic engineering scenarios, it is sometimes necessary to push three labels on the next hop, using a swap-push-push or triple-push operation.

By default, the top MPLS EXP label of an outgoing packet is not rewritten when you configure swap-push-push and triple-push operations. You can rewrite the EXP bits of all three labels of an outgoing packet, thereby maintaining the CoS of an incoming MPLS or non-MPLS packet.

When the software performs a swap-push-push operation and no rewriting is configured, the EXP fields of all three labels are the same as in the old label. If there is EXP rewriting configured, the EXP bits of the bottom two labels are overwritten with the table entry. The EXP setting of the top label is retained even with rewriting.

To push three labels on all incoming MPLS packets, include the `exp-swap-push-push default` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules]
exp-swap-push-push default;
```

When the software performs a push-push-push operation and if no rewriting is configured, the EXP fields of the bottom two labels are zero. If EXP rewriting is configured, the EXP fields of the bottom two labels are rewritten with the table entry's rewrite value. The EXP field of the top label is inserted with the Qn+PLP value. This Qn reflects the final classification by a multifield classifier if one exists, regardless of whether rewriting is configured.

To push three labels on incoming non-MPLS packets, include the `exp-push-push-push default` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules]
exp-push-push-push default;
```

These configurations apply the default MPLS EXP rewrite table, as described in "Rewriting MPLS and IPv4 Packet Headers" on page 579. You can configure these operations and override the default MPLS EXP rewrite table with a custom table. For more information about writing and applying a custom rewrite table, see "Configuring Rewrite Rules" on page 564 and "Applying Rewrite Rules to Output Logical Interfaces" on page 577.

NOTE: With a three-label stack, if you do not include the `exp-swap-push-push default` or `exp-push-push-push default` statement in the configuration, the top label's EXP bits are set to zero.

Example: Rewriting the EXP Bits of All Three Labels of an Outgoing Packet

Configure a swap-push-push operation, and override the default rewrite table with a custom table:

```
[edit class-of-service]
forwarding-classes {
    queue 0 be;
    queue 1 ef;
    queue 2 af;
    queue 3 nc;
}
interfaces {
    xe-1/1/3 {
        unit 0 {
            rewrite-rules {
                exp exp_rew; # Apply custom rewrite table
                exp-swap-push-push default;
            }
        }
    }
}
rewrite-rules {
    exp exp_rew {
        forwarding-class be {
            loss-priority low code-point 000;
            loss-priority high code-point 100;
        }
        forwarding-class ef {
            loss-priority low code-point 001;
            loss-priority high code-point 101;
        }
        forwarding-class af {
            loss-priority low code-point 010;
            loss-priority high code-point 110;
        }
        forwarding-class nc {
            loss-priority low code-point 011;
            loss-priority high code-point 111;
        }
    }
}
```

RELATED DOCUMENTATION

Rewriting MPLS and IPv4 Packet Headers | 579

Configuring Rewrite Rules | 564

Applying Rewrite Rules to Output Logical Interfaces | 577

Example: Per-Node Rewriting of EXP Bits

To configure a custom table to rewrite the EXP bits, also known as CoS bits, on a particular node, the classifier table and the rewrite table must specify exactly the same CoS values.

In addition, the least significant bit of the CoS value itself must represent the PLP value. For example, CoS value `000` must be associated with PLP `low`, `001` must be associated with PLP `high`, and so forth.

This example configures a custom table to rewrite the EXP bits on a particular node:

```
[edit class-of-service]
classifiers {
    exp exp-class {
        forwarding-class be {
            loss-priority low code-points 000;
            loss-priority high code-points 001;
        }
        forwarding-class af {
            loss-priority low code-points 010;
            loss-priority high code-points 011;
        }
        forwarding-class ef {
            loss-priority low code-points 100;
            loss-priority high code-points 101;
        }
        forwarding-class nc {
            loss-priority low code-points 110;
            loss-priority high code-points 111;
        }
    }
}
rewrite-rules {
    exp exp-rw {
        forwarding-class be {
            loss-priority low code-point 000;
            loss-priority high code-point 001;
        }
        forwarding-class af {
            loss-priority low code-point 010;
            loss-priority high code-point 011;
        }
        forwarding-class ef {
            loss-priority low code-point 100;
            loss-priority high code-point 101;
        }
        forwarding-class nc {
            loss-priority low code-point 110;
            loss-priority high code-point 111;
        }
    }
}
```

Example: Rewriting CoS Information at the Network Border to Enforce CoS Strategies

IN THIS SECTION

This example shows how to rewrite (remark) class-of-service (CoS) values at the network border to enforce your internal CoS strategies. This is typically done when the CoS values of the inbound traffic at the network border cannot be trusted, or the values do not match your internal network's CoS strategy.

A thorough explanation of the CoS rewriting and its underlying algorithms is beyond the scope of this document. For more information about traffic policing, and CoS in general, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at www.juniper.net/books .

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos OS. The example shown here was tested and verified on MX Series routers running a supported Junos OS release.

Overview

IN THIS SECTION

The purpose of this example is to demonstrate CoS rewriting at a network border to convey the traffics's CoS profile to the next-hop router, based on the forwarding class and packet loss priority (PLP) assigned to the traffic. CoS information rewriting is performed as the last step before a packet is transmitted onto the egress network.

In this example the rewriting is done when sending traffic from the host connected to Device R1 to the host connected to Device R2. The information required for rewriting the CoS parameters in the other direction is not included in this example. However, you can use the rewriting information in Device R1 (making changes for the interfaces used) and apply it to Device R2 to achieve bidirectional CoS rewriting.

Junos OS contains several default rewrite rules that might meet your requirements. You display them with the `show class-of-service rewrite-rule` command. Table 56 on page 590 shows a partial list of the default rewrite rule mapping.

Table 56: Partial list of the default rewrite rule mapping

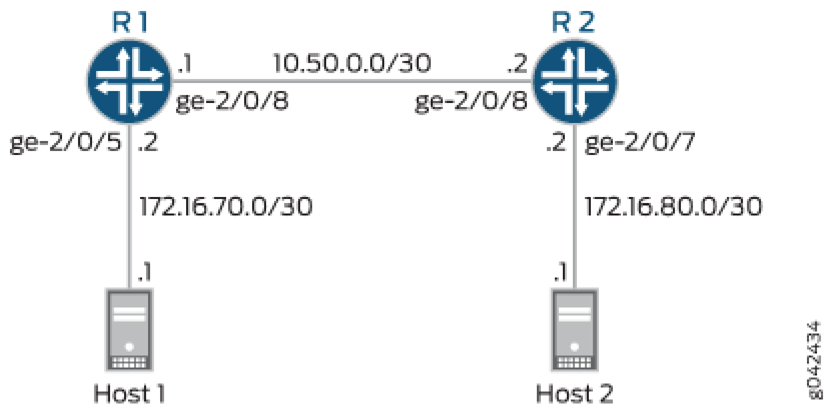
Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
expedited-forwarding	low	ef
expedited-forwarding	high	ef
assured-forwarding	low	af11
assured-forwarding	high	af12(DSCP/DSCP IPv6/EXP)
best-effort	low	be
best-effort	high	be
network-control	low	nc1/cs6
network-control	high	nc2/cs7

You can also define your own custom rewrite-rules table, or use a mixture of the default rewrite-rules and a custom table that you create. This example uses default rewrite-rules.

Topology

This example uses the topology in Figure 48 on page 591.

Figure 48: Rewriting CoS Information at the Network Border to Enforce CoS Strategies Scenario



This video explains the topics used in this example. We recommend that you watch the video before proceeding.

Video: Learning Bytes CoS Remarking Video.

Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description looback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set class-of-service forwarding-classes queue 0 BE-data
set class-of-service forwarding-classes queue 1 Premium-data
set class-of-service forwarding-classes queue 2 voice
set class-of-service forwarding-classes queue 3 NC
set class-of-service interfaces ge-2/0/8 scheduler-map test-map
set class-of-service interfaces ge-2/0/8 unit 0 rewrite-rules dscp IPv4-rewrite-table
set class-of-service rewrite-rules dscp IPv4-rewrite-table forwarding-class BE-data loss-
priority low code-point be
set class-of-service rewrite-rules dscp IPv4-rewrite-table forwarding-class Premium-data loss-
priority low code-point ef
set class-of-service scheduler-maps test-map forwarding-class BE-data scheduler BE-data
set class-of-service scheduler-maps test-map forwarding-class Premium-data scheduler Prem-data
set class-of-service schedulers BE-data transmit-rate 1m
set class-of-service schedulers BE-data buffer-size percent 25
set class-of-service schedulers BE-data priority low
set class-of-service schedulers Prem-data transmit-rate 1m
set class-of-service schedulers Prem-data buffer-size percent 25
set class-of-service schedulers Prem-data priority high
set firewall family inet filter mf-classifier term BE-data from protocol tcp
set firewall family inet filter mf-classifier term BE-data from port 80
set firewall family inet filter mf-classifier term BE-data then count BE-data
set firewall family inet filter mf-classifier term BE-data then forwarding-class BE-data
set firewall family inet filter mf-classifier term Prem-data from protocol tcp
set firewall family inet filter mf-classifier term Prem-data from port 12345
set firewall family inet filter mf-classifier term Prem-data then count Prem-data
set firewall family inet filter mf-classifier term Prem-data then forwarding-class Premium-data
set firewall family inet filter mf-classifier term accept then accept
set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Device R2

```
set interfaces ge-2/0/7 description to-Host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.1/30
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces ge-2/0/8 unit 0 family inet filter input mf-classifier
set interfaces unit 0 description looback-interface
set interfaces unit 0 family inet address 192.168.14.1/32
set firewall family inet filter mf-classifier term BE-data from dscp be
set firewall family inet filter mf-classifier term BE-data then count BE-data
set firewall family inet filter mf-classifier term Premium-data from dscp ef
set firewall family inet filter mf-classifier term Premium-data then count Premium-data
set firewall family inet filter mf-classifier term accept then accept
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure Device R1:

1. Configure the device interfaces.

```
[edit ]
user@R1# set interfaces ge-2/0/5 description to-Host
user@R1# set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
user@R1# set interfaces ge-2/0/8 description to-R2
user@R1# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set interfaces lo0 unit 0 description looback-interface
user@R1# set interfaces lo0 unit 0 family inet address 192.168.13.1/32
```

2. Configure the firewall parameters.

```
[edit ]
user@R1# set firewall family inet filter mf-classifier term BE-data from protocol tcp
user@R1# set firewall family inet filter mf-classifier term BE-data from port 80
user@R1# set firewall family inet filter mf-classifier term BE-data then count BE-data
user@R1# set firewall family inet filter mf-classifier term BE-data then forwarding-class BE-
data
user@R1# set firewall family inet filter mf-classifier term Prem-data from protocol tcp
user@R1# set firewall family inet filter mf-classifier term Prem-data from port 12345
user@R1# set firewall family inet filter mf-classifier term Prem-data then count Prem-data
```

```
user@R1# set firewall family inet filter mf-classifier term Prem-data then forwarding-class
Premium-data
user@R1# set firewall family inet filter mf-classifier term accept then accept
```

3. Configure the class-of-service parameters.

```
[edit ]
user@R1# set class-of-service forwarding-classes queue 0 BE-data
user@R1# set class-of-service forwarding-classes queue 1 Premium-data
user@R1# set class-of-service forwarding-classes queue 2 voice
user@R1# set class-of-service forwarding-classes queue 3 NC
user@R1# set class-of-service interfaces ge-2/0/8 scheduler-map test-map
user@R1# set class-of-service interfaces ge-2/0/8 unit 0 rewrite-rules dscp IPv4-rewrite-table
user@R1# set class-of-service rewrite-rules dscp IPv4-rewrite-table forwarding-class BE-data
loss-priority low code-point be
user@R1# set class-of-service rewrite-rules dscp IPv4-rewrite-table forwarding-class Premium-
data loss-priority low code-point ef
user@R1# set class-of-service scheduler-maps test-map forwarding-class BE-data scheduler BE-
data
user@R1# set class-of-service scheduler-maps test-map forwarding-class Premium-data scheduler
Prem-data
user@R1# set class-of-service schedulers BE-data transmit-rate 1m
user@R1# set class-of-service schedulers BE-data buffer-size percent 25
user@R1# set class-of-service schedulers BE-data priority low
user@R1# set class-of-service schedulers Prem-data transmit-rate 1m
user@R1# set class-of-service schedulers Prem-data buffer-size percent 25
user@R1# set class-of-service schedulers Prem-data priority high
```

4. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

Step-by-Step Procedure

To configure Device R2:

1. Configure the device interface.

```
[edit ]
user@R1# set interfaces ge-2/0/7 description to-Host
user@R1# set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.1/30
user@R1# set interfaces ge-2/0/8 description to-R1
user@R1# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@R2# set interfaces ge-2/0/8 unit 0 family inet filter input mf-classifier
user@R1# set interfaces unit 0 description looback-interface
user@R1# set interfaces unit 0 family inet address 192.168.14.1/32
```

2. Configure the firewall parameters.

```
[edit ]
user@R2# set firewall family inet filter mf-classifier term BE-data from dscp be
user@R2# set firewall family inet filter mf-classifier term BE-data then count BE-data
user@R2# set firewall family inet filter mf-classifier term Premium-data from dscp ef
user@R2# set firewall family inet filter mf-classifier term Premium-data then count Premium-
data
user@R2# set firewall family inet filter mf-classifier term accept then accept
```

3. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/7.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show firewall`, `show class-of-service` , and `show protocols ospf` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@R1 show interfaces
    ge-2/0/5 {
    description to-Host;
    unit 0 {
        family inet {
            filter {
                input mf-classifier;
            }
            address 172.16.70.2/30;
        }
    }
}
ge-2/0/8 {
    description to-R2;
    unit 0 {
        family inet {
            address 10.50.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
```

```
user@R1 show firewall
family inet {
    filter mf-classifier {
        term BE-data {
            from {
                protocol tcp;
                port 80;
            }
            then {
                count BE-data;
                forwarding-class BE-data;
            }
        }
        term Prem-data {
            from {
                protocol tcp;
                port 12345;
            }
            then {
                count Prem-data;
                forwarding-class Premium-data;
            }
        }
        term accept {
            then accept;
        }
    }
}
```

```
user@R1 **show class-of-service**
forwarding-classes {
    queue 0 BE-data;
    queue 1 Premium-data;
    queue 2 voice;
    queue 3 NC;
}
interfaces {
    ge-2/0/8 {
        scheduler-map test-map;
        unit 0 {
            rewrite-rules {
                dscp IPv4-rewrite-table;
            }
        }
    }
}
rewrite-rules {
    dscp IPv4-rewrite-table {
        forwarding-class BE-data {
            loss-priority low code-point be;
        }
        forwarding-class Premium-data {
            loss-priority low code-point ef;
        }
    }
}
scheduler-maps {
    test-map {
        forwarding-class BE-data scheduler BE-data;
        forwarding-class Premium-data scheduler Prem-data;
    }
}
schedulers {
    BE-data {
        transmit-rate 1m;
        buffer-size percent 25;
        priority low;
    }
    Prem-data {
        transmit-rate 1m;
        buffer-size percent 25;
        priority high;
    }
}
```

```
user@R1# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/5.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R1, enter `commit` from configuration mode.

```
user@R2# show interfaces
ge-2/0/7 {
    unit 0 {
        description to-Host;
        family inet {
            address 172.16.80.2;
        }
    }
}
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            filter {
                input mf-classifier;
            }
            address 10.50.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.14.1/32;
        }
    }
}
```

```
user@R2# show firewall
family inet {
    filter mf-classifier {
        term BE-data {
            from {
                dscp be;
            }
            then count BE-data;
        }
        term Premium-data {
            from {
                dscp ef;
            }
            then count Premium-data;
        }
        term accept {
            then accept;
        }
    }
}
```

```
user@R2# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/7.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}
```

If you are done configuring Device R2, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly.

Clearing the Firewall Counters

Purpose

Confirm that the firewall counters are cleared.

Action

On Devices R1 and R2, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@R1> clear firewall all
user@R2> clear firewall all
```

Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Monitoring the Results

Purpose

Send traffic from the host connected to Device 1 into the network so that it can be monitored by the firewall on Device R1 and Device R2.

Action

1. Use a traffic generator to send 20 TCP packets with a source port of 80 into the network.

   The -s flag sets the source port. The -k flag causes the source port to remain steady at 80 instead of incrementing. The -c flag sets the number of packets to 20. The -d flag sets the packet size.

   ```
   [User@host]#  hping 172.16.80.1  -c 20 -s 80  -k -d 300
   HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 0 data bytes
   len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.9 ms
   .
   .
   .
   --- 172.16.80.1 hping statistic ---
   20 packets transmitted, 20 packets received, 0% packet loss
   round-trip min/avg/max = 0.9/9501.4/19002.4 ms
   ```

2. Use a traffic generator to send 20 TCP packets with a source port of 12345 into the network.

   ```
   [User@host]#  hping 172.16.80.1  -c 20 -s 12345  -k -d 300
   HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 0 data bytes
   len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.3 ms
   .
   .
   .
   --- 172.16.80.1 hping statistic ---
   ```

```
20 packets transmitted, 20 packets received, 0% packet loss
round-trip min/avg/max = 0.3/9501.5/19002.7 ms
```

3. On Device R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall
Filter: mf-classifier
Counters:
Name                                                Bytes              Packets
BE-data                                               800                20
Prem-data                                             800                20
```

4. On Device R2, check the firewall counters using the `show firewall` command.

```
user@R2> show firewall
Filter: mf-classifier
Counters:
Name                                                Bytes              Packets
BE-data                                               800                20
Premium-data                                          800                20
```

Meaning

Device R1 correctly set the code point for TCP packets to port 12345 to bf. Device R1 correctly set the code point for TCP packets to port 80 to ef. Device R2 correctly recognized the code point for TCP packets to port 12345 as bf. Device R2 correctly recognized the code point for TCP packets to port 80 as ef.

RELATED DOCUMENTATION

Example: Configuring and Applying Scheduler Maps | **361**

Example: Remarking Diffserv Code Points to MPLS EXPs to Carry CoS Profiles Across a Service Provider's L3VPN MPLS Network

IN THIS SECTION

This example is an introduction in how to rewrite (remark) DSCP class-of-service (CoS) code point values at the network border of a customer network and a service provider's MPLS network while maintaining the original CoS profile of the traffic so that the traffic can be remarked with the original DSCP code points when it exits the MPLS network.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

The functionality in this procedure is widely supported on devices that run Junos OS. The example shown here was tested and verified on MX Series routers running a supported Junos OS release.

Overview

IN THIS SECTION

The purpose of rewriting the IP DSCP code point values to MPLS EXP code point values is to carry the packet's CoS profile across the service provider's MPLS network. The rewriting is performed by the provider edge (PE) routers at the borders of the service provider's network. See Figure 50 on page 606.

Junos OS contains several DSCP default rewrite rules that might meet your requirements. You display them with the `show class-of-service rewrite-rule` command. A partial set of the default rewrite DSCP code point rule mappings is shown in the following table.

You can also define your own custom rewrite-rules table, or use a mixture of the default rewrite-rules and a custom table that you create. This example uses default rewrite-rules.

Table 57: Default rewrite rules

Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
expedited-forwarding	low	ef
expedited-forwarding	high	ef
assured-forwarding	low	af11
assured-forwarding	high	af12 (DSCP/DSCP IPv6/EXP)
best-effort	low	be
best-effort	high	be
network-control	low	nc1/cs6
network-control	high	nc2/cs7

Junos OS uses the values shown in the following table for MPLS CoS in the EXP fields of the MPLS header.

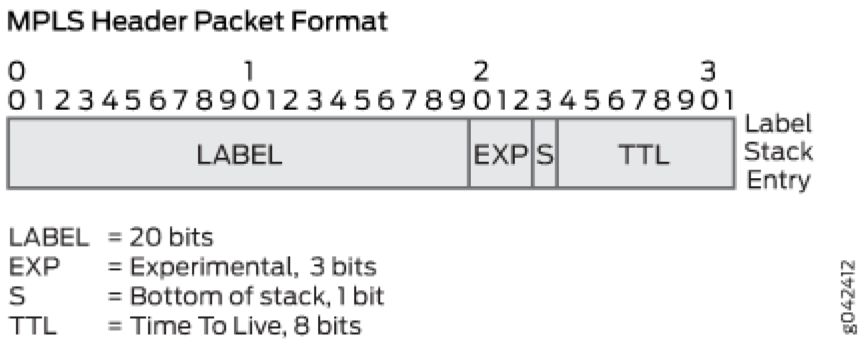
Forwarding Class	Loss Priority	EXP Code Point
best-effort	low	000
best-effort	high	001
expedited-forwarding	low	010
expedited-forwarding	high	011

(Continued)

Forwarding Class	Loss Priority	EXP Code Point
assured-forwarding	low	100
assured-forwarding	high	101
network-control	low	110
network-control	high	111

Figure 49 on page 605 shows the MPLS packet structure.

Figure 49: MPLS Packet Structure



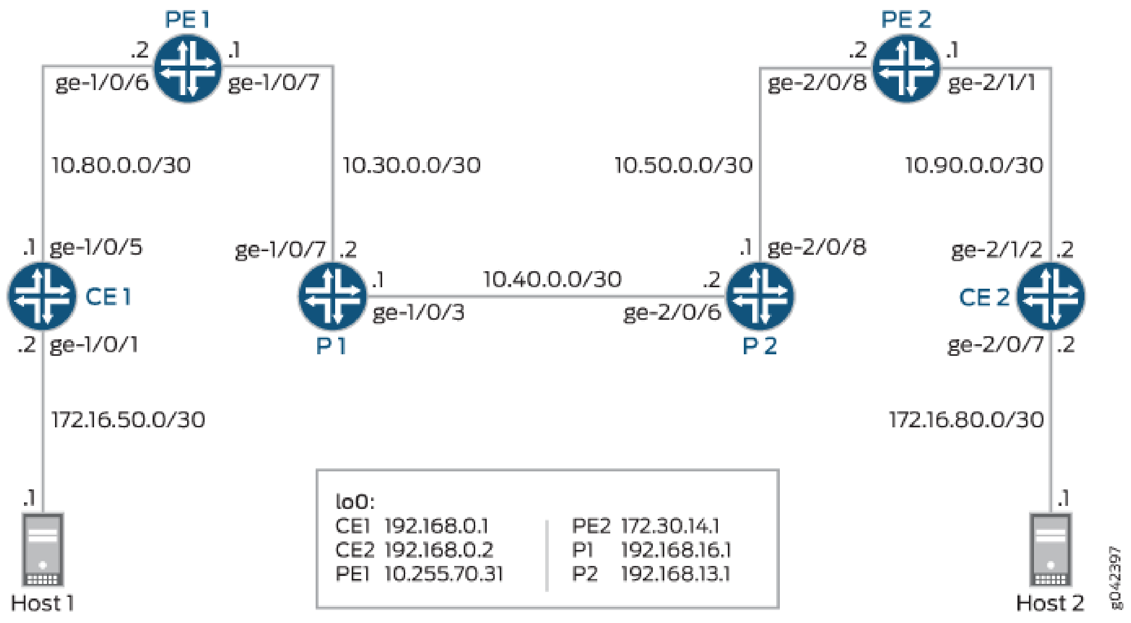
NOTE: In addition to providing the necessary information to complete the purpose of this example, this example also includes all of the commands required to re-create the Layer 3 VPN (L3VPN) network as shown in Figure 50 on page 606. A full explanation of the tasks required to configure an L3VPN network is not included in this example. If you require more information regarding configuring an L3VPN network, refer to the *Layer 3 VPNs User Guide for Routing Devices* available at http://juniper.net/documentation .

A thorough explanation of the required CoS rewriting and the underlying algorithms used in this example is beyond the scope of this document. For more information, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at www.juniper.net/books .

Topology

This example uses the topology in Figure 50 on page 606.

Figure 50: Rewriting CoS Information at the Network Border to Transit an MPLS Network Scenario



Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device CE1

```
set interfaces ge-1/0/1 unit 0 description to-host
set interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
set interfaces ge-1/0/1 unit 0 family inet filter input ip-v4
set interfaces ge-1/0/5 unit 0 description to_Provider
set interfaces ge-1/0/5 unit 0 family inet address 10.80.0.1/30
set interfaces lo0 unit 1 description loopback-interface
set interfaces lo0 unit 1 family inet address 192.168.0.1/32
set protocols bgp group to_Provider type external
set protocols bgp group to_Provider export send-direct
set protocols bgp group to_Provider peer-as 64511
set protocols bgp group to_Provider neighbor 10.80.0.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
set firewall family inet filter ip-v4 term tcp80 from port 80
set firewall family inet filter ip-v4 term tcp80 then dscp ef
set firewall family inet filter ip-v4 term 12345 from port 12345
set firewall family inet filter ip-v4 term 12345 then dscp be
set firewall family inet filter ip-v4 term accept then accept
```

Device PE1

```
set interfaces ge-1/0/6 description to_vpna
set interfaces ge-1/0/6 unit 0 family inet address 10.80.0.2/30
set interfaces ge-1/0/7 description to_P1
set interfaces ge-1/0/7 unit 0 family inet address 10.30.0.1/30
set interfaces ge-1/0/7 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 10.255.70.31/32
set routing-options router-id 10.255.70.31
set routing-options autonomous-system 64511
set protocols mpls interface ge-1/0/7.0
set protocols bgp group to_PE2 type internal
set protocols bgp group to_PE2 local-address 10.255.70.31
set protocols bgp group to_PE2 family inet-vpn unicast
set protocols bgp group to_PE2 neighbor 172.30.14.1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/7.0
set protocols ldp interface ge-1/0/7.0
set protocols ldp interface lo0.0
set routing-instances vpna instance-type vrf
set routing-instances vpna interface ge-1/0/6.0
set routing-instances vpna route-distinguisher 64511:1
set routing-instances vpna vrf-target target:64511:1
set routing-instances vpna protocols bgp group to_vpna type external
set routing-instances vpna protocols bgp group to_vpna peer-as 64510
set routing-instances vpna protocols bgp group to_vpna neighbor 10.80.0.1
set class-of-service classifiers dscp dscpv4 forwarding-class expedited-forwarding loss-priority
low code-points ef
set class-of-service classifiers dscp dscpv4 forwarding-class best-effort loss-priority low code-
points be
set class-of-service classifiers exp exp-in forwarding-class expedited-forwarding loss-priority
low code-points 010
set class-of-service classifiers exp exp-in forwarding-class best-effort loss-priority low code-
points 000
set class-of-service interfaces ge-1/0/6 unit 0 classifiers dscp dscpv4
set class-of-service interfaces ge-1/0/6 unit 0 rewrite-rules dscp dscpv4-rw
set class-of-service interfaces ge-1/0/7 unit 0 classifiers exp exp-in
set class-of-service interfaces ge-1/0/7 unit 0 rewrite-rules exp exp-out
set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class expedited-forwarding loss-
priority low code-point ef
set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class best-effort loss-priority low
code-point be
set class-of-service rewrite-rules exp exp-out forwarding-class expedited-forwarding loss-
priority low code-point 010
set class-of-service rewrite-rules exp exp-out forwarding-class best-effort loss-priority low
code-point 000
```

Device P1

```
set interfaces ge-1/0/3 description to_P2
set interfaces ge-1/0/3 unit 0 family inet address 10.40.0.1/30
set interfaces ge-1/0/3 unit 0 family mpls
set interfaces ge-1/0/7 description to_PE1
set interfaces ge-1/0/7 unit 0 family inet address 10.30.0.2/30
set interfaces ge-1/0/7 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.16.1/32
set routing-options router-id 10.255.187.32
set protocols mpls interface ge-1/0/7.0
set protocols mpls interface ge-1/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/0/7.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/0/3.0
set protocols ldp interface ge-1/0/7.0
set protocols ldp interface lo0.0
```

Device P2

```
set interfaces ge-2/0/6 description to_P1
set interfaces ge-2/0/6 unit 0 family inet address 10.40.0.2/30
set interfaces ge-2/0/6 unit 0 family mpls
set interfaces ge-2/0/8 description to_PE2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set routing-options router-id 192.168.187.3
set protocols mpls interface ge-2/0/6.0
set protocols mpls interface ge-2/0/8.0
set protocols ospf area 0.0.0.0 interface ge-2/0/6.0
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-2/0/6.0
set protocols ldp interface ge-2/0/8.0
set protocols ldp interface lo0.0
```

Device PE2

```
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces ge-2/1/1 unit 0 description to-vpna
set interfaces ge-2/1/1 unit 0 family inet address 10.90.0.1/30
set interfaces ge-2/1/7 unit 0 family inet address 10.0.31.2/30
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 172.30.14.1
set routing-options router-id 172.30.14.1
set routing-options autonomous-system 64511
set protocols mpls interface ge-2/0/8.0
set protocols bgp group to_PE2 type internal
set protocols bgp group to_PE2 local-address 172.30.14.1
set protocols bgp group to_PE2 family inet-vpn unicast
set protocols bgp group to_PE2 neighbor 10.255.70.31
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-2/0/8.0
set protocols ldp interface lo0.0
set routing-instances vpna instance-type vrf
set routing-instances vpna interface ge-2/1/1.0
set routing-instances vpna route-distinguisher 64511:1
set routing-instances vpna vrf-target target:64511:1
set routing-instances vpna protocols bgp group to_vpna type external
set routing-instances vpna protocols bgp group to_vpna peer-as 64512
set routing-instances vpna protocols bgp group to_vpna neighbor 10.90.0.2
set class-of-service classifiers dscp dscpv4 forwarding-class expedited-forwarding loss-priority
low code-points ef
set class-of-service classifiers dscp dscpv4 forwarding-class best-effort loss-priority low code-
points be
set class-of-service classifiers exp exp-in forwarding-class expedited-forwarding loss-priority
low code-points 010
set class-of-service classifiers exp exp-in forwarding-class best-effort loss-priority low code-
points 000
set class-of-service interfaces ge-2/0/8 unit 0 classifiers exp exp-in
set class-of-service interfaces ge-2/0/8 unit 0 rewrite-rules exp exp-out
set class-of-service interfaces ge-2/1/1 unit 0 classifiers dscp dscpv4
set class-of-service interfaces ge-2/1/1 unit 0 rewrite-rules dscp dscpv4-rw
set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class expedited-forwarding loss-
priority low code-point ef
set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class best-effort loss-priority low
code-point be
set class-of-service rewrite-rules exp exp-out forwarding-class expedited-forwarding loss-
priority low code-point 010
set class-of-service rewrite-rules exp exp-out forwarding-class best-effort loss-priority low
code-point 000
```

Device CE2

```
set interfaces ge-2/0/7 unit 0 description to-host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces ge-2/0/7 unit 0 family inet filter input ip-v4
set interfaces ge-2/1/2 unit 0 description to-Provider
set interfaces ge-2/1/2 unit 0 family inet address 10.90.0.2/30
set interfaces lo0 unit 1 description loopback-interface
set interfaces lo0 unit 1 family inet address 192.168.0.2/32
set protocols bgp group to_Provider type external
set protocols bgp group to_Provider export send-direct
set protocols bgp group to_Provider peer-as 64511
set protocols bgp group to_Provider neighbor 10.90.0.1
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options router-id 192.168.0.2
set routing-options autonomous-system 64512
set firewall family inet filter ip-v4 term tcp80 from port 80
set firewall family inet filter ip-v4 term tcp80 then dscp ef
set firewall family inet filter ip-v4 term 12345 from port 12345
set firewall family inet filter ip-v4 term 12345 then dscp be
set firewall family inet filter ip-v4 term accept then accept
```

Step-by-Step Procedure

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

To configure Device CE1:

1. Configure the device interfaces.

```
[edit ]
user@CE1# set interfaces ge-1/0/1 unit 0 description to-host
user@CE1# set interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
user@CE1# set interfaces ge-1/0/1 unit 0 family inet filter input ip-v4
user@CE1# set interfaces ge-1/0/5 unit 0 description to_Provider
user@CE1# set interfaces ge-1/0/5 unit 0 family inet address 10.80.0.1/30
user@CE1# set interfaces lo0 unit 1 description loopback-interface
user@CE1# set interfaces lo0 unit 1 family inet address 192.168.0.1/32
```

2. Configure the BGP parameters

```
[edit ]
user@CE1# set protocols bgp group to_Provider type external
```

```
user@CE1# set protocols bgp group to_Provider export send-direct
user@CE1# set protocols bgp group to_Provider peer-as 64511
user@CE1# set protocols bgp group to_Provider neighbor 10.80.0.2
```

3. Configure the policy option parameters.

```
[edit ]
user@CE1# set policy-options policy-statement send-direct from protocol direct
user@CE1# set policy-options policy-statement send-direct then accept
```

4. Configure the routing option parameters.

```
[edit ]
user@CE1# set routing-options router-id 192.168.0.1
user@CE1# set routing-options autonomous-system 64510
```

5. Configure the DSCP code point rewrite parameters.

```
[edit ]
user@CE1# set firewall family inet filter ip-v4 term tcp80 from port 80
user@CE1# set firewall family inet filter ip-v4 term tcp80 then dscp ef
user@CE1# set firewall family inet filter ip-v4 term 12345 from port 12345
user@CE1# set firewall family inet filter ip-v4 term 12345 then dscp be
user@CE1# set firewall family inet filter ip-v4 term accept then accept
```

Step-by-Step Procedure

To configure Device PE1:

1. Configure the device interfaces.

```
[edit ]
user@PE1# set interfaces ge-1/0/6 description to_vpna
user@PE1# set interfaces ge-1/0/6 unit 0 family inet address 10.80.0.2/30
user@PE1# set interfaces ge-1/0/7 description to_P1
user@PE1# set interfaces ge-1/0/7 unit 0 family inet address 10.30.0.1/30
user@PE1# set interfaces ge-1/0/7 unit 0 family mpls
```

```
user@PE1# set interfaces lo0 unit 0 description loopback-interface
user@PE1# set interfaces lo0 unit 0 family inet address 10.255.70.31/32
```

2. Configure the routing option parameters.

```
[edit ]
user@PE1# set routing-options router-id 10.255.70.31
user@PE1# set routing-options autonomous-system 64511
```

3. Configure the protocol parameters.

```
user@PE1# set protocols mpls interface ge-1/0/7.0
user@PE1# set protocols bgp group to_PE2 type internal
user@PE1# set protocols bgp group to_PE2 local-address 10.255.70.31
user@PE1# set protocols bgp group to_PE2 family inet-vpn unicast
user@PE1# set protocols bgp group to_PE2 neighbor 172.30.14.1
user@PE1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@PE1# set protocols ospf area 0.0.0.0 interface ge-1/0/7.0
user@PE1# set protocols ldp interface ge-1/0/7.0
user@PE1# set protocols ldp interface lo0.0
```

4. Configure the routing instance parameters.

```
[edit ]
user@PE1# set routing-instances vpna instance-type vrf
user@PE1# set routing-instances vpna interface ge-1/0/6.0
user@PE1# set routing-instances vpna route-distinguisher 64511:1
user@PE1# set routing-instances vpna vrf-target target:64511:1
user@PE1# set routing-instances vpna protocols bgp group to_vpna type external
user@PE1# set routing-instances vpna protocols bgp group to_vpna peer-as 64510
user@PE1# set routing-instances vpna protocols bgp group to_vpna neighbor 10.80.0.1
```

5. Configure the class-of-service parameters that perform the DSCP code point to MPLS EXP rewriting.

```
user@PE1# set class-of-service classifiers dscp dscpv4 forwarding-class expedited-forwarding
loss-priority low code-points ef
user@PE1# set class-of-service classifiers dscp dscpv4 forwarding-class best-effort loss-
priority low code-points be
user@PE1# set class-of-service classifiers exp exp-in forwarding-class expedited-forwarding
```

```
loss-priority low code-points 010
user@PE1# set class-of-service classifiers exp exp-in forwarding-class best-effort loss-
priority low code-points 000
user@PE1# set class-of-service interfaces ge-1/0/6 unit 0 classifiers dscp dscpv4
user@PE1# set class-of-service interfaces ge-1/0/6 unit 0 rewrite-rules dscp dscpv4-rw
user@PE1# set class-of-service interfaces ge-1/0/7 unit 0 classifiers exp exp-in
user@PE1# set class-of-service interfaces ge-1/0/7 unit 0 rewrite-rules exp exp-out
user@PE1# set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class expedited-
forwarding loss-priority low code-point ef
user@PE1# set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class best-effort loss-
priority low code-point be
user@PE1# set class-of-service rewrite-rules exp exp-out forwarding-class expedited-
forwarding loss-priority low code-point 010
user@PE1# set class-of-service rewrite-rules exp exp-out forwarding-class best-effort loss-
priority low code-point 000
```

Step-by-Step Procedure

To configure Device P1:

1. Configure the device interfaces.

```
[edit ]
user@P1# set interfaces ge-1/0/3 description to_P2
user@P1# set interfaces ge-1/0/3 unit 0 family inet address 10.40.0.1/30
user@P1# set interfaces ge-1/0/3 unit 0 family mpls
user@P1# set interfaces ge-1/0/7 description to_PE1
user@P1# set interfaces ge-1/0/7 unit 0 family inet address 10.30.0.2/30
user@P1# set interfaces ge-1/0/7 unit 0 family mpls
user@P1# set interfaces lo0 unit 0 description loopback-interface
user@P1# set interfaces lo0 unit 0 family inet address 192.168.16.1/32
```

2. Configure the routing option parameters.

```
[edit ]
user@P1# set routing-options router-id 10.255.187.32
```

3. Configure the protocol parameters.

```
[edit ]
user@P1# set protocols mpls interface ge-1/0/7.0
user@P1# set protocols mpls interface ge-1/0/3.0
user@P1# set protocols ospf area 0.0.0.0 interface ge-1/0/3.0
user@P1# set protocols ospf area 0.0.0.0 interface ge-1/0/7.0
user@P1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@P1# set protocols ldp interface ge-1/0/3.0
user@P1# set protocols ldp interface ge-1/0/7.0
user@P1# set protocols ldp interface lo0.0
```

Step-by-Step Procedure

To configure Device P2:

1. Configure the device interfaces.

```
[edit ]
user@P2# set interfaces ge-2/0/6 description to_P1
user@P2# set interfaces ge-2/0/6 unit 0 family inet address 10.40.0.2/30
user@P2# set interfaces ge-2/0/6 unit 0 family mpls
user@P2# set interfaces ge-2/0/8 description to_PE2
user@P2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@P2# set interfaces ge-2/0/8 unit 0 family mpls
user@P2# set interfaces lo0 unit 0 description loopback-interface
user@P2# set interfaces lo0 unit 0 family inet address 192.168.13.1/32
```

2. Configure the routing option parameters.

```
[edit ]
user@P2# set routing-options router-id 192.168.187.3
```

3. Configure the protocol parameters.

```
[edit ]
user@P2# set protocols mpls interface ge-2/0/6.0
user@P2# set protocols mpls interface ge-2/0/8.0
user@P2# set protocols ospf area 0.0.0.0 interface ge-2/0/6.0
```

```
user@P2# set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
user@P2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@P2# set protocols ldp interface ge-2/0/6.0
user@P2# set protocols ldp interface ge-2/0/8.0
user@P2# set protocols ldp interface lo0.0
```

Step-by-Step Procedure

To configure Device PE2:

1. Configure the device interfaces.

```
[edit ]
user@PE2# set interfaces ge-2/0/8 description to-R1
user@PE2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@PE2# set interfaces ge-2/0/8 unit 0 family mpls
user@PE2# set interfaces ge-2/1/1 unit 0 description to-vpna
user@PE2# set interfaces ge-2/1/1 unit 0 family inet address 10.90.0.1/30
user@PE2# set interfaces lo0 unit 0 description loopback-interface
user@PE2# set interfaces lo0 unit 0 family inet address 172.30.14.1/32
```

2. Configure the routing option parameters.

```
[edit ]
user@PE2# set routing-options router-id 172.30.14.1
user@PE2# set routing-options autonomous-system 64511
```

3. Configure the protocol parameters.

```
[edit ]
user@PE2# set protocols mpls interface ge-2/0/8.0
user@PE2# set protocols bgp group to_PE2 type internal
user@PE2# set protocols bgp group to_PE2 local-address 172.30.14.1
user@PE2# set protocols bgp group to_PE2 family inet-vpn unicast
user@PE2# set protocols bgp group to_PE2 neighbor 10.255.70.31
user@PE2# set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
user@PE2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@PE2# set protocols ldp interface ge-2/0/8.0
user@PE2# set protocols ldp interface lo0.0
```

4. Configure the routing instance parameters.

```
[edit ]
user@PE2# set routing-instances vpna instance-type vrf
user@PE2# set routing-instances vpna interface ge-2/1/1.0
user@PE2# set routing-instances vpna route-distinguisher 64511:1
user@PE2# set routing-instances vpna vrf-target target:64511:1
user@PE2# set routing-instances vpna protocols bgp group to_vpna type external
user@PE2# set routing-instances vpna protocols bgp group to_vpna peer-as 64512
user@PE2# set routing-instances vpna protocols bgp group to_vpna neighbor 10.90.0.2
```

5. Configure the class-of-service parameters that perform the DSCP code point to MPLS EXP rewriting.

```
[edit ]
user@PE2# set class-of-service classifiers dscp dscpv4 forwarding-class expedited-forwarding
loss-priority low code-points ef
user@PE2# set class-of-service classifiers dscp dscpv4 forwarding-class best-effort loss-
priority low code-points be
user@PE2# set class-of-service classifiers exp exp-in forwarding-class expedited-forwarding
loss-priority low code-points 010
user@PE2# set class-of-service classifiers exp exp-in forwarding-class best-effort loss-
priority low code-points 000
user@PE2# set class-of-service interfaces ge-2/0/8 unit 0 classifiers exp exp-in
user@PE2# set class-of-service interfaces ge-2/0/8 unit 0 rewrite-rules exp exp-out
user@PE2# set class-of-service interfaces ge-2/1/1 unit 0 classifiers dscp dscpv4
user@PE2# set class-of-service interfaces ge-2/1/1 unit 0 rewrite-rules dscp dscpv4-rw
user@PE2# set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class expedited-
forwarding loss-priority low code-point ef
user@PE2# set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class best-effort loss-
priority low code-point be
user@PE2# set class-of-service rewrite-rules exp exp-out forwarding-class expedited-
forwarding loss-priority low code-point 010
user@PE2# set class-of-service rewrite-rules exp exp-out forwarding-class best-effort loss-
priority low code-point 000
```

Step-by-Step Procedure

To configure Device CE2:

1. Configure the device interfaces.

```
[edit ]
user@CE2# set interfaces ge-2/0/7 unit 0 description to-host
user@CE2# set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
user@CE2# set  interfaces ge-2/0/7 unit 0 family inet filter input ip-v4
user@CE2# set interfaces ge-2/1/2 unit 0 description to-Provider
user@CE2# set interfaces ge-2/1/2 unit 0 family inet address 10.90.0.2/30


                        set interfaces lo0 unit 1 description loopback-interface


                        set interfaces lo0 unit 1 family inet address 192.168.0.2/32
```

2. Configure the protocol parameters.

```
[edit ]
user@CE2# set protocols bgp group to_Provider type external
user@CE2# set protocols bgp group to_Provider export send-direct
user@CE2# set protocols bgp group to_Provider peer-as 64511
user@CE2# set protocols bgp group to_Provider neighbor 10.90.0.1
```

3. Configure the policy option parameters.

```
[edit ]
user@CE2# set policy-options policy-statement send-direct from protocol direct
user@CE2# set policy-options policy-statement send-direct then accept
```

4. Configure the routing option parameters.

```
[edit ]
user@CE2# set routing-options router-id 192.168.0.2
user@CE2# set routing-options autonomous-system 64512
```

5. Configure the DSCP code point rewrite parameters.

```
[edit ]
user@CE2# set  firewall family inet filter ip-v4 term tcp80 from port 80
user@CE2# set  firewall family inet filter ip-v4 term tcp80 then dscp ef
user@CE2# set  firewall family inet filter ip-v4 term 12345 from port 12345
user@CE2# set  firewall family inet filter ip-v4 term 12345 then dscp be
user@CE2# set  firewall family inet filter ip-v4 term accept then accept
```

Results

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

```
user@CE1# show interfaces
ge-1/0/1 {
    unit 0 {
        description to-host;
        family inet {
            filter {
                input ip-v4;
            }
            address 172.16.50.2/30;
        }
    }
}
ge-1/0/5 {
    unit 0 {
        description to_Provider;
        family inet {
            address 10.80.0.1/30;
        }
    }
}
lo0 {
    unit 1 {
        description loopback-interface;
        family inet {
            address 192.168.0.1/32;
        }
    }
}
```

```
user@CE1# show protocols
bgp {
    group to_Provider {
        type external;
        export send-direct;
        peer-as 64511;
        neighbor 10.80.0.2;
    }
}
```

```
user@CE1# show policy-options
policy-statement send-direct {
    from protocol direct;
    then accept;
}
```

```
user@CE1# show routing-options
router-id 192.168.0.1;
autonomous-system 64510;
```

```
user@CE1# show firewall
family inet {
    filter ip-v4 {
        term tcp80 {
            from {
                port 80;
            }
            then dscp ef;
        }
        term 12345 {
            from {
                port 12345;
            }
            then dscp be;
        }
        term accept {
            then accept;
        }
    }
}
```

If you are done configuring Device CE1, enter `commit` from configuration mode.

```
user@PE1# show interfaces
ge-1/0/6 {
    description to_vpna;
    unit 0 {
        family inet {
            address 10.80.0.2/30;
        }
    }
}
ge-1/0/7 {
    description to_P1;
    unit 0 {
        family inet {
            address 10.30.0.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 10.255.70.31/32;
        }
    }
}
```

```
user@PE1# show protocols
mpls {
    interface ge-1/0/7.0;
}
bgp {
    group to_PE2 {
        type internal;
        local-address 10.255.70.31;
        family inet-vpn {
            unicast;
        }
        neighbor 172.30.14.1;
    }
}
ospf {
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-1/0/7.0;
    }
}
ldp {
    interface ge-1/0/7.0;
    interface lo0.0;
}
```

```
user@PE1# show routing-options
router-id 10.255.70.31;
autonomous-system 64511;
```

```
user@PE1# show routing-instances
vpna {
    instance-type vrf;
    interface ge-1/0/6.0;
    route-distinguisher 64511:1;
    vrf-target target:64511:1;
    protocols {
        bgp {
            group to_vpna {
                type external;
                peer-as 64510;
                neighbor 10.80.0.1;
            }
        }
    }
}
```

```
user@PE1# show class-of-service
classifiers {
    dscp dscpv4 {
        forwarding-class expedited-forwarding {
            loss-priority low code-points ef;
        }
        forwarding-class best-effort {
            loss-priority low code-points be;
        }
    }
    exp exp-in {
        forwarding-class expedited-forwarding {
            loss-priority low code-points 010;
        }
        forwarding-class best-effort {
            loss-priority low code-points 000;
        }
    }
}
interfaces {
    ge-1/0/6 {
        unit 0 {
            classifiers {
                dscp dscpv4;
            }
            rewrite-rules {
                dscp dscpv4-rw;
            }
        }
    }
    ge-1/0/7 {
        unit 0 {
            classifiers {
                exp exp-in;
            }
            rewrite-rules {
                exp exp-out;
            }
        }
    }
}
rewrite-rules {
    dscp dscpv4-rw {
        forwarding-class expedited-forwarding {
            loss-priority low code-point ef;
        }
        forwarding-class best-effort {
            loss-priority low code-point be;
        }
    }
    exp exp-out {
        forwarding-class expedited-forwarding {
            loss-priority low code-point 010;
        }
        forwarding-class best-effort {
            loss-priority low code-point 000;
        }
    }
}
```

If you are done configuring Device PE1, enter `commit` from configuration mode.

```
user@P1# show interfaces
ge-1/0/3 {
    description to_P2;
    unit 0 {
        family inet {
            address 10.40.0.1/30;
        }
        family mpls;
    }
}
ge-1/0/7 {
    description to_PE1;
    unit 0 {
        family inet {
            address 10.30.0.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.16.1/32;
        }
    }
}
```

```
user@P1# show protocols
mpls {
    interface ge-1/0/7.0;
    interface ge-1/0/3.0;
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/3.0;
        interface ge-1/0/7.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/3.0;
    interface ge-1/0/7.0;
    interface lo0.0;
}
```

```
user@P1# show routing-options
router-id 10.255.187.32;
```

If you are done configuring Device P1, enter `commit` from configuration mode.

```
user@P2# show interfaces
ge-2/0/6 {
    description to_P1;
    unit 0 {
        family inet {
            address 10.40.0.2/30;
        }
        family mpls;
    }
}
ge-2/0/8 {
    description to_PE2;
    unit 0 {
        family inet {
            address 10.50.0.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
```

```
user@P2# show protocols
mpls {
    interface ge-2/0/6.0;
    interface ge-2/0/8.0;
}
ospf {
    area 0.0.0.0 {
        interface ge-2/0/6.0;
        interface ge-2/0/8.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-2/0/6.0;
    interface ge-2/0/8.0;
    interface lo0.0;
}
```

```
user@P2# show routing-options
router-id 192.168.187.3;
```

If you are done configuring Device P2, enter `commit` from configuration mode.

```
user@PE2# show interfaces
```

```
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.50.0.2/30;
        }
        family mpls;
    }
}
ge-2/1/1 {
    unit 0 {
        description to-vpna;
        family inet {
            address 10.90.0.1/30;
        }
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 172.30.14.1/32;
        }
    }
}
```

```
user@PE2# show protocols
mpls {
    interface ge-2/0/8.0;
}
bgp {
    group to_PE1 {
        type internal;
        local-address 172.30.14.1;
        family inet-vpn {
            unicast;
        }
        neighbor 10.255.70.31;
    }
}
ospf {
    area 0.0.0.0 {
        interface ge-2/0/8.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-2/0/8.0;
    interface lo0.0;
}
```

```
user@PE2# show routing-options
router-id 172.30.14.1;
autonomous-system 64511;
```

```
user@PE2# show routing-instances
vpna {
    instance-type vrf;
    interface ge-2/1/1.0;
    route-distinguisher 64511:1;
    vrf-target target:64511:1;
    protocols {
        bgp {
            group to_vpna {
                type external;
                peer-as 64512;
                neighbor 10.90.0.2;
            }
        }
    }
}
```

```
user@PE2# **show class-of-service**
classifiers {
    dscp dscpv4 {
        forwarding-class expedited-forwarding {
            loss-priority low code-points ef;
        }
        forwarding-class best-effort {
            loss-priority low code-points be;
        }
    }
    exp exp-in {
        forwarding-class expedited-forwarding {
            loss-priority low code-points 010;
        }
        forwarding-class best-effort {
            loss-priority low code-points 000;
        }
    }
}
interfaces {
    ge-2/0/8 {
        unit 0 {
            classifiers {
                exp exp-in;
            }
            rewrite-rules {
                exp exp-out;
            }
        }
    }
    ge-2/1/1 {
        unit 0 {
            classifiers {
                dscp dscpv4;
            }
            rewrite-rules {
                dscp dscpv4-rw;
            }
        }
    }
}
rewrite-rules {
    dscp dscpv4-rw {
        forwarding-class expedited-forwarding {
            loss-priority low code-point ef;
        }
        forwarding-class best-effort {
            loss-priority low code-point be;
        }
    }
    exp exp-out {
        forwarding-class expedited-forwarding {
            loss-priority low code-point 010;
        }
        forwarding-class best-effort {
            loss-priority low code-point 000;
        }
    }
}
```

If you are done configuring Device PE2, enter `commit` from configuration mode.

```
user@CE2# show interfaces
ge-2/0/7 {
    unit 0 {
        description to-host;
        family inet {
            filter {
                input ip-v4;
            }
            address 172.16.80.2/30;
        }
    }
}
ge-2/1/2 {
    unit 0 {
        description to-Provider;
        family inet {
            address 10.90.0.2/30;
        }
    }
}
lo0 {
    unit 1 {
        description loopback-interface;
        family inet {
            address 192.168.0.2/32;
        }
    }
}
```

```
user@CE2# show protocols
bgp {
    group to_Provider {
        type external;
        export send-direct;
        peer-as 64511;
        neighbor 10.90.0.1;
    }
}
```

```
user@CE2# show policy-options
policy-statement send-direct {
    from protocol direct;
    then accept;
}
```

```
user@CE2# show routing-options
router-id 192.168.0.2;
autonomous-system 64512;
```

```
user@CE2# show firewall
family inet {
    filter ip-v4 {
        term tcp80 {
            from {
                port 80;
            }
            then dscp ef;
        }
        term 12345 {
            from {
                port 12345;
            }
            then dscp be;
        }
        term accept {
            then accept;
        }
    }
}
```

If you are done configuring Device CE2, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly by verifying that the DSCP code points are maintained from CE1 to CE2.

Clearing the Firewall Counters

Purpose

Confirm that the firewall counters are cleared.

Action

On Device CE2, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@CE2> clear firewall all
```

Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Monitoring the Results

Purpose

Send traffic into the network from the host connected to Device CE1 so that it that can be monitored at Device CE2.

Action

A different firewall is required on interface ge-2/0/7 to count the traffic that is being transmitted outbound to the destination. The following commands apply the firewall filter that counts the marked traffic as it is transmitted to the destination.

NOTE: To capture traffic at Device CE1, apply this command `set interfaces ge-1/0/1 unit 0 family inet filter output count`, followed by the commands below.

NOTE: To capture traffic at Device CE2, apply this command `set interfaces ge-2/0/7 unit 0 family inet filter output count`, followed by the commands below.

```
set firewall family inet filter count term be from dscp be
set firewall family inet filter count term be then count be
set firewall family inet filter count term ef from dscp ef
set firewall family inet filter count term ef then count ef
set firewall family inet filter count term accept then accept
set interfaces ge-2/0/7 unit 0 family inet filter output count
```

When you are done testing, you can leave the counting filter in place, or remove it.

1. On host 1 use a traffic generator to send 20 TCP packets with a source port of 80 into the network, and repeat the task using a source port of 12345.

```
[user@host]#  hping 172.16.80.1  -s 80  -k -c 20
[user@host]#  hping 172.16.80.1  -s 12345  -k -c 20
```

2. On Device CE2, check the firewall counters by using the `show firewall` command.

```
user@CE2> show firewall

Filter: __CE2/ip-v4

Filter: __CE2/count
Counters:
Name                                                Bytes              Packets
be                                                    800                20
ef                                                    800                20
```

Meaning

The code point for TCP packets to port 12345 is maintained as be. The code point for TCP packets to port 80 is maintained as ef.

RELATED DOCUMENTATION

Example: Configuring and Applying Scheduler Maps | **361**

Example: Remarking Diffserv Code Points to 802.1P PCPs to Carry CoS Profiles Across a Service Provider's VPLS Network

IN THIS SECTION

This configuration example explains how to implement class-of-service (CoS) capabilities over a Virtual Private LAN Service (VPLS) network.

Requirements

To verify this procedure, this example uses a traffic generator. The traffic generator can be hardware-based or it can be software running on a server or host machine.

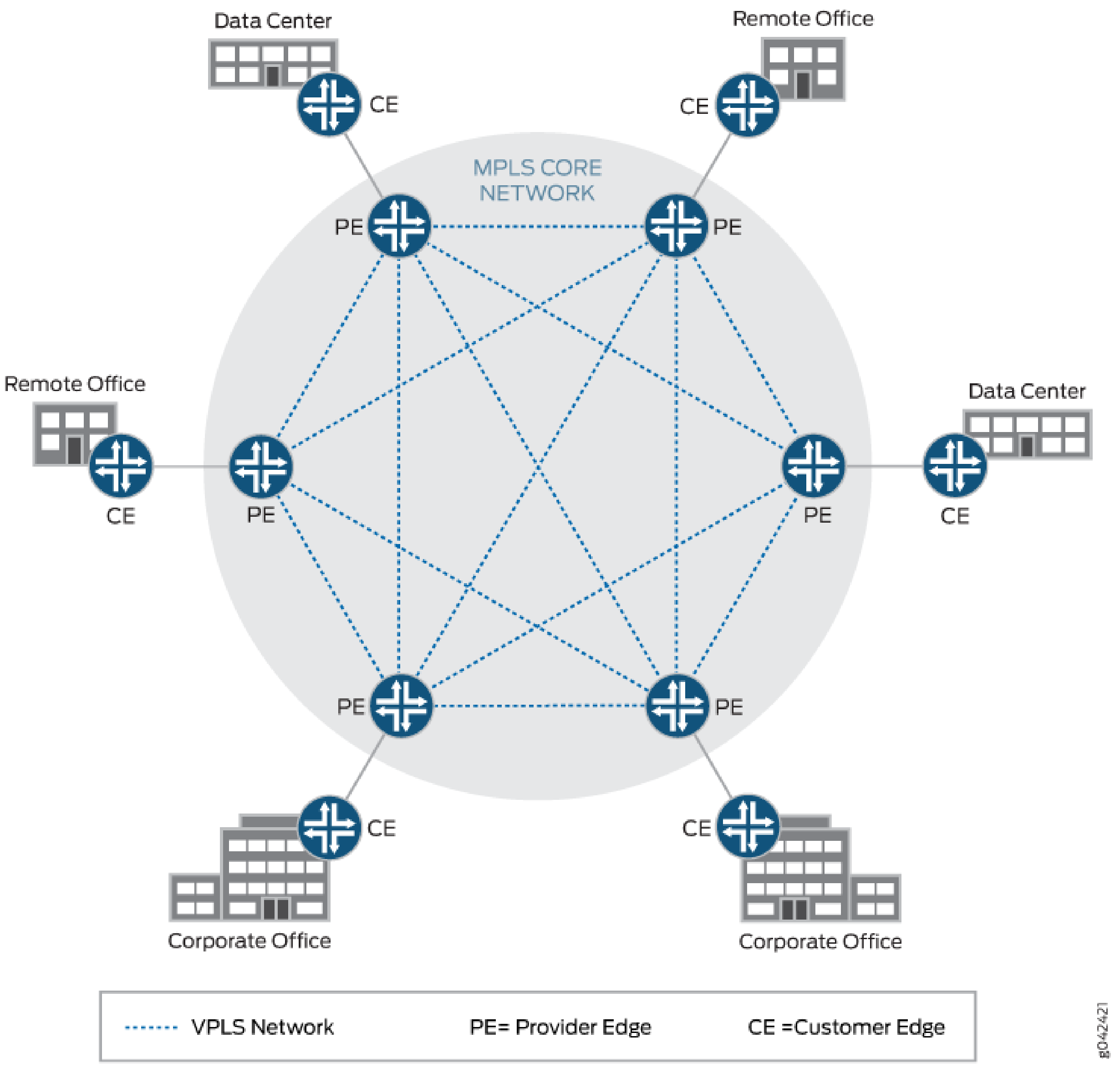
The functionality in this procedure is widely supported on devices that run Junos OS. The example shown here was tested and verified on MX Series routers running a supported Junos OS release.

Overview

IN THIS SECTION

VPLS networks create a Virtual Private LAN that provides a very close approximation of an Ethernet LAN to customers of a service provider. In a VPLS network, it is not necessary for all customers to be connected to a single LAN. Instead, the customers can be spread across two or more LANs. In the simplest sense, a VPLS network connects individual LANs across a packet-switched network so that they appear as a single LAN. See Figure 51 on page 636 for an example of a typical VPLS topology.

Figure 51: Typical VPLS Topology



Junos OS contains several DiffServ code point (DSCP) default rewrite rules that might meet your requirements. You display them with the `show class-of-service rewrite-rule` command. A partial set of the default rewrite DSCP rule mappings is shown in the following table.

You can also define your own custom rewrite-rules table, or use a mixture of the default rewrite-rules and a custom table that you create. This example uses default rewrite-rules.

Table 58: Default rewrite rules

Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
expedited-forwarding	low	ef
expedited-forwarding	high	ef
assured-forwarding	low	af11
assured-forwarding	high	af12 (DSCP/DSCP IPv6/EXP)
best-effort	low	be
best-effort	high	be
network-control	low	nc1/cs6
network-control	high	nc2/cs7

Junos OS uses the values shown in the following table for MPLS CoS in the EXP fields of the MPLS header.

Forwarding Class	Loss Priority	EXP Code Point
best-effort	low	000
best-effort	high	001
expedited-forwarding	low	010
expedited-forwarding	high	011
assured-forwarding	low	100

(Continued)

Forwarding Class	Loss Priority	EXP Code Point
assured-forwarding	high	101
network-control	low	110
network-control	high	111

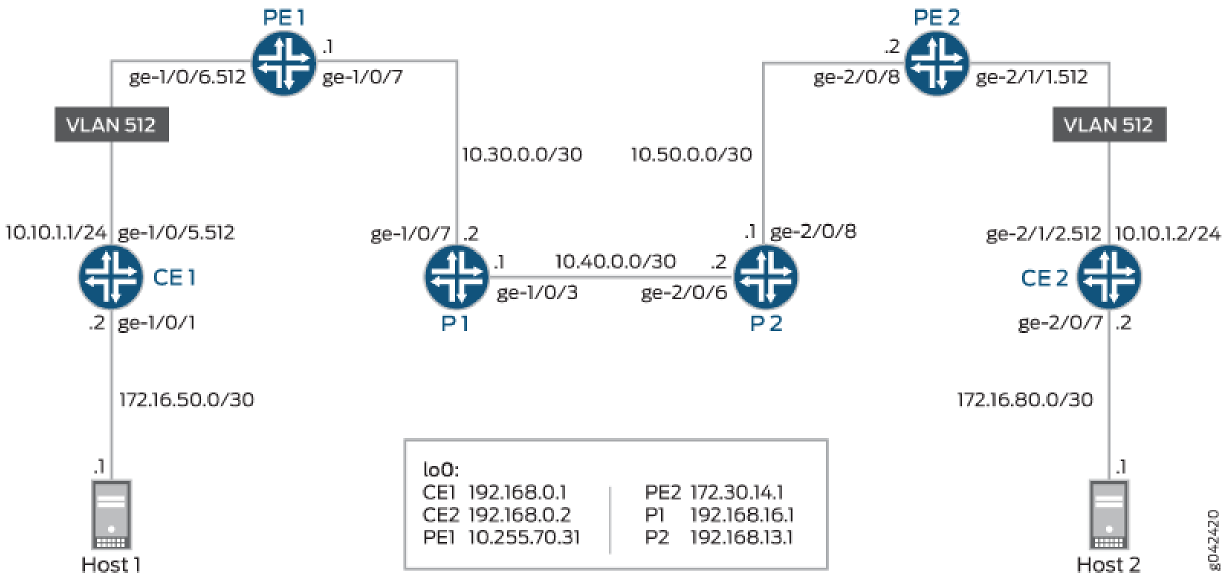
NOTE: In addition to providing the necessary information to complete the purpose of this example, this example also includes all of the commands required to recreate the VPLS network as shown in Figure 52 on page 639. A full explanation of the tasks required to configure a VPLS network is not included in this example. If you need more information regarding configuring a VPLS network, see the *VPLS User Guide for Routing Devices* at http://juniper.net/documentation and RFC 4761 at http://tools.ietf.org/html/rfc4761 .

A thorough explanation of the required CoS tasks and the underlying algorithms used in this example is beyond the scope of this document. For more information, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at www.juniper.net/books .

Topology

This example uses the topology in Figure 52 on page 639.

Figure 52: VPLS with CoS Scenario



Configuration

IN THIS SECTION

Procedure

CLI Quick Configuration

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

Device CE1

```
set  interfaces ge-1/0/1 unit 0 description to-Host1
set  interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
set  interfaces ge-1/0/1 unit 0 family inet filter input ip-v4
set interfaces ge-1/0/5 vlan-tagging
set interfaces ge-1/0/5 unit 512 description to_PE1
set interfaces ge-1/0/5 unit 512 vlan-id 512
set interfaces ge-1/0/5 unit 512 family inet address 10.10.1.1/24
set  interfaces lo0 unit 1 description loopback-interface
set  interfaces lo0 unit 1 family inet address 192.168.0.1/32
set  protocols ospf area 0.0.0.0 interface ge-1/0/5.512
set  protocols ospf area 0.0.0.0 interface ge-1/0/1.0 passive
set  protocols ospf area 0.0.0.0 interface lo0.1 passive
set  firewall family inet filter ip-v4 term tcp80 from port 80
set  firewall family inet filter ip-v4 term tcp80 then dscp ef
set  firewall family inet filter ip-v4 term 12345 from port 12345
set  firewall family inet filter ip-v4 term 12345 then dscp be
set  firewall family inet filter ip-v4 term accept then accept
set class-of-service classifiers ieee-802.1 dscp1 forwarding-class expedited-forwarding loss-
priority low code-points ef
set class-of-service classifiers ieee-802.1 dscp1 forwarding-class best-effort loss-priority low
code-points be
set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class expedited-forwarding
loss-priority low code-point 010
set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class best-effort loss-
priority low code-point 000
set class-of-service interfaces ge-1/0/5 unit 512 classifiers ieee-802.1 dscp1
set class-of-service interfaces ge-1/0/5 unit 512 rewrite-rules ieee-802.1 ieee1-c2
```

Device PE1

```


set interfaces ge-1/0/6 vlan-tagging
set interfaces ge-1/0/6 encapsulation vlan-vpls
set interfaces ge-1/0/6 unit 512 description to_vpls
set interfaces ge-1/0/6 unit 512 encapsulation vlan-vpls
set interfaces ge-1/0/6 unit 512 vlan-id 512
set interfaces ge-1/0/9 description to_P1
set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.1/30
set interfaces ge-1/0/9 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 10.255.70.31/32
set protocols mpls interface ge-1/0/9.0
set protocols bgp group to_PE2 type internal
set protocols bgp group to_PE2 local-address 10.255.70.31
set protocols bgp group to_PE2 family l2vpn signaling
set protocols bgp group to_PE2 neighbor 172.30.14.1
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
set protocols ldp interface ge-1/0/9.0
set protocols ldp interface lo0.0
set routing-options router-id 10.255.70.31
set routing-options autonomous-system 64511
set routing-instances vpls_a instance-type vpls
set routing-instances vpls_a interface ge-1/0/6.512
set routing-instances vpls_a route-distinguisher 64511:1
set routing-instances vpls_a vrf-target target:64511:1
set routing-instances vpls_a protocols vpls no-tunnel-services
set routing-instances vpls_a protocols vpls site 1 site-identifier 1
set routing-instances vpls_a protocols vpls site 1 interface ge-1/0/6.512
```

Device P1

```
set interfaces ge-1/0/3 description to_P2
set interfaces ge-1/0/3 unit 0 family inet address 10.40.0.1/30
set interfaces ge-1/0/3 unit 0 family mpls
set interfaces ge-1/0/9 description to_PE1
set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.2/30
set interfaces ge-1/0/9 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.16.1/32
set protocols mpls interface ge-1/0/9.0
set protocols mpls interface ge-1/0/3.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-1/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/0/3.0
set protocols ldp interface ge-1/0/9.0
set protocols ldp interface lo0.0
set routing-options router-id 192.168.16.1
```

Device P2

```
set interfaces ge-2/0/6 description to_P1
set interfaces ge-2/0/6 unit 0 family inet address 10.40.0.2/30
set interfaces ge-2/0/6 unit 0 family mpls
set interfaces ge-2/0/8 description to_PE2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set protocols mpls interface ge-2/0/6.0
set protocols mpls interface ge-2/0/8.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-2/0/6.0
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-2/0/6.0
set protocols ldp interface ge-2/0/8.0
set protocols ldp interface lo0.0
set routing-options router-id 192.168.13.1
```

Device PE2

```
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces ge-2/1/1 vlan-tagging
set interfaces ge-2/1/1 encapsulation vlan-vpls
set interfaces ge-2/1/1 unit 512 description to_vpls
set interfaces ge-2/1/1 unit 512 encapsulation vlan-vpls
set interfaces ge-2/1/1 unit 512 vlan-id 512
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 172.30.14.1/32
set protocols mpls interface ge-2/0/8.0
set protocols bgp group to_PE1 type internal
set protocols bgp group to_PE1 local-address 172.30.14.1
set protocols bgp group to_PE1 family l2vpn signaling
set protocols bgp group to_PE1 neighbor 10.255.70.31
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-2/0/8.0
set protocols ldp interface lo0.0
set routing-options router-id 172.30.14.1
set routing-options autonomous-system 64511
set routing-instances vpls_a instance-type vpls
set routing-instances vpls_a interface ge-2/1/1.512
set routing-instances vpls_a route-distinguisher 64511:1
set routing-instances vpls_a vrf-target target:64511:1
set routing-instances vpls_a protocols vpls no-tunnel-services
set routing-instances vpls_a protocols vpls site 2 site-identifier 2
set routing-instances vpls_a protocols vpls site 2 interface ge-2/1/1.512
```

Device CE2

```
set  interfaces ge-2/0/7 unit 0 description to-Host2
set  interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set  interfaces ge-2/0/7 unit 0 family inet filter input ip-v4
set interfaces ge-2/1/2 vlan-tagging
set interfaces ge-2/1/2 unit 512 description to-PE2
set interfaces ge-2/1/2 unit 512 vlan-id 512
set interfaces ge-2/1/2 unit 512 family inet address 10.10.1.2/24
set  interfaces lo0 unit 1 description loopback-interface
set  interfaces lo0 unit 1 family inet address 192.168.0.2/32
set  protocols ospf area 0.0.0.0 interface lo0.1 passive
set  protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set  protocols ospf area 0.0.0.0 interface ge-2/1/2.512
set  firewall family inet filter ip-v4 term tcp80 from port 80
set  firewall family inet filter ip-v4 term tcp80 then dscp ef
set  firewall family inet filter ip-v4 term 12345 from port 12345
set  firewall family inet filter ip-v4 term 12345 then dscp be
set  firewall family inet filter ip-v4 term accept then accept
set class-of-service classifiers ieee-802.1 dscp1 forwarding-class expedited-forwarding loss-
priority low code-points ef
set class-of-service classifiers ieee-802.1 dscp1 forwarding-class best-effort loss-priority low
code-points be
set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class expedited-forwarding
loss-priority low code-point 010
set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class best-effort loss-
priority low code-point 000
set class-of-service interfaces ge-2/1/2 unit 512 rewrite-rules ieee-802.1 ieee1-c2
set class-of-service interfaces ge-2/1/2 unit 512 classifiers ieee-802.1 dscp1
```

Step-by-Step Procedure

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

To configure Device CE1:

1. Configure the device interfaces.

```
[edit ]
user@CE1# set interfaces ge-1/0/1 unit 0 description to-Host1
user@CE1# set interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
user@CE1#set interfaces ge-1/0/1 unit 0 family inet filter input ip-v4
user@CE1#set interfaces lo0 unit 1 description loopback-interface
user@CE1# set interfaces lo0 unit 1 family inet address 192.168.0.1/32
```

2. Configure the VLAN parameters.

```
[edit ]
user@CE1# set interfaces ge-1/0/5 vlan-tagging
user@CE1# set interfaces ge-1/0/5 unit 512 description to_PE1
user@CE1# set interfaces ge-1/0/5 unit 512 vlan-id 512
user@CE1# set interfaces ge-1/0/5 unit 512 family inet address 10.10.1.1/24
```

3. Configure the class-of-service parameters.

```
[edit ]
user@CE1# set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class expedited-
forwarding loss-priority low code-point 010
user@CE1# set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class best-effort
loss-priority low code-point 000
user@CE1# set class-of-service classifiers ieee-802.1 dscp1 forwarding-class expedited-
forwarding loss-priority low code-points ef
user@CE1# set class-of-service classifiers ieee-802.1 dscp1 forwarding-class best-effort loss-
priority low code-points be
user@CE1# set class-of-service interfaces ge-1/0/5 unit 512 rewrite-rules ieee-802.1 ieee1-c2
user@CE1# set class-of-service interfaces ge-1/0/5 unit 512 classifiers ieee-802.1 dscp1
```

4. Configure the protocol parameters.

```
[edit ]
user@CE1# set protocols ospf area 0.0.0.0 interface ge-1/0/5.512
user@CE1# set protocols ospf area 0.0.0.0 interface ge-1/0/1.0 passive
user@CE1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
```

5. Configure the firewall DSCP rewrite parameters.

```
[edit ]
user@CE1# set firewall family inet filter ip-v4 term tcp80 from port 80
user@CE1# set firewall family inet filter ip-v4 term tcp80 then dscp ef
user@CE1# set firewall family inet filter ip-v4 term 12345 from port 12345
user@CE1# set firewall family inet filter ip-v4 term 12345 then dscp be
user@CE1# set firewall family inet filter ip-v4 term accept then accept
```

Step-by-Step Procedure

To configure Device PE1:

1. Configure the device interfaces.

```
[edit ]
user@PE1# set interfaces ge-1/0/9 description to_P1
user@PE1# set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.1/30
user@PE1# set interfaces ge-1/0/9 unit 0 family mpls
user@PE1# set interfaces lo0 unit 0 description loopback-interface
user@PE1# set interfaces lo0 unit 0 family inet address 10.255.70.31/32
```

2. Configure the VLAN parameters.

```
[edit ]
user@PE1# set interfaces ge-1/0/6 vlan-tagging
user@PE1# set interfaces ge-1/0/6 encapsulation vlan-vpls
user@PE1# set interfaces ge-1/0/6 unit 512 description to_vpls
user@PE1# set interfaces ge-1/0/6 unit 512 encapsulation vlan-vpls
user@PE1# set interfaces ge-1/0/6 unit 512 vlan-id 512
```

3. Configure the protocol parameters.

```
[edit ]
user@PE1# set protocols mpls interface ge-1/0/9.0
user@PE1# set protocols bgp group to_PE2 type internal
user@PE1# set protocols bgp group to_PE2 local-address 10.255.70.31
user@PE1# set protocols bgp group to_PE2 family l2vpn signaling
user@PE1# set protocols bgp group to_PE2 neighbor 172.30.14.1
user@PE1# set protocols ospf traffic-engineering
user@PE1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@PE1# set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
user@PE1# set protocols ldp interface ge-1/0/9.0
user@PE1# set protocols ldp interface lo0.0
```

4. Configure the routing option parameters.

```
[edit ]
user@PE1# set routing-options router-id 10.255.70.31
user@PE1# set routing-options autonomous-system 64511
```

5. Configure the routing instance parameters.

```
[edit ]
user@PE1# set routing-instances vpls_a instance-type vpls
user@PE1# set routing-instances vpls_a interface ge-1/0/6.512
user@PE1# set routing-instances vpls_a route-distinguisher 64511:1
user@PE1# set routing-instances vpls_a vrf-target target:64511:1
user@PE1# set routing-instances vpls_a protocols vpls no-tunnel-services
user@PE1# set routing-instances vpls_a protocols vpls site 1 site-identifier 1
user@PE1# set routing-instances vpls_a protocols vpls site 1 interface ge-1/0/6.512
```

Step-by-Step Procedure

To configure Device P1:

1. Configure the device interfaces.

```
[edit ]
user@P1# set interfaces ge-1/0/3 description to_P2
```

```
user@P1# set interfaces ge-1/0/3 unit 0 family inet address 10.40.0.1/30
user@P1# set interfaces ge-1/0/3 unit 0 family mpls
user@P1# set interfaces ge-1/0/9 description to_PE1
user@P1# set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.2/30
user@P1# set interfaces ge-1/0/9 unit 0 family mpls
user@P1# set interfaces lo0 unit 0 description loopback-interface
user@P1# set interfaces lo0 unit 0 family inet address 192.168.16.1/32
```

2. Configure the protocol parameters.

```
[edit ]
user@P1# set protocols mpls interface ge-1/0/9.0
user@P1# set protocols mpls interface ge-1/0/3.0
user@P1# set protocols ospf traffic-engineering
user@P1# set protocols ospf area 0.0.0.0 interface ge-1/0/3.0
user@P1# set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
user@P1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@P1# set protocols ldp interface ge-1/0/3.0
user@P1# set protocols ldp interface ge-1/0/9.0
user@P1# set protocols ldp interface lo0.0
```

3. Configure the routing options parameter.

```
[edit ]
user@P1# set routing-options router-id 192.168.16.1
```

Step-by-Step Procedure

To configure Device P2:

1. Configure the device interfaces.

```
[edit ]
user@P2# set interfaces ge-2/0/6 description to_P1
user@P2#set interfaces ge-2/0/6 unit 0 family inet address 10.40.0.2/30
user@P2# set interfaces ge-2/0/6 unit 0 family mpls
user@P2# set interfaces ge-2/0/8 description to_PE2
user@P2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@P2# set interfaces ge-2/0/8 unit 0 family mpls
```

```
user@P2# set interfaces lo0 unit 0 description loopback-interface
user@P2# set interfaces lo0 unit 0 family inet address 192.168.13.1/32
```

2. Configure the protocol parameters.

```
[edit ]
user@P2# set protocols mpls interface ge-2/0/6.0
user@P2# set protocols mpls interface ge-2/0/8.0
user@P2# set protocols ospf traffic-engineering
user@P2#  set protocols ospf area 0.0.0.0 interface ge-2/0/6.0
user@P2# set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
user@P2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@P2# set protocols ldp interface ge-2/0/6.0
user@P2# set protocols ldp interface ge-2/0/8.0
user@P2# set protocols ldp interface lo0.0
```

3. Configure the routing option parameter.

```
[edit ]
user@P2# set routing-options router-id 192.168.13.1
```

Step-by-Step Procedure

To configure Device PE2:

1. Configure the device interfaces.

```
[edit ]
user@PE2# set interfaces ge-2/0/8 description to-R1
user@PE2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@PE2# set interfaces ge-2/0/8 unit 0 family mpls
user@PE2# set interfaces lo0 unit 0 description loopback-interface
user@PE2# set interfaces lo0 unit 0 family inet address 172.30.14.1/32
```

2. Configure the VLAN parameters.

```
[edit ]
user@PE2# set interfaces ge-2/1/1 vlan-tagging
user@PE2# set interfaces ge-2/1/1 encapsulation vlan-vpls
```

```
user@PE2# set interfaces ge-2/1/1 unit 512 description to_vpls
user@PE2# set interfaces ge-2/1/1 unit 512 encapsulation vlan-vpls
user@PE2# set interfaces ge-2/1/1 unit 512 vlan-id 512
```

3. Configure the protocol parameters.

```
[edit ]
user@PE2# set protocols mpls interface ge-2/0/8.0
user@PE2# set protocols bgp group to_PE1 type internal
user@PE2# set protocols bgp group to_PE1 local-address 172.30.14.1
user@P2# set protocols bgp group to_PE1 family l2vpn signaling
user@PE2# set protocols bgp group to_PE1 neighbor 10.255.70.31
user@PE2# set protocols ospf traffic-engineering
user@PE2# set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
user@PE2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@PE2# set protocols ldp interface ge-2/0/8.0
user@PE2# set protocols ldp interface lo0.0
```

4. Configure the routing option parameters.

```
[edit ]
user@PE2# set routing-options router-id 172.30.14.1
user@PE2# set routing-options autonomous-system 64511
```

5. Configure the routing instance parameters.

```
[edit ]
user@P2# set routing-instances vpls_a instance-type vpls
user@PE2# set routing-instances vpls_a interface ge-2/1/1.512
user@PE2# set routing-instances vpls_a route-distinguisher 64511:1
user@PE2# set routing-instances vpls_a vrf-target target:64511:1
user@PE2# set routing-instances vpls_a protocols vpls no-tunnel-services
user@PE2# set routing-instances vpls_a protocols vpls site 2 site-identifier 2
user@PE2# set routing-instances vpls_a protocols vpls site 2 interface ge-2/1/1.512
```

Step-by-Step Procedure

To configure Device CE2:

1. Configure the device interfaces.

```
[edit ]
user@CE2# set interfaces ge-2/0/7 unit 0 description to-Host2
user@CE2# set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
user@CE2# set interfaces ge-2/0/7 unit 0 family inet filter input ip-v4
user@CE2# set interfaces lo0 unit 1 description loopback-interface
user@CE2# set interfaces lo0 unit 1 family inet address 192.168.0.2/32
```

2. Configure the VLAN parameters

```
[edit ]
user@CE2# set interfaces ge-2/1/2 vlan-tagging
user@CE2# set interfaces ge-2/1/2 unit 512 description to-PE2
user@CE2# set interfaces ge-2/1/2 unit 512 vlan-id 512
user@CE2# set interfaces ge-2/1/2 unit 512 family inet address 10.10.1.2/24
```

3. Configure the class-of-service parameters.

```
[edit ]
user@CE2# set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class expedited-
forwarding loss-priority low code-point 010
user@CE2# set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class best-effort
loss-priority low code-point 000
user@CE2# set class-of-service classifiers ieee-802.1 dscp1 forwarding-class expedited-
forwarding loss-priority low code-points ef
user@CE2# set class-of-service classifiers ieee-802.1 dscp1 forwarding-class best-effort loss-
priority low code-points be
user@CE2# set class-of-service interfaces ge-2/1/2 unit 512 rewrite-rules ieee-802.1 ieee1-c2
user@CE2# set class-of-service interfaces ge-2/1/2 unit 512 classifiers ieee-802.1 dscp1
```

4. Configure the protocol parameters.

```
[edit ]
user@CE2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@CE2# set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
user@CE2# set protocols ospf area 0.0.0.0 interface ge-2/1/2.512
```

5. Configure the firewall DSCP rewrite parameters.

```
[edit ]
user@CE2# set firewall family inet filter ip-v4 term tcp80 from port 80
user@CE2# set firewall family inet filter ip-v4 term tcp80 then dscp ef
user@CE2# set firewall family inet filter ip-v4 term 12345 from port 12345
user@CE2# set firewall family inet filter ip-v4 term 12345 then dscp be
user@CE2# set firewall family inet filter ip-v4 term accept then accept
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show class-of-service`, `show protocols`, `show routing-options`, `show routing-instances`, and `show firewall`, commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@CE1# show interfaces
ge-1/0/1 {
    unit 0 {
        description to-Host1;
        family inet {
            filter {
                input ip-v4;
            }
            address 172.16.50.2/30;
        }
    }
}
ge-1/0/5 {
    vlan-tagging;
    unit 512 {
        description to_PE1;
        vlan-id 512;
        family inet {
            address 10.10.1.1/24;
        }
    }
}
lo0 {
    unit 1 {
        description loopback-interface;
        family inet {
            address 192.168.0.1/32;
        }
    }
}
```

```
user@CE1# show class-of-service
classifiers {
    ieee-802.1 dscp1 {
        forwarding-class expedited-forwarding {
            loss-priority low code-points ef;
        }
        forwarding-class best-effort {
            loss-priority low code-points be;
        }
    }
}
interfaces {
    ge-1/0/5 {
        unit 512 {
            classifiers {
                ieee-802.1 dscp1;
            }
            rewrite-rules {
                ieee-802.1 ieee1-c2;
            }
        }
    }
}
rewrite-rules {
    ieee-802.1 ieee1-c2 {
        forwarding-class expedited-forwarding {
            loss-priority low code-point 010;
        }
        forwarding-class best-effort {
            loss-priority low code-point 000;
        }
    }
}
```

```
user@CE1# show protocols
ospf {
    area 0.0.0.0 {
        interface ge-1/0/5.512;
        interface ge-1/0/1.0 {
            passive;
        }
        interface lo0.1 {
            passive;
        }
    }
}
```

```
user@CE1# show firewall
family inet {
    filter ip-v4 {
        term tcp80 {
            from {
                port 80;
            }
            then dscp ef;
        }
        term 12345 {
            from {
                port 12345;
            }
            then dscp be;
        }
        term accept {
            then accept;
        }
    }
}
```

If you are done configuring Device CE1, enter `commit` from configuration mode.

```
user@PE1# show interfaces
ge-1/0/6 {
    vlan-tagging;
    encapsulation vlan-vpls;
    unit 512 {
        description to_vpls;
        encapsulation vlan-vpls;
        vlan-id 512;
    }
}
ge-1/0/9 {
    description to_P1;
    unit 0 {
        family inet {
            address 10.30.0.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 10.255.70.31/32;
        }
    }
}
```

```
user@PE1# show protocols
mpls {
    interface ge-1/0/9.0;
}
bgp {
    group to_PE2 {
        type internal;
        local-address 10.255.70.31;
        family l2vpn {
            signaling;
        }
        neighbor 172.30.14.1;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface lo0.0 {
            passive;
        }
        interface ge-1/0/9.0;
    }
}
ldp {
    interface ge-1/0/9.0;
    interface lo0.0;
}
```

```
user@PE1# show routing-options
router-id 10.255.70.31;
autonomous-system 64511;
```

```
user@PE1# show routing-instances
vpls_a {
    instance-type vpls;
    interface ge-1/0/6.512;
    route-distinguisher 64511:1;
    vrf-target target:64511:1;
    protocols {
        vpls {
            no-tunnel-services;
            site 1 {
                site-identifier 1;
                interface ge-1/0/6.512;
            }
        }
    }
}
```

If you are done configuring Device PE1, enter `commit` from configuration mode.

```
user@P1# show interfaces
ge-1/0/3 {
    description to_P2;
    unit 0 {
        family inet {
            address 10.40.0.1/30;
        }
        family mpls;
    }
}
ge-1/0/9 {
    description to_PE1;
    unit 0 {
        family inet {
            address 10.30.0.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.16.1/32;
        }
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.16.1/32;
        }
    }
}
```

```
user@P1# show protocols
mpls {
    interface ge-1/0/9.0;
    interface ge-1/0/3.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-1/0/3.0;
        interface ge-1/0/9.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/3.0;
    interface ge-1/0/9.0;
    interface lo0.0;
}
```

```
user@P1# show routing-options
router-id 192.168.16.1;
```

If you are done configuring Device P1, enter `commit` from configuration mode.

```
user@P2# show interfaces
ge-2/0/6 {
    description to_P1;
    unit 0 {
        family inet {
            address 10.40.0.2/30;
        }
        family mpls;
    }
}
ge-2/0/8 {
    description to_PE2;
    unit 0 {
        family inet {
            address 10.50.0.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
```

```
user@P2# show protocols
mpls {
    interface ge-2/0/6.0;
    interface ge-2/0/8.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-2/0/6.0;
        interface ge-2/0/8.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-2/0/6.0;
    interface ge-2/0/8.0;
    interface lo0.0;
}
```

```
user@P2# show routing-options
router-id 192.168.13.1;
```

If you are done configuring Device P2, enter `commit` from configuration mode.

```
user@PE2# show interfaces
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.50.0.2/30;
        }
        family mpls;
    }
}
ge-2/1/1 {
    vlan-tagging;
    encapsulation vlan-vpls;
    unit 512 {
        description to_vpls;
        encapsulation vlan-vpls;
        vlan-id 512;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 172.30.14.1/32;
        }
    }
}
```

```
user@PE2# show protocols
mpls {
    interface ge-2/0/8.0;
}
bgp {
    group to_PE1 {
        type internal;
        local-address 172.30.14.1;
        family l2vpn {
            signaling;
        }
        neighbor 10.255.70.31;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-2/0/8.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-2/0/8.0;
    interface lo0.0;
}
```

```
user@PE2# show routing-options
router-id 172.30.14.1;
autonomous-system 64511;
```

```
user@PE2# show routing-instances
vpls_a {
    instance-type vpls;
    interface ge-2/1/1.512;
    route-distinguisher 64511:1;
    vrf-target target:64511:1;
    protocols {
        vpls {
            no-tunnel-services;
            site 2 {
                site-identifier 2;
                interface ge-2/1/1.512;
            }
        }
    }
}
```

If you are done configuring Device PE2, enter `commit` from configuration mode.

```
user@CE2# show interfaces
ge-2/0/7 {
    unit 0 {
        description to-Host2;
        family inet {
            filter {
                input ip-v4;
            }
            address 172.16.80.2/30;
        }
    }
}
ge-2/1/2 {
    vlan-tagging;
    unit 512 {
        description to-PE2;
        vlan-id 512;
        family inet {
            address 10.10.1.2/24;
        }
    }
}
lo0 {
    unit 1 {
        description loopback-interface;
        family inet {
            address 192.168.0.2/32;
        }
    }
}
user@CE2# show class-of-service
classifiers {
    ieee-802.1 dscp1 {
        forwarding-class expedited-forwarding {
            loss-priority low code-points ef;
        }
        forwarding-class best-effort {
            loss-priority low code-points be;
        }
    }
}
interfaces {
    ge-2/1/2 {
        unit 512 {
            classifiers {
                ieee-802.1 dscp1;
            }
            rewrite-rules {
                ieee-802.1 ieee1-c2;
            }
        }
    }
}
rewrite-rules {
    ieee-802.1 ieee1-c2 {
        forwarding-class expedited-forwarding {
            loss-priority low code-point 010;
        }
        forwarding-class best-effort {
            loss-priority low code-point 000;
        }
    }
}
user@CE2# show protocols
ospf {
    area 0.0.0.0 {
        interface lo0.1 {
            passive;
        }
        interface ge-2/0/7.0 {
            passive;
        }
        interface ge-2/1/2.512;
    }
}
user@CE2# show firewall
family inet {
    filter ip-v4 {
        term tcp80 {
            from {
                port 80;
            }
            then dscp ef;
        }
        term 12345 {
            from {
                port 12345;
            }
            then dscp be;
        }
        term accept {
            then accept;
        }
    }
}
```

If you are done configuring Device CE2, enter `commit` from configuration mode.

Verification

IN THIS SECTION

Confirm that the configuration is working properly by verifying that the DSCP aliases are maintained from Device CE1 to Device CE2.

Clearing the Firewall Counters

Purpose

Confirm that the firewall counters are cleared.

Action

On Device CE2, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@CE2> clear firewall all
```

Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Verifying the Results

Purpose

Send traffic into the network that can be verified at Device CE2.

Action

Configure a new firewall on Device CE2 if you want to verify that the traffic that is being transmitted to Device Host2 from Device Host1 still has the correct DSCP aliases. The following commands create and apply the firewall filter that displays the traffic counts for each code point alias:

```
user@CE2# set firewall family inet filter count term be from dscp be
user@CE2# set firewall family inet filter count term be then count be
user@CE2# set firewall family inet filter count term ef from dscp ef
user@CE2# set firewall family inet filter count term ef then count ef
user@CE2# set firewall family inet filter count term accept then accept
user@CE2# set interfaces ge-2/0/7 unit 0 family inet filter output count
```

When you are done configuring Device CE2, enter `commit` from configuration mode.

When you are done testing, you can leave the counting filter in place, or remove it.

1. On Device Host1 use a traffic generator to send 20 TCP packets with a source port of 80 into the network.

   The -s flag sets the source port. The -k flag causes the source port to remain steady instead of incrementing. The -c flag sets the number of packets to 20.

   Repeat the task using a source port of 12345.

   ```
   [user@host1]#  hping 172.16.80.1  -s 80  -k -c 20
   [user@host1]#  hping 172.16.80.1  -s 12345  -k -c 20
   ```

2. On Device CE2, display the firewall counters by using the `show firewall` command.

   ```
   user@CE2> show firewall
   show firewall

   Filter: __CE2/count
   Counters:
   Name                                               Bytes              Packets
   be                                                   800                   20
   ef                                                   800                   20
   ```

Meaning

The code point aliases set by Device CE1 are maintained across the VPLS backbone and appear intact at Device CE2.

RELATED DOCUMENTATION

Example: Configuring and Applying Scheduler Maps | 361

Assigning Rewrite Rules on a Per-Customer Basis Using Policy Maps

SUMMARY

This topic describes the purpose and configuration of policy maps. Policy maps enable you to assign rewrite rules on a per-customer basis.

IN THIS SECTION

Policy Maps Overview

Most commonly, packet marking (that is, setting rewrite rules) in Junos uses the forwarding class and loss priority determined through a behavior aggregate (BA) classifier or multifield classifier. The forwarding class and loss priority are also used to decide queuing and scheduling behavior. This approach does not allow you to directly assign rewrite rules to each customer because of the limited number of combinations of forwarding class and loss priority. When a new customer is added, setting rewrite rules using this approach requires changes to the configuration on the core interfaces, which is undesirable as one mistake can affect traffic from all customers.

An alternative packet marking scheme, called *policy map*, enables you to define rewrite rules on a per-customer basis (that is, for each customer) separate from queuing and scheduling behavior. The policy map makes it possible to use any packet field to identify a given flow and specify a rewrite value for that flow.

A policy map is defined at the `[edit class-of-service policy-map]` hierarchy level. Depending on your platform and software release, the policy map can define the following types of packet marking:

- IPv4 Precedence with the following options:
  - `proto-ip` – Mark the packet for IPv4 to IPv4 traffic.
  - `proto-mpls` – Mark the packet for an IPv4 packet entering an MPLS tunnel.
  - `proto-all` – Enable IP header marking.
- IPv6 Precedence with the following options:
  - `proto-ip` – Mark the packet for IPv6 to IPv6 traffic.
  - `proto-mpls` – Mark the packet for an IPv6 packet entering an MPLS tunnel.
- IPv4 DSCP with the following options:
  - `proto-ip` – Mark the packet for IPv4 to IPv4 traffic.
  - `proto-mpls` – Mark the packet for an IPv4 packet entering an MPLS tunnel.
  - `proto-all` – Enable IP header marking.
- IPv6 DSCP with the following options:
  - `proto-ip` – Mark the packet for IPv6 to IPv6 traffic.
  - `proto-mpls` – Mark the packet for an IPv6 packet entering an MPLS tunnel.
  - `proto-all` – Enable IPv6 header marking.
- MPLS EXP with the following options:
  - `all-label` – Mark all labels.
  - `outer-label` – Mark only the outer label.
- IEEE 802.1p with the following options:
  - `outer` – Mark only the outer VLAN header.
  - `outer-and-inner` – Mark both the outer and inner VLAN headers.
- IEEE 802.1ad with the following options:
  - `outer` – Mark only the outer VLAN header.
  - `outer-and-inner` – Mark both the outer and inner VLAN headers.

NOTE: To enable the policy map feature:

- On most devices, enable `enhanced-ip`, `enhanced-ethernet`, or `enhanced-mode` under `[edit chassis network-services]`.
- On PTX10002-36QDD routers, enable `policy-map-marking` on each egress logical interface that you want to do policy map marking. You enable `policy-map-marking` at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`unit-number`*`]` hierarchy level.

NOTE: Policy maps have the following configuration restrictions:

- When configuring both `proto-ip` and `proto-mpls` options for `inet-precedence`, `inet6-precedence`, `dscp`, or `dscp-ipv6`, you must configure both options with the same code point or code point alias.
- You cannot configure `inet-precedence` and `dscp` in the same policy map.
- You cannot configure `inet6-precedence` and `dscp-ipv6` in the same policy map.
- In case of MPLS SWAP/PUSH operation, only the new labels are marked on all label-switching routers (LSRs), except the penultimate hop case where if it exposes the next label in the stack, then the exposed label is marked. Therefore, with the penultimate hop, the service label is changed.
- You cannot configure `ieee-802.1` and `ieee-802.1ad` in the same policy map.
- You cannot configure both `outer` and `outer-and-inner` options for `ieee-802.1` and `ieee-802.1ad` code points in the same policy map.
- For IEEE 802.1ad with the `outer-and-inner` option, the discard eligibility (DE) bit is marked only for the outer VLAN header. For the inner VLAN header, only the three CoS Bits are marked.

You assign a policy map to a customer through a firewall action on an ingress or egress firewall filter (where the match conditions identify the customer). Alternatively, you can assign a policy map to an ingress interface or a routing instance. You can assign multiple policy maps to a customer, one for each of the customer’s traffic flows. Also, you can assign a single policy map to multiple customers.

A policy map is executed on a packet just before it is queued, so it overrides any other packet- marking scheme that was previously applied to the packet.

Configuring Policy Maps

To assign rewrite rules on a per-customer basis:

1. Configure a policy map.

   ```
   [edit class-of-service policy-map policy-map-name]
   user@host# set inet-precedence proto-ip code-point [alias | bits];
   user@host# set inet-precedence proto-mpls code-point [alias | bits]
   user@host# set inet-precedence proto-all code-point [alias | bits]
   user@host# set inet6-precedence proto-ip code-point [alias | bits];
   user@host# set inet6-precedence proto-mpls code-point [alias | bits]
   user@host# set dscp proto-ip code-point [alias | bits]
   user@host# set dscp proto-mpls code-point [alias | bits]
   user@host# set dscp proto-all code-point [alias | bits]
   user@host# set dscp-ipv6 proto-ip code-point [alias | bits]
   user@host# set dscp-ipv6 proto-mpls code-point [alias | bits]
   user@host# set dscp-ipv6 proto-all code-point [alias | bits]
   user@host# set exp all-label code-point [alias | bits]
   user@host# set exp outer-label code-point [alias | bits]
   user@host# set ieee-802.1 outer code-point [alias | bits]
   user@host# set ieee-802.1 outer-and-inner code-point [alias | bits]
   user@host# set ieee-802.1ad outer code-point [alias | bits]
   user@host# set ieee-802.1ad outer-and-inner code-point [alias | bits]
   ```

   For example:

   ```
   [edit class-of-service]
   user@host# set policy-map pm1 dscp proto-ip code-point 111000
   user@host# set policy-map pm1 ieee-802.1 outer code-point 001
   user@host# show
   policy-map {
       pm1 {
           dscp proto-ip code-point 111000;
           ieee-802.1 outer code-point 001;
       }
   }
   ```

2. Apply the policy map.

   - Apply the policy map to an ingress or egress firewall filter.

     ```
     [edit firewall family protocol-family-name filter filter-name]
     user@host# set term term-name from match-conditions
     user@host# set term term-name then policy-map policy-map-name
     ```

For example:

```
[edit firewall family inet filter f1]
user@host# set term t1 from address 10.2.2.0/24
user@host# set term t1 then policy-map pm1
user@host# show
term t1 {
    from {
        address {
            10.2.2.0/24;
        }
    }
    then policy-map pm1;
}
```

NOTE: In this example, every IPv4 packet arriving from IP address 10.2.2.0/24 is assigned a DSCP value of `111000`.

- Alternatively, apply the policy map to a routing instance.

```
[edit class-of-service]
user@host# set routing-instances routing-instance-name  policy-map policy-map-name
```

For example:

```
[edit class-of-service]
user@host# set routing-instances r1 policy-map pm1
user@host# show
routing-instances {
    r1 {
        policy-map pm1;
    }
}
```

NOTE: In this example, every IPv4 packet in routing instance `r1` is assigned a DSCP value of `111000`.

- Alternatively, apply the policy map directly to an *ingress* interface.

```
[edit class-of-service]
user@host# set interfaces interface-name  unit logical-unit-number  policy-map policy-map-
name
```

For example:

```
[edit class-of-service]
user@host# set interfaces xe-4/0/0 unit 0 policy-map pm1
user@host# show
    xe-4/0/0 {
        unit 0 {
            policy-map pm1;
        }
    }
```

NOTE: In this example, every IPv4 packet arriving on interface `xe-4/0/0 unit 0` is assigned a DSCP value of `111000`.

Use Policy Maps to Pass Meta Data

On PTX10002-36QDD routers, you can also use policy maps to pass meta data between firewall filters. That is, a policy map value set in an ingress filter can be matched on in an egress filter, enabling you to perform any firewall action based on that meta data.

NOTE: Use of policy maps for this purpose is not a CoS function and does not require a QoS license. When committing a configuration for this purpose, you can safely ignore any notice of the requirement of a QoS license.

As an example, consider a device with an ingress interface of xe-0/0/0 and an egress interface of xe-1/0/0. You can define an input firewall filter for xe-0/0/0 with a policy map as an action based on any desired match conditions. You can then define an egress firewall filter for xe-1/0/0 with that policy map as a match condition and apply any desired actions for packets that have that policy map applied.

1. Enable policy map marking on each egress interface that you want to do policy map marking.

   ```
   [edit class-of-service]
   user@host# set interfaces interface-name  unit logical-unit-number  policy-map-marking
   ```

   For example:

   ```
   [edit class-of-service]
   user@host# set interfaces xe-1/0/0 unit 0 policy-map-marking
   user@host# show
       xe-1/0/0 {
           unit 0 {
               policy-map-marking;
           }
       }
   ```

2. Configure a policy map.

   CAUTION: Be sure to define this policy map so it does not interfere with your packet marking (rewrite rule) scheme. For example, use a packet marking type here that you don't use for your normal packet marking.

   ```
   [edit class-of-service policy-map policy-map-name]
   user@host# set inet-precedence proto-ip code-point [alias | bits];
   user@host# set inet-precedence proto-mpls code-point [alias | bits]
   user@host# set inet-precedence proto-all code-point [alias | bits]
   user@host# set inet6-precedence proto-ip code-point [alias | bits];
   user@host# set inet6-precedence proto-mpls code-point [alias | bits]
   user@host# set dscp proto-ip code-point [alias | bits]
   user@host# set dscp proto-mpls code-point [alias | bits]
   user@host# set dscp proto-all code-point [alias | bits]
   user@host# set dscp-ipv6 proto-ip code-point [alias | bits]
   user@host# set dscp-ipv6 proto-mpls code-point [alias | bits]
   user@host# set dscp-ipv6 proto-all code-point [alias | bits]
   user@host# set exp all-label code-point [alias | bits]
   user@host# set exp outer-label code-point [alias | bits]
   user@host# set ieee-802.1 outer code-point [alias | bits]
   user@host# set ieee-802.1 outer-and-inner code-point [alias | bits]
   ```

```
user@host# set ieee-802.1ad outer code-point [alias | bits]
user@host# set ieee-802.1ad outer-and-inner code-point [alias | bits]
```

For example:

```
[edit class-of-service]
user@host# set policy-map pm1 ieee-802.1ad outer code-point 1111
user@host# show
policy-map {
    pm1 {
        ieee-802.1ad outer code-point 1111;
    }
}
```

3. Define an input firewall filter with the policy map as an action.

```
[edit firewall family protocol-family-name filter filter-name]
user@host# set term term-name from match-conditions
user@host# set term term-name then policy-map policy-map-name
```

For example:

```
[edit firewall family inet filter IFF1]
user@host# set term t1 from match-conditions
user@host# set term t1 then policy-map pm1
user@host# show
term t1 {
    from {
        match-conditions;
    }
    then policy-map pm1;
}
```

4. Apply the input firewall filter to an ingress interface.

```
[edit interfaces interface-name  unit logical-unit-number ]
user@host# set family family-type filter input filter-name
```

For example:

```
[edit interfaces xe-0/0/0 unit 0]
user@host# set family inet filter input IFF1
user@host# show
    family inet {
        filter input {
            IFF1;
        }
    }
```

5. Define an output firewall filter with the policy map as a match condition.

```
[edit firewall family protocol-family-name filter filter-name]
user@host# set term term-name from policy-map policy-map-name
user@host# set term term-name then actions
```

For example:

```
[edit firewall family inet filter OFF1]
user@host# set term t1 from policy-map pm1
user@host# set term t1 then actions
user@host# show
term t1 {
    from {
        policy-map {
            pm1;
        }
    }
    then actions;
}
```

6. Apply the output firewall filter to an egress interface.

```
[edit interfaces interface-name  unit logical-unit-number ]
user@host# set family family-type filter output filter-name
```

For example:

```
[edit interfaces xe-1/0/0 unit 0]
user@host# set family inet filter output OFF1
user@host# show
    family inet {
        filter output {
            OFF1;
        }
    }
```

Platform-Specific Policy Map Behavior

Use Feature Explorer to confirm platform and release support for policy maps.

Use the following table to review platform-specific behaviors for your platform:

Platform	Difference
PTX10002-36QDD	• Enable `policy-map-marking` on each egress logical interface that you want to do policy map marking. You enable `policy-map-marking` at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`unit-number`*`]` hierarchy level. • INET-Precedence option marks both IPv4 and IPv6 packets. • Due to hardware limitations, you can't create a policy map with both Layer 2 and Layer 3 packet marking types. • Enabling `policy-map` action on an egress filter interface does not have any impact on rewrite values. • You can also use policy maps to pass meta data between firewall filters. That is, a policy map value set in an ingress filter can be matched on in an egress filter.

RELATED DOCUMENTATION

policy-map (Class of Service)

show class-of-service policy-map

Host Outbound Traffic IEEE802.1p Rewrite

SUMMARY

This topic describes how to rewrite the IEEE 802.1p field in the Ethernet frame header for host outbound traffic, that is, traffic generated by the Routing Engine. You can set a global value for the priority code point that applies to all host outbound traffic. Additionally, or alternatively, you can specify that rewrite rules are applied to all host outbound traffic on egress logical interfaces. These are rules that you have previously configured to set the IEEE 802.1p field for data traffic on those interfaces.

IN THIS SECTION

To configure the IEEE 802.1p field settings:

1. (Optional) Specify a global default value for the IEEE 802.1p field for all host outbound traffic.

2. (Optional) Specify that the IEEE 802.1p rewrite rules for the egress logical interfaces are applied to all host outbound traffic on those interfaces.

Configure a Global Default IEEE 802.1p Value for All Host Outbound Traffic

SUMMARY

This task describes how to configure a global default value for the IEEE 802.1p field for all host outbound traffic.

To configure a global default value for the IEEE 802.1p field:

1. Specify the value. The value can be a three-bit binary number or code point alias.

   ```
   [edit class-of-service host-outbound-traffic ieee-802.1]
   user@host# set default value
   ```

2. For example, specify that a value of 010 is applied to all host outbound traffic:

   ```
   [edit class-of-service host-outbound-traffic ieee-802.1]
   user@host# set default 010
   ```

Apply Egress Interface Rewrite Rules to the IEEE 802.1p Field for All Host Outbound Traffic on an Interface

SUMMARY

This topic describes how to apply rewrite rules for egress logical interfaces to the IEEE 802.1p field for all host outbound traffic on those interfaces.

This task requires separately configured rewrite rules that map packet loss priority information to the code point value in the 802.1p field for data traffic on egress logical interfaces. See "Rewriting Packet Headers to Ensure Forwarding Behavior" on page 558.

To configure the rewrite rules:

1. Configure the CoS rewrite rules to map the forwarding class to the desired value for the 802.1p field. See "Configuring Rewrite Rules" on page 564.

2. Associate the rewrite rules to the desired egress logical interfaces. See "Applying Rewrite Rules to Output Logical Interfaces" on page 577.

3. (Optional) Configure the forwarding class for host outbound traffic. Do not configure this forwarding class if you want to use the default forwarding class assignment (input classification). See "Overriding the Input Classification" on page 340.

To configure the rewrite rules to apply to the host outbound traffic IEEE 802.1p field:

NOTE: Enabling IEEE 802.1p rewrite rules for host outbound traffic without creating any corresponding IEEE 802.1p rewrite rules sets the IEEE 802.1p code point to be automatically 000 for all host generated traffic that exits that interface.

Configure the rewrite rules:

```
[edit class-of-service host-outbound-traffic ieee-802.1]
user@host# set rewrite-rules
```

For example:

```
[edit class-of-service]
rewrite-rules {
    ieee-802.1 rewrite_foo {
        forwarding-class network-control {
            loss-priority low code-point 101;
        }
    }
}
interfaces {
    et-1/0/0 {
        unit 100 {
            rewrite-rules {
                ieee-802.1 rewrite_foo vlan-tag outer-and-inner;
            }
        }
    }
}
host-outbound-traffic {
    forwarding-class network-control;
    ieee-802.1 {
        rewrite-rules;
    }
}
```

Platform-Specific Host Outbound Traffic Rewrite Behavior

Use Feature Explorer to confirm platform and release support for 802.1p rewrite rules for host outbound traffic.

Use the following table to review platform-specific behaviors for your platform:

Platform	Difference
ACX7000 Series	ACX7000 Series routers support only the `default` option to configure a global default value for the IEEE 802.1p field for all host outbound traffic.
PTX10000 Series	PTX10000 Series routers support only the `default` option to configure a global default value for the IEEE 802.1p field for all host outbound traffic.

RELATED DOCUMENTATION

Rewriting Packet Headers to Ensure Forwarding Behavior | **558**

3

PART

Configuring Platform-Specific Functionality

CHAPTER 11

Configuring Class of Service on ACX Series Universal Metro Routers

IN THIS CHAPTER

CoS on ACX Series Routers Features Overview

IN THIS SECTION

NOTE: Unless otherwise noted in the following topics, CoS on ACX Series Universal Metro Routers functions the same as CoS on other routers, and information common CoS functionality can be found in the Configuring Class of Service section of Class of Service User Guide for Routers.

CoS on ACX routers running Junos OS

The following key CoS features are supported on ACX Series Universal Metro Routers:

- Physical interface-based classifiers at the `[edit class-of-service interfaces` *`interfaces-name`*`]` hierarchy level
- Fixed classification for all ingress packets traversing a *logical interface* to a single forwarding class. Fixed classification is supported on all interface types.
- EXP bits located in each MPLS label and used to encode the CoS value of a packet as it traverses a label-switched path (LSP). To configure global EXP bits, include the `exp` statement at the `[edit class-of-service system-defaults classifiers]` hierarchy level.
- *Rewrite rules* at the physical and logical interface levels including the following: IP type-of-service (ToS), DSCP, MPLS EXP bit value, and IEEE 802.1p bit value.
- Attachment of the following rewrite rules to the physical interface at the `[edit class-of-service interfaces` *`interface-name`* `rewrite-rules]` hierarchy level: IP ToS, DSCP, and IEEE 802.1p bit value.
- Rewrite rules for MPLS EXP bits on the logical interface at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`unit-number`* `rewrite-rule]` hierarchy level.

  NOTE: Fine-grained rewrite is not possible, even when you use multifield filters, because of the application-specific integrated circuit (ASIC) limitation.

Queuing and scheduling features include:

- Support for up to eight forwarding classes.
- Support for up to eight egress queues per port.
- Internal buffer of 2 MB with per-egress queue buffer management.
- Three weighted random early detection (WRED) curves for TCP and one WRED curve for non-TCP. There are two fill levels and two drop probabilities per WRED curve; the drop probability corresponding to the first fill must be zero.

- Strict-priority and weighted deficit round-robin scheduling.
- Multiple strict-priority queues per port.
- Per-queue committed information rate (CIR) and peak information rate (PIR).
- Per-physical-port shaping.
- Up to four levels of hierarchical scheduling, depending on the platform.

Queue statistics features include:

- Per-egress-queue enqueue statistics in packets, bytes, packets per second (pps), and bits per second (bps).
- Per-egress-queue transmit statistics in packets, bytes, pps, and bps.
- Per-egress-queue drop statistics in packets and pps.

CoS on ACX routers running Junos OS Evolved

ACX7000 routers support the following CoS features:

- Behavior aggregate (BA) classifiers (Inet-Prec/DSCP/DSCP-v6/EXP/IEEE-802.1p/IEEE-802.1ad) at the logical interface level
- Fixed and multifield classifiers
- Maximum of eight forwarding classes
- Eight virtual output queues (VOQ) per port
- Eight scheduling priority levels
- Multiple `strict-high` priority queues with round robin (RR) scheduling
- Multiple `low` priority queues with weighted fair queue (WFQ) scheduling
- CIR and PIR for `low` priority queues
- PIR at the port level
- Rewrite rules (Inet-Prec/DSCP/DSCP-v6/EXP/IEEE-802.1p/IEEE-802.1ad) at the logical interface level
- Hierarchical class of service (HCoS) features including:
  - Four levels of hierarchical scheduling (physical interfaces, logical interface sets, logical interfaces, queues)

  - Support for Layer 3 VPN, Layer 2 VPN, Layer 2 Circuit, VPLS, and EVPN services
  - Per-queue buffer configuration
  - HCoS on aggregated Ethernet (`ae-`) interfaces except on logical interface sets
  - Six scheduling priority levels (low-latency, strict-high, high, medium-high, medium-low, low (low-high, low-medium, low)
- Host outbound traffic forwarding class assignment and global default IEEE 802.1p rewrite

NOTE: To apply CoS to an aggregated Ethernet (AE) interface, do not configure CoS on a physical interface and then add the physical interface to the AE interface. First configure the AE interface, then configure CoS on the AE interface.

NOTE: ACX routers running Junos OS Evolved do not support the following CLI show commands:

- show class-of-service forwarding-table
- show class-of-service routing-instance
- show class-of-service system-defaults

RELATED DOCUMENTATION

Configuring CoS on ACX Series Routers | 687

Hierarchical Class of Service in ACX Series Routers | 717

DSCP Propagation and Default CoS on ACX Series Routers

IN THIS SECTION

Junos OS Evolved uses default CoS and rewrite functions on ACX Series routers. As a result, Differentiated Services code points (DSCP) can't propagate from the native payload to the VXLAN tunnel header by default.

NOTE: Use the statement `set system packet-forwarding-options no-ip-tos-rewrite` to enable DSCP propagation for VXLAN traffic. When you configure or delete DSCP propagation, the `evo-pfemand` process restarts.

This statement can handle IP code point propagation, but VLAN Priority Code Point (PCP) propagation is not supported.

Benefits of DSCP Propagation and Default CoS Support for EVPN VXLANs

The `set system packet-forwarding-options no-ip-tos-rewrite` statement enables the DSCP field from the incoming packet to propagate to the VXLAN encapsulated packet. This allows CoS to be applied to VXLAN encapsulated packets as they traverse the network.

Default CoS and DSCP propagation on VXLAN interfaces

By default, the DSCP information in the incoming (native) payload does not propagate to the VXLAN tunnel header. Therefore, the DSCP of the VXLAN packet is set to zero. With the `set system packet-forwarding-options no-ip-tos-rewrite` statement, the DSCP of the native payload propagates to the VXLAN tunnel header.

The Layer 3 classifier (inet-precedence, DSCP, DSCPv6) at the ingress interface classifies packets to a forwarding class and loss priority. With the `set system packet-forwarding-options no-ip-tos-rewrite` statement, the Layer 3 classifier classifies packets using the DSCP of the native packet since the DSCP of the native payload is always retained.

The Table 59 on page 684 provide a summary of the CoS, Layer 3, and L3 and L2 classifier behavior when you enable the `set system packet-forwarding-options no-ip-tos-rewrite` statement.

Table 59: CoS Behavior with CLI Statement

CoS Behavior with CLI Statement	CoS Behavior without CLI Statement
Disables L3 rewrite	Existing CoS function
Retains the DSCP of the native payload	Retains the DSCP of the native payload

Table 59: CoS Behavior with CLI Statement *(Continued)*

CoS Behavior with CLI Statement	CoS Behavior without CLI Statement
Enables DSCP propagation for both VXLAN L2 and VXLAN L3, with the L3 classifier explicitly attached to the user-to-network (UNI) interface for an L2.	DSCP of the VXLAN packet is set to zero

Layer 3 Classifier Behavior

Table 60: Layer 3 Classifier Behavior

Ingress Classifier Type	Egress Rewrite Type	Ingress Packet DSCP	Support for Rewrite as per Configured Rewrite
L3 (DSCP, DSCP IPv6, IP Precedence) on L2 or L3 IFL	L3	not applicable	Preserves only the configured rewrite
L3 (DSCP, DSCP IPv6, IP Precedence) on L2 or L3 IFL	IEEE 802.1p/ad, EXP	not applicable	Preserves only the configured rewrite Does not allow rewrite or preserve the configured rewrite for IEEE 802.1p and IEEE 802.1pad/EXP
L2 (IEEE 802.1p/ad) on L2 IFL	IEEE 802.1p/ad	not applicable	Supports rewrite and preserves if DSCP is present
MPLS EXP	IEEE 802.1p/ad, MPLS EXP	not applicable	Yes
L3 (DSCP, DSCP IPv6, IP Precedence)	IEEE 802.1p/ad, EXP	0 to 7	The rewrite configuration is preserved if the ingress DSCP matches the forwarding class id used in the rewrite rule. For IEEE 802.1p, EXP rewrite occurs per the rewrite rule IEEE 802.1p, EXP rewrite occurs as per the rewrite rule (forwarding class, loss priority to IEEE 802.1p/EXP). For IEEE 802.1p, the EXP bits are set to zero.

Table 60: Layer 3 Classifier Behavior *(Continued)*

Ingress Classifier Type	Egress Rewrite Type	Ingress Packet DSCP	Support for Rewrite as per Configured Rewrite
L3 (DSCP, DSCP IPv6, IP Precedence)	IEEE 802.1p/ad, EXP	More than 7	For IEEE 802.1p, the EXP bits are set to zero

The Table 61 on page 686 provides a summary of the L3 and L2 classifier behavior when you enable the CLI statement.

Table 61: L3 and L2 Classifier Behavior with CLI Statement

Input Packet	Type of Interface at UNI	Classifier	User Configuration	Ingress Remark	Egress Rewrite	VXLAN Encapsulated Packet DSCP	Description
L3 IPv4 and IPv6	IRB/L3 UNI	DSCP default	None	No	No	PCP/DEI of inner packet	None
L2 carrying IPv4 and IPv6	L2	PCP default	DSCP classifier	No	No	PCP/DEI of inner packet	User needs to configure DSCP classifier explicitly on L2 UNI.
Pure L2 (no IP payload)	L2	PCP default	DSCP classifier	No	No	PCP /DEI of inner packet having VLAN tag Example: PCP/DEI bits = 100/1 => DSCP = 1001 =9 (decimal); DSCP =0 with untagged packet; DSCP=0 with native tagging	User needs to configure DSCP classifier explicitly on L2 UNI; On QFX, DSCP=0 on VXLAN encapsulated packet irrespective of single tagged, untagged and native tagged packets at L2 UNI.

Limitations

When you enable the `set system packet-forwarding-options no-ip-tos-rewrite` statement on an ACX Series router:

- The IP DSCP or IP Precedence rewrite is disabled in the system and network. As a result, you can't execute L3 rewrite (DSCP, DSCP IPv6, IP precedence).
- IEEE 802.1p code point bits in the packet header retain their original header. IEEE 802.1p propagation is not supported.

RELATED DOCUMENTATION

Configuring CoS on ACX Series Routers | 687

Applying DSCP and DSCP IPv6 Classifiers on ACX Series Routers | 694

Configuring CoS on ACX Series Routers

Physical interface-based classifiers are supported at the `[edit class-of-service interfaces` *`interfaces-name`*`]` hierarchy level. EXP bits are located in each MPLS label and used to encode the CoS value of a packet as it traverses an LSP. To configure global EXP bits, include the `exp` statement at the `[edit class-of-service system-defaults classifiers]` hierarchy level.

To configure CoS on ACX Series routers:

1. Configure the class of service.

```
[edit]
user@host# edit class-of-service
```

2. Configure the rewrite rules.

```
[edit class-of-service]
user@host# edit rewrite-rules (dscp | inet-precedence) rewrite-name
user@host# edit forwarding-class class-name
user@host# set loss-priority low class-name code-points (alias / bits)
```

3. Configure behavior aggregate classifiers for DiffServ CoS.

```
[edit class-of-service]
user@host# edit classifiers (dscp | inet-precedence) classifier-name
user@host# edit forwarding-classes class-name
user@host# set loss-priority class-name code-points (alias / bits)
```

4. Configure expedited forwarding class classifiers.

```
[edit class-of-service classifiers]
user@host# edit forwarding-classes class-name
user@host# set loss-priority class-name code-points (alias / bits)
```

5. Define the forwarding-class mappings.

```
[edit class-of-service]
user@host# edit forwarding-classes class queue-number queue-number
```

6. Configure network control forwarding class classifiers.

```
[edit class-of-service]
user@host# edit forwarding-class class-name
user@host# set loss-priority low class-name code-points (alias / bits)
```

7. Apply the rewrite rules and classifiers to the interfaces.

```
[edit class-of-service interface interface-name unit unit-number]
user@host# set rewrite-rule (dscp | inet-precedence ) (rewrite-name| default)
user@host# set classifiers (dscp | inet-precedence ) classifier-name | default)
```

8. Set the global system default.

```
[edit ]
user@host# edit class-of-service system-defaults classifiers exp classifier-name
```

RELATED DOCUMENTATION

CoS on ACX Series Routers Features Overview | 680

Classifiers and Rewrite Rules at the Global, Physical, and Logical Interface Levels Overview

IN THIS SECTION

Depending on your platform, CoS supports classification and rewrite at the global, physical, and logical interface levels.

At a global level, depending on your platform, you can define EXP classification.

At a physical and logical interface level, depending on your platform, you can define the following features:

- DSCP, DSCP-IPV6, and IPv4 precedence classifiers
- DSCP, DSCP-IPV6, and IPv4 precedence rewrites
- EXP classifiers
- EXP rewrites
- IEEE 802.1p and IEEE 802.1ad classifiers (inner and outer)
- IEEE 802.1p and IEEE 802.1ad rewrites (outer)

The IEEE 802.1ad classifier uses IEEE 802.1p and DEI bits together.

NOTE: You cannot configure both IEEE 802.1p and IEEE 802.1ad classifiers together.

To configure global EXP classifiers, include the **classfiers exp** ***classifier-name*** statement at the **[edit class-of-service system-defaults]** hierarchy level.

To configure classifiers or *rewrite rules* at the physical interface, include either the **classifiers** statement or the **rewrite-rules** statement at the **[edit class-of-service interfaces** ***interface-name*****]** hierarchy level.

To configure fixed classifiers or rewrite rules at the logical interface, include the **forwarding-class** *fc* or the **rewrite-rules** statement at the **[edit class-of-service interfaces** ***interface-name*** **unit** ***number*****]** hierarchy level.

To configure EXP rewrite at the logical interface, include the **[edit class-of-service interfaces *interface-name* unit *number* rewrite-rules exp** *rewrite-rule*] statement.

To display classifiers configured under **system-defaults**, enter the **show class-of-service system-defaults** command.

To display classifiers and rewrite rules bound to physical interfaces, enter the **show class-of-service interfaces *interface-name*** command.

Platform-Specific Classifier Behavior

Use the following table to review platform-specific behaviors for your platforms.

Table 62: Platform-Specific Classifier Behavior

Platform	Difference
ACX6360	The ACX6360 router does not support L2 (IEEE802.1p and IEEE802.1ad) classifiers.
ACX7000 Series	• ACX7000 Series routers support classifiers of all types (Inet-Prec/DSCP/DSCP-v6/EXP/ IEEE-802.1p/IEEE-802.1ad) only at the logical interface level. • ACX7000 Series routers do not support global classifiers.

Platform-Specific Rewrite Rule Behavior

Use the following table to review platform-specific behaviors for your platforms.

Table 63: Platform-Specific Rewrite Rule Behavior

Platform	Difference
ACX6360	The ACX6360 router does not support rewrite rules.

Table 63: Platform-Specific Rewrite Rule Behavior *(Continued)*

Platform	Difference
ACX7000 Series	ACX7000 Series routers support rewrite rules of all types (Inet-Prec/DSCP/DSCP-v6/EXP/IEEE-802.1p/IEEE-802.1ad) only at the logical interface level.

RELATED DOCUMENTATION

Applying Classifiers and Rewrite Rules at the Global, Physical, and Logical Interface Levels

IN THIS SECTION

Depending on your platform, CoS supports classification and rewrite at the global, physical, and logical interface levels.

To apply a global EXP classifier, include the following statements at the **[edit class-of-service] system-defaults** hierarchy level.

```
[edit class-of-service]
user@device# set system-defaults classifiers exp classifier-name
```

On supported platforms, CoS supports one global system default classifier of the EXP type, as shown in the following example:

```
[edit class-of-service]
{
    system-defaults {
        classifiers {
            exp exp-classf-core;
        }
    }
}
```

To configure classifiers and rewrite rules at the physical interface level, include the following statements at the **[edit class-of-service] interfaces** hierarchy level.

```
[edit class-of-service]
interfaces {
    interface-name
        classifiers dscp classifier-name
        classifiers inet-precedence classifier-name
        classifiers ieee-802.1 [vlan-tag (outer | inner)] classifier-name
        rewrite-rules dscp rewrite-name
        rewrite-rules inet-prec  rewrite-name
        rewrite-rules ieee-802.1  rewrite-name
}
```

The following example shows classifiers and rewrite rules configured on physical and logical interfaces:

```
ge-0/1/0 {
    unit 0 {
        rewrite-rules {
            exp custom-exp;
        }
    }
    classifiers {
        dscp d1;
        ieee-802.1 ci;
    }
    rewrite-rules {
        dscp default;
    }
}
    ge-0/1/2 {
        classifiers {
            ieee-802.1 ci;
        }
        rewrite-rules {
            ieee-802.1 ri;
        }
    }
    ge-0/1/3 {
        unit 0 {
            rewrite-rules {
                exp custom-exp2;
            }
        }
    }
    ge-0/1/7 {
        classifiers {
            dscp d1;
        }
    }
    ge-0/1/8 {
        classifiers {
            dscp d1;
        }
    }
```

Rewrite Rules on ACX7000 Series Routers

ACX7000 Series routers do not support rewrite functionality on ethernet-bridge and ethernet-ccc encapsulation. For rewrite rules to work on an ACX7000 Series router, the router always needs tagged packets to be present, and the logical interface must be VLAN aware. For example, only packet forwarding will work with the following configuration:

```
[edit]
set interfaces et-0/0/6 encapsulation ethernet-ccc
set interfaces et-0/0/6 unit 0 family ccc
```

For both forwarding and rewrite to work on a logical interface, use the following configuration:

```
[edit]
set interfaces et-0/0/0 flexible-vlan-tagging
set interfaces et-0/0/0 encapsulation flexible-ethernet-services
set interfaces et-0/0/0 unit 0 encapsulation vlan-ccc
set interfaces et-0/0/0 unit 0 vlan-id-list [1-4094]
set interfaces et-0/0/0 unit 0 family ccc
```

RELATED DOCUMENTATION

Classifiers and Rewrite Rules at the Global, Physical, and Logical Interface Levels Overview | 689

Applying DSCP and DSCP IPv6 Classifiers on ACX Series Routers

For ACX Series routers, you cannot apply separate DSCP and DSCP IPv6 classifiers for IPv4 and IPv6 packets on a physical interface. Instead, classifier assignment works as follows:

- If you assign a DSCP classifier only, IPv4 and IPv6 packets are classified using the DSCP classifier.
- If you assign a DSCP IPv6 classifier only, IPv4 and IPv6 packets are classified using the DSCP IPv6 classifier.
- If you assign an IP precedence classifier only, IPv4 and IPv6 packets are classified using the IP precedence classifier. In this case, the lower three bits of the DSCP field are ignored because IPv4 precedence mapping requires the upper three bits only.
- You can assign either DSCP, DSCP IPv6, or IPv4 precedence classifier types on a physical interface.
- You can configure either DSCP, DSCP IPv6, or IPv4 precedence rewrite rules on a physical interface. The rewrite rule applies to both IPv4 and IPv6 packets.
- If you assign either the DSCP or the IP precedence classifier in conjunction with the DSCP IPv6 classifier, the commit fails.

RELATED DOCUMENTATION

Default DSCP and DSCP IPv6 Classifiers | 65

Schedulers Overview for ACX Series Routers

IN THIS SECTION

You use *schedulers* to define the properties of output queues. These properties include the amount of interface bandwidth assigned to the queue, the size of the memory buffer allocated for storing packets, the priority of the queue, and the random early detection (RED) drop profiles associated with the queue.

You associate the schedulers with forwarding classes by means of *scheduler maps*. You can then associate each scheduler map with an interface, thereby configuring the hardware queues, packet schedulers, and RED processes that operate according to this mapping.

In ACX Series routers, you can configure more than one strict-priority queue per port. The hardware services the queues in the descending order of queue numbers marked as strict priority. All the strict-priority queues are given preferential treatment by the scheduler as long as their shaping rates (or peak information rates) are not met. Unlike MX Series routers, the ACX Series routers configured with queues as *strict-high* at the `[edit class-of-service schedulers` *`scheduler-name`* `priority` *`strict-high`*`]` statement hierarchy, the service is based on queue number and not based on sharing the *strict-high* queues.

Scheduling on ACX Routers Running Junos OS

Unlike other ACX Series routers, ACX5048 and ACX5096 router supports CIR among strict-priority queues. There is no implicit queue number-based priority among the strict-priority queues. Unlike other ACX Series routers, ACX5048 and ACX5096 router supports configuring drop profiles for loss-priority `low`, `medium-high`, and `high` for non-TCP protocols as well.

NOTE: The options `buffer-partition multicast percent` *`<0-100>`* at the `[edit class-of-service schedulers` *`scheduler-name`* `buffer-size]` hierarchy level and `multicast` *`<0-100>`* at the `[edit class-of-service schedulers` *`scheduler-name`* `shared-buffer-maximum]` hierarchy level are supported only on ACX5048 and ACX5096 routers. For more information, see "Shared and Dedicated Buffer Memory Pools on ACX Series Routers" on page 699.

ACX5448 routers support port-based queueing, scheduling, and shaping. You can configure up to eight queues (virtual output queues) per physical interface (port). Scheduling properties can be applied at both physical as well as logical interface levels. The egress scheduler supports two priority levels (`strict-high` and `low`). Multiple strict-high priority queues and multiple low (default) priority queues can be configured.

By default a port on an ACX5448 router gets a dedicated buffer of 100 microseconds and shared buffer from DRAM. Delay buffer controls the latency of the queue during congestion and maximum number of packets that can be held in a queue. Default buffer size per port is 100 microseconds.

NOTE: On all ACX Series routers running Junos OS, the `strict` priority queues cannot have `transmit-rate` configured.

NOTE: On ACX710 routers, class-of-service commit changes, particularly for schedulers on a physical interface, cause very short (less than one second) pauses in forwarding packets. This can cause packet loss. For some protocols that use very short keepalive intervals, such as BFD, we recommend you disable or enlarge the keepalive interval before you apply scheduler changes. We also recommend you configure scheduling before you enable live traffic.

NOTE: On ACX 4000 routers, whenever the scheduling and shaping parameters of a port or any of its queues are changed, the entire scheduling configuration on the port is erased and the new configuration is applied. During this window, the traffic pattern does not adhere to user parameters. We recommend you configure scheduling before you enable live traffic.

Scheduling on ACX Routers Running Junos OS Evolved

All ACX7000 routers run Junos OS Evolved.

Scheduling Priorities

ACX7000 routers support eight scheduler priority levels for port-level scheduling and six levels for hierarchical scheduling. ACX7000 routers also support multiple queues with the same priority.

Table 64: Scheduling Priority Support

Priority	Port Scheduling	Hierarchical Scheduling
low-latency	Yes	Yes

Table 64: Scheduling Priority Support *(Continued)*

Priority	Port Scheduling	Hierarchical Scheduling
strict-hight	Yes	Yes
high	Yes	Yes
medium-high	Yes	Yes
medium-low	Yes	Yes
low-high	Yes	No
low-medium	Yes	No
low	Yes	Yes

NOTE: ACX routers *do not* guarantee round-robin distribution between same priority queues.

NOTE: On all ACX routers running Junos OS Evolved, you can only configure `transmit-rate` on `low` priority queues.

Low Latency Queuing (LLQ)

ACX7000 routers support low latency queuing (LLQ). LLQ enables delay-sensitive data to have preferential treatment over other traffic. A `low-latency` queue has the highest priority over any other priority queues, including `strict-high` queues, as well as a low delay scheduling profile.

For port scheduling of virtual output queues (VOQs), low latency VOQs receive their own dedicated egress queue. High priority VOQs receive a second dedicated egress queue, and low priority VOQs receive a third dedicated egress queue.

Due to the scheduling hierarchy of hierarchical class of service (HCoS), a hierarchical scheduling can use a maximum of two egress queues. Therefore for hierarchical scheduling of VOQs, low latency VOQs and high priority VOQs receive a common dedicated egress queue, and low priority VOQs receive the second dedicated egress queue.

NOTE: We recommend the following when configuring low-latency VOQs:

- Use policers to normalize the burstiness of traffic before it reaches a low-latency VOQ.
- Configure a maximum of two low-latency VOQs on a physical or logical interface.
- Classify and schedule traffic (that is, reserve bandwidth) for low-latency VOQs so that there is no congestion for those queues.

NOTE: Low-latency queues receive the same buffers as other queues to efficiently use the limited hardware VOQ buffer profiles.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
Junos OS Evolved 23.4R1	ACX7000 routers support six scheduler priority levels for port scheduling and hierarchical scheduling.
Junos OS Evolved 23.4R1	ACX7000 routers support low latency queuing (LLQ).
Junos OS Evolved 24.4R1	ACX7000 routers support eight priority levels for port scheduling.

RELATED DOCUMENTATION

RED Drop Profiles for Congestion Management | 462

Shared and Dedicated Buffer Memory Pools on ACX Series Routers | 699

Understand Virtual Output Queues | 790

Shared and Dedicated Buffer Memory Pools on ACX Series Routers

ACX Routers Running Junos OS

The ACX5048 and ACX5096 router has 12 megabytes of Packet Forwarding Engine (PFE) wide common packet buffer memory that is used to store packets on interface queues. The buffer memory is divided into two pools, shared buffers and dedicated buffers or reserved buffers.

Shared buffers are a global memory pool that the router allocates dynamically to ports as needed, so the buffers are shared among the ports. To configure a maximum amount of shared buffer that the multicast packets can consume, include the `multicast` *`percentage`* CLI statement at the `[edit class-of-service schedulers` *`scheduler-name`* `shared-buffer maximum]` hierarchy level. The value that you can specify for `multicast` *`percentage`* CLI command can be from 0 through 100 percent. If the `multicast` *`percentage`* CLI statement is not added, then the value defined by the `shared-buffer maximum percent` *`percentage`* is used for multicast packets as well.

Dedicated buffers or reserved buffers are a memory pool divided equally among the router ports. Each port receives a minimum guaranteed amount of buffer space, dedicated to each port, not shared among ports. To configure a dedicated buffer for multicast packets, include the `buffer-partition multicast` *`percentage`* CLI statement at the `[edit class-of-service schedulers scheduler-name buffer-size]` hierarchy level. The value that you can specify for `buffer-partition multicast` *`percentage`* CLI command can be from 0 through 100 percent. If the `buffer-partition multicast` *`percentage`* CLI statement is not configured, then a default value of 25% is reserved for multicast packets.

NOTE: The total amount of actual queue buffer is defined using the `buffer-size` CLI command.

The ACX5048 router supports delay bandwidth buffer (DBB) for virtual output queues (VOQs). ACX5048 router supports an external DRAM memory, as well as an on-chip buffer (OCB) for storing packet data. A packet is either fully stored in the DRAM or fully stored in the OCB and can consume one or more buffers (upto 40 buffers) depending on the packet size versus the buffer size. A buffer contains a single packet or a part of a single packet.

NOTE: The ACX5048 router does not support buffering for IRB multicast traffic and therefore CLIs for configuring multicast is not supported.

The ACX5048 router does not support `buffer-partition multicast percent` *percentage* option for `buffer-size` and `multicast` *`percentage`* option for `shared-buffer-maximum`.

To configure shared and dedicated buffers, include the `multicast` *percentage* and `buffer-partition multicast` *percentage* CLI statements at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
schedulers {
    scheduler-name {
        buffer-size (percent percentage | remainder | temporal microseconds | buffer-partition
multicast percent percentage                        );
        shared-buffer maximum (percent percentage | multicast percentage);
    }
}
```

The following is a sample configuration for shared and dedicated buffers in ACX5048 and ACX5096 routers:

```
[edit class-of-service]
schedulers schd1{
    buffer-size percent 80;
        buffer-partition {
            multicast {
                percent 30;
            }
        }
        shared-buffer {
            maximum {
                20;
                multicast {
                    10;
                }
            }
        }
    }
```

The port gets 50 microseconds worth of reserved buffer. For a 10 Gigabyte port without any shaper, this translates to 62500 Bytes.

In the above sample configuration, the total buffer size allotted for the queue is 80 percent.

Under buffer-partition, the multicast packets get 30 percent of the total buffer-size, which translates to about 24 percent of port-buffer. The unicast packets get the remaining 70 percent of 80 percent of the port buffer, which translates to 56 percent of port buffer.

Under shared-buffer, the multicast packets get up to 10 percent of the total shared buffer. Unicast packets use up to 20 percent of the total shared buffer.

The following is a sample configuration for shared and dedicated buffers in ACX5048 router:

```
[edit class-of-service]
schedulers schd1{
    buffer-size percent 80;
        shared-buffer {
            maximum {
                20;
            }
        }
    }
```

The ACX5048 router has OCB size of 16MB and DRAM size of 6GB. The default buffer size per port is 100 microseconds. The total buffer size for 48X10GE + 4X100GE comes to 11MB. The ACX5048 router supports deep buffering of oversubscribed traffic using external DRAM to queue traffic to oversubscribed ports. The ACX5048 router uses the DRAM-Mix mode by default, which uses DRAM buffers during oversubscription cases. The ACX5048 router supports configuring buffer size (dedicated buffers) per egress queue, which is similar to ACX5000 line of routers.

The ACX6360 routerhas a 39 MB total switch buffer pool. By default, 15 percent of the total buffer pool is allocated to the dedicated buffer pool and the remaining is allocated to the shared buffer pool. If you configure the shared buffer pool as less than 100 percent of the available buffer pool, the remaining buffer space is added to the dedicated buffer pool. You can distribute the shared buffer pool among lossless, lossy, and multicast queues with the following configuration:

```
[edit class-of-service shared-buffer]
user@router# set egress percent 100
user@router# set egress buffer-partition lossless percent percent-value
user@router# set egress buffer-partition lossy percent percent-value
user@router# set egress buffer-partition multicast percent percent-value
```

ACX Routers Running Junos OS Evolved

ACX7000 Series routers have On-chip buffers (OCB) and external DRAM buffers for VOQ Buffering. Table 65 on page 702 shows information OCB and external DRAM buffers for ACX7000 Series routers.

Table 65: OCB and External DRAM Buffers for ACX7000 Routers

Platform	Total OCB in MB	Total DRAM buffer in MB
ACX7024	8	2
ACX7024X	8	2
ACX7100-32C	32	8
ACX7100-48L	32	8
ACX7332	16	4
ACX7348	16	4
ACX7509	64	8

Shared buffers are a global memory pool that the router allocates dynamically to ports as needed, so the buffers are shared among the ports. Table 66 on page 702 shows the shared buffer per VOQ based on port speed for ACX7000 routers.

Table 66: Shared Buffers per VOQ for ACX7000 Routers

Interface Speed in Gbps	Shared Buffer per VOQ in MB	Shared Buffer per VOQ in ms
1	5	40
10	50	40
25	250	80
40	250	50
50	250	40

Table 66: Shared Buffers per VOQ for ACX7000 Routers *(Continued)*

Interface Speed in Gbps	Shared Buffer per VOQ in MB	Shared Buffer per VOQ in ms
100	500	40
200	500	20
400	500	10

Dedicated buffers are a memory pool divided equally among the router ports. Each VOQ of a port receives a minimum guaranteed amount of buffer space, dedicated to each VOQ, not shared among ports. Table 67 on page 703 shows the default dedicated buffer based on port speed for ACX7000 routers.

Table 67: Dedicated Buffer per VOQ for ACX7000 Routers

Interface Speed in Gbps	Dedicated Buffer per VOQ in KB
1	125
10	125
25	625
40	625
50	625
100	1250
200	2500
400	5000

To configure shared and dedicated buffers, use the CLI below. In ACX7000 Series routers, use the `temporal` option to configure the shared buffer and use the `percent | remainder` options to configure the dedicated buffer.

```
[edit class-of-service]
schedulers {
    scheduler-name {
        buffer-size (percent percentage | remainder | temporal microseconds);
    }
}
```

RELATED DOCUMENTATION

Schedulers Overview for ACX Series Routers | 695

CoS for PPP and MLPPP Interfaces on ACX Series Routers

IN THIS SECTION

Junos CoS enables you to divide traffic into classes and offer various levels of throughput and packet loss when congestion occurs. This functionality allows packet loss to happen according to rules that you configure. The Junos CoS features provide a set of mechanisms that you can use to provide differentiated services when best-effort traffic delivery is insufficient.

CoS functionalities are supported on PPP and MLPPP interfaces. Up to four forwarding classes and four queues are supported per logical interface for PPP and MLPPP packets.

Use Feature Explorer to confirm platform and release support for CoS for PPP and MLPPP interfaces.

The Junos OS CoS features provide a set of mechanisms that you can use to provide differentiated services when best-effort traffic delivery is insufficient. In designing CoS applications, you must give careful consideration to your service needs, and you must thoroughly plan and design your CoS configuration to ensure consistency across all routing devices in a CoS domain. You must also consider all the routing devices and other networking equipment in the CoS domain to ensure interoperability among all equipment.

Limitations That are Common for CoS on PPP and MLPPP Interfaces

The following restrictions apply for configuring CoS on PPP and MLPPP interfaces on ACX Series routers:

- For interfaces with PPP encapsulation, you can configure interfaces to support the IPv4, Internet Protocol Control Protocol (IPCP), PPP Challenge Handshake Authentication Protocol (CHAP), and Password Authentication Protocol (PAP) applications.

- Drop timeout, which defines a recovery method for any packets dropped by the member links in a link services or multilink bundle, is not applicable.

- Loss of traffic occurs during a change of CoS configuration; you cannot modify scheduling attributes instantaneously. The link moves to the down state for PPP, and the protocol is denoted as down for MLPPP interfaces.

- Scheduling and shaping capabilities are based on the CIR-EIR model and are not in accordance with the weighed fair queuing mode. The minimum transmit speed is 32 Kbps, and the minimum difference that can be supported between the transmit rate and shaping rate is also 32 Kbps.

- Buffer size is calculated in terms of packets using 256 bytes as the average packet size.

  Because there are no shared buffers, the usage of "buffer-size" and "buffer-size exact" attributes result in the same behavior.

- Only two loss priority levels, namely low and high, are supported. Traffic that arrives from the Packet Forwarding Engine with a medium-high priority is treated as high priority traffic. Although you can configure the medium-high loss priority type when you configure the action for a firewall term, it is considered by the system as high priority traffic.

- A fixed, in-built mapping between forwarding class and queue number as follows is performed: Best-effort is queue 0, expedited-forwarding is queue 1, assured-forwarding is queue 2, and network-control is queue 3

- For WRED configurations, the difference between maximum fill-level and minimum-fill level is a number raised to the power of 2 in terms of number of packets (x^2). Otherwise, the lower fill-level is tuned to turn the difference into a value raised to the power of 2. For example, queues contain a size of 64 packets. If the following configuration is performed:

```
 fill-level 50 drop-probability 0;
fill-level 100 drop-probability 100;
```

- For the lower fill level, the minimum number of packets is 32. However, if you specify the fill-level to be 45 instead of 50, the lower fill level is 28. Because 64 - 28, which equals 36 is not a power of 2, the lower fill-level is internally adjusted to convert it to be a number exponentially raised to 2.
- When fragmentation-map is configured, the forwarding-class carrying the multi-class 0 traffic must be assigned the highest priority and the forwarding-class carrying the multi-class 3 traffic must be assigned the lowest priority. Such a configuration is necessary because of the NPU design.

Limitations for CoS on PPP Interfaces

The following restrictions apply for configuring CoS on PPP interfaces:

- The distribution of excess rate between 2 or more queues that contain the same priority occurs on a first-come, first-served basis. For example, consider two Queues configured as follows:

  Q1 : Transmit-rate = 10%, Shaping-rate = 20%

  Q2 : Transmit-rate = 10%, Shaping-rate = 30% on same priority

  The excess rate for Q1 = (20 - 10) = 10%

  The excess rate for Q2 = (30 - 10) = 20%
- The excess rate distribution between Q1 and Q2 does not follow the same ratio but packets in these queues are served in first-come, first-served manner. The shaping rate continues to be honored in such a scenario.

Guidelines for Configuring CoS on PPP and MLPPP Interfaces

Keep the following points in mind when you configure CoS on PPP and MLPPP interfaces:

- You can configure only the any option with the `protocol` statement at the `[edit class-of-service schedulers` *`scheduler-name`* `drop-profile-map]` hierarchy level to specify the protocol type for the specified scheduler. You cannot specify the TCP or non-TCP protocol types.
- Weighted fair queuing (WFQ) shaping and scheduling model is not supported. Instead of WFQ, CIR-EIR model is supported to handle shaping and scheduling requirements.

- Percentage-based rate configuration is not supported for MLPPP LSQ interfaces; only absolute rate configration in bps is supported.
- All valid configurations specified for MLPPP interfaces with inet address families are also valid for MLPPP interfaces with MPLS address families. For example, EXP classifier as a global classifier is supported for ingress classification and EXP rewrite rule is supported for egress logical interfaces.
- A maximum of 1000 logical interfaces can be supported.
- A maximum of 280 PPP or MLPPP logical interfaces can support drop-profiles on a system. On each MIC, a maximum of 140 PPP or MLPPP interfaces are supported.

Limitations for CoS on MLPPP Interfaces

The following restrictions apply for configuring CoS on MLPPP interfaces:

- Percentage-based configuration for scheduling and shaping parameters is not supported; only absolute rate configuration is supported.
- A temporal value, in microseconds, is used to compute the buffer size. For the temporal setting, the queuing algorithm starts dropping packets when it queues more than a computed number of bytes. This maximum is computed by multiplying the transmission rate of the queue by the configured temporal value. The default queue size and percentage-based queue size are not based on the current bandwidth.
- If you configure a scheduler map without a fragmentation map, any scheduler-map configuration including the default settings are applied the same behavior as the exact transmission rate functionality. Priorities of traffic are not honored and no excess rates are provisioned. The forwarding class with no rate configuration receives the minimum fixed rate allocated to it, which is 32 Kbps.
- Support for oversubscription and priority is not available, which might cause inefficient bandwidth utilization.

Support for oversubscription between two multi-classes on the same priority is not provided. The queue corresponding to which there is no-multiclass entry is moved to the disabled state. Only one-to-one mapping between forwarding-class to multi-class is supported. One forwarding class can send traffic corresponding to only one multi-class.

CoS Functionalities for IPv4 Over PPP Interfaces

The following CoS capabilities are supported on PPP interfaces for IPv4 traffic:

- Ingress Classification can be either specified as fixed classifiers or behavior aggregate (BA) classifiers. Fixed classifiers map all traffic on an interface to the forwarding class. The forwarding class determines the output queue. To configure a fixed classifier, include the `forwarding-class class-name` statement at the `[edit class-of-service interface *interface-name* unit *logical-unit-number*]` hierarchy level.

- BA classification, or CoS value traffic classification, refers to a method of packet classification that uses a CoS configuration to set the forwarding class of a packet based on the CoS value in the IP packet header. The CoS value examined for BA classification purposes can be the Differentiated Services code point (DSCP) value, or the IP precedence value, or EXP bits. The default classifier is based on the IP precedence value.

- To configure the global EXP classifier, include the following statements at the **[edit class-of-service system-defaults]** hierarchy level.

```
[edit class-of-service]
{
    system-defaults
        {
        classifiers exp classifier-name
    }
}
```

  CoS supports one global system default classifier of the EXP type, as shown in the following example:

```
[edit class-of-service]
{
    system-defaults {
        classifiers {
            exp exp-classf-core;
        }
    }
}
```

- All packets that are received on a logical interface in the ingress direction can be classified to a single forwarding class using fixed classification.

- BA Classification based on the following packet fields is supported at the logical interface level:

  - IPv4 - inet-precedence

  - DSCP

The following is the configuration stanza for defining BA classifiers:

```
[edit class-of-service]
        classifiers {
            (inet-precedence|dscp ) classifier-name
            {
                forwarding-class class-name loss-priority [low | high] code-points
([ aliases ] | [ dscp-bits ]);
            }
```

- Queuing and scheduling functionalities comprise the following parameters:
  - Transmit rate per queue
  - Shaping rate per queue
  - WRED
  - Forwarding classes. A maximum of 4 forwarding classes can be defined for PPP interfaces.
- The four priorities supported for logical interface-level queuing are strict-high, medium-high, medium-low, or low. The transmit rate per queue (CIR) is the minimum committed rate can be specified for each queue. The shaping rate per queue (PIR) is the maximum transmit rate can be specified for each queue. For WRED, the default behavior is to enable tail drops. The drop profile configuration enables WRED and enables different drop behavior based on the drop precedence to be entered. Loss priority settings help determine which packets are dropped from the network during periods of congestion. The software supports multiple packet loss priority (PLP) designations: low and high
- Buffer-size can be specific in percentage and temporal configuration. This size is turned into the number of packets per queue, with 256 bytes treated as the average packet size. The following settings can be configured at the queue level:
  - Guaranteed transmit rate (CIR)
  - Shaping rate (PIR)
  - Drop profile
  - Buffer size

```

    [edit class-of-service]
        schedulers scheduler_name {
            transmit-rate (percent number | rate);
            shaping-rate (percent number | rate);
            buffer-size (percent | temporal)
            drop-profile-map map_name;
        }
```

Only 4 forwarding classes and only 4 queues per logical interface are supported. Also, logical interface-based shaping is not supported.

- Packet and byte counters for transmitted and dropped packets are available per queue. These statistical details are displayed using the show interfaces queue command. Aggregation to provide port-level statistics, if needed, is also supported by the system. The logical interface-level statistics are correctly available for egress direction and are displayed in the output, but the statistics pertaining to dropped packets are not considered because of hardware limitations. The following configuration stanza defines the rewrite rules:

```
        [edit class-of-service]
        rewrite-rules {
            (inet-precedence | dscp | exp) rewrite-name
            {
                forwarding-class class-name loss-priority [low | high] code-point value;
            }
        }
```

Each of the rewrite rules can be attached to an interface by using the following configuration syntax:

```
[edit class-of-service interfaces interface-name]
```

```
rewrite-rules {
    dscp (rewrite-name | default) protocol (inet-both | inet-outer | mpls);
    exp (rewrite-name | default) protocol protocol-types;
    inet-precedence (rewrite-name | default) protocol (inet-both | inet-outer | mpls);
}
```

- For rewrite rules, IPv4 or inet precedence and EXP rules are supported. EXP rewrite rules apply only to logical interfaces. You cannot apply EXP rewrite rules to physical interfaces. There are no default EXP rewrite rules. If you want to rewrite the EXP value in MPLS packets, you must configure EXP rewrite rules and apply them to logical interfaces. If no rewrite rules are applied, all MPLS labels that are pushed have a value of zero (0). The EXP value remains unchanged on MPLS labels that are swapped.

CoS Functionalities for IPv4 Over MLPPP Interfaces

The following CoS capabilities are supported on MLPPP interfaces for IPv4 traffic:

- Ingress Classification can either be specified as fixed or BA classifiers.
- Fixed classification using forwarding classes and BA classification using IPv4 precedence value are supported.
- The following scheduling and queuing properties are supported:
  - Transmit rate per queue
  - Shaping rate per queue
  - WRED
  - Forwarding classes
  - Buffer size per queue
- Forwarding class to multilink class mapping. You use the `multilink-class` statement to map a forwarding class into a multiclass MLPPP (MCML).
- All of the firewall features supported are applicable for MLPPP interfaces for IPv4 packets
- For rewrite rules, IPv4 precedence rule is supported.

The following CoS capabilities are supported on MLPPP interfaces for MPLS packets:

- Ingress Classification can either be specified as fixed or BA classifiers. Fixed classification using forwarding classes and BA classification using EXP bits are supported.
- The following scheduling and queuing properties are supported:
  - Transmit rate per queue
  - Shaping rate per queue
  - WRED
  - Forwarding classes
  - Buffer size per queue
- Forwarding class to multilink class mapping. You use the `multilink-class` statement to map a forwarding class into a multiclass MLPPP (MCML).
- All of the firewall features supported are applicable for MLPPP interfaces for IPv4 packets

- For rewrite rules, the EXP rule is supported.

The following example illustrates an MLPPP CoS configuration set:

```
[edit]
class-of-service {
        classifiers {
            inet-precedence all-traffic-inet {
                forwarding-class assured-forwarding {
                    loss-priority low code-points 101;
                }
                forwarding-class expedited-forwarding {
                    loss-priority low code-points 000;
                }
            }
        }
        drop-profiles {
            plp-low-red {
                fill-level 50 drop-probability 0;
                fill-level 100 drop-probability 100;
            }
            plp-high-red {
                fill-level 25 drop-probability 0;
                fill-level 50 drop-probability 100;
            }
        }
        forwarding-classes {
            queue 0 best-effort;
            queue 1 assured-forwarding;
            queue 2 expedited-forwarding;
            queue 3 network-control;
        }

        schedulers {
            evdo-mlppp-best-effort {
                transmit-rate 1M;
                buffer-size percent 80;
                priority medium-high;
            }
            evdo-mlppp-assured-forwarding {
                transmit-rate 500000;
                buffer-size percent 10;
                drop-profile-map loss-priority low protocol any drop-profile plp-low-red;
                drop-profile-map loss-priority high protocol any drop-profile plp-high-red;
                priority medium-low;
            }
            evdo-mlppp-expedited-forwarding {
                transmit-rate 300000;
                buffer-size percent 5;
                priority low;
            }
            evdo-mlppp-network-control {
                transmit-rate 200000;
                buffer-size percent 5;
                priority strict-high;
            }
        }

        scheduler-maps {
            evdo-mlppp-cos-map {
                forwarding-class best-effort scheduler evdo-mlppp-best-effort;
                forwarding-class assured-forwarding scheduler evdo-mlppp-assured-forwarding;
                forwarding-class network-control scheduler evdo-mlppp-network-control;
                forwarding-class expedited-forwarding scheduler evdo-mlppp-expedited-forwarding;
            }
        }
        fragmentation-maps {
            frag-mlppp {
                forwarding-class {
                    assured-forwarding {
                        multilink-class 2;
                    }
                    expedited-forwarding {
                        multilink-class 3;
                    }
                    best-effort {
                        multilink-class 1;
                    }
                    network-control {
                        multilink-class 0;
                    }
                }
            }
        }
```

```
        interfaces {
            lsq-0/* {
                    unit * {
                        scheduler-map evdo-mlppp-cos-map;
                        fragmentation-map frag-mlppp;
                    }
            }
        }
```

NOTE: In supported routers, the forwarding class and queue mapping is fixed for PPP and MLPPP interfaces.

The following example illustrates an MLPPP firewall configuration set:

```
[edit]
firewall {
            filter classify-evdo-traffic-mlppp {
                interface-specific;
                term signalling {
                    from {
                        dscp ef;
                    }
                then {
                    count signalling-counter;
            forwarding-class network-control;
            accept;
        }
            }
        term user-speech {
            from {
                    dscp af31;
                }
            then {
                policer user-speech-rate-limit;
                    count user-speech-counter;
                    forwarding-class network-control;
                    accept;
                }
            }
        term ptt-mcs {
            from {
                dscp af11;
            }
            then {
                count ptt-mcs-counter;
                forwarding-class network-control;
                accept;
            }
        }
        term user-video {
            from {
                dscp af21;
            }
            then {
                count user-video-counter;
                    loss-priority low;
                    forwarding-class assured-forwarding;
                    accept;
                }
            }
                term user-cmcs {
                    from {
                        dscp af12;
                    }
                then {
                    count user-cmcs-counter;
                    loss-priority low;
                    forwarding-class assured-forwarding;
                    accept;
                }
            }
            term ran-rlp-retransmission {
                from {
                    dscp af41;
                }
                then {
                    count rlp-retransmit-counter;
                    loss-priority high;
                    forwarding-class assured-forwarding;
                    accept;
                }
            }
            term user-best-effort {
                then {
                    count user-be-counter;
                forwarding-class best-effort;
                    accept;
                }
            }
        }
    }
```

CoS for NAT Services on ACX Series Routers

For packets that go through NAT services, the classification is done when the packet enters the router, and the resulting forwarding class and the packet loss priority are preserved when the packet goes through the inline services interface. When the packet exits the router through a physical interface, the queuing, scheduling, and rewrite rules configured on the egress physical interface are applied on the basis of the forwarding class and packet loss priority derived during ingress.

NOTE: CoS for NAT services is not applicable on ACX5048, ACX5096, or ACX7000 Series routers.

On *all* ACX Series routers, you cannot configure classification and rewrite policies on the inline services interface.

Queuing and scheduling are supported on the inline services interface to handle congestion at the inline services interface. Congestion at inline services interface is possible when the interface is oversubscribed. In this case, you need to selectively drop packets based on the packet's forwarding class and packet loss priority. To drop packets based on the packet's forwarding class and packet loss priority, scheduler map configurations are supported on the inline services interface.

RELATED DOCUMENTATION

Network Address Translation Overview on ACX Series

Network Address Port Translation Overview

Enabling Inline Services Interface on ACX Series

Understanding Service Sets

Service Filters in ACX Series

Guidelines for Applying Service Filters

Service Filter Match Conditions for IPv4 Traffic
Service Filter Match Conditions for IPv4 Traffic
Network Address Translation Address Overload in ACX Series
Configuring Address Pools for Network Address Port Translation (NAPT) Overview
Configuring Service Sets to Be Applied to Services Interfaces
Configuring Queuing and Scheduling on Inline Services Interface

Hierarchical Class of Service in ACX Series Routers

IN THIS SECTION

Scheduling properties can be applied at physical as well as logical interface and logical interface set levels. Service providers can support hierarchical class of service (HCoS) at multiple levels to meet the service level agreements and bandwidth allocations for subscribers.

Hierarchical Scheduling on the Physical Interface

By default, the queuing mode on all the physical interfaces in ACX routers that support HCoS is 8 queues per physical interface (port). In the hierarchical scheduler mode, you can configure up to 3 levels (physical interface, logical interface, and queues) or 4 levels (including logical interfaces sets) of scheduling, depending on the platform.

You can enable hierarchical scheduling by including the `hierarchical-scheduler` CLI command under the `[edit interfaces` *`interface-name`*`]` hierarchy:

```
[edit]
interfaces xe-0/0/1 {
    hierarchical-scheduler;
}
```

NOTE: If you change the physical interface queuing mode from default to hierarchical scheduler mode or vice-versa, the traffic flowing out of the physical interface during the mode change results in a transient loss of traffic data.

Traffic Control Profiles

Traffic control profiles hold parameters for levels above the queue level of the scheduler hierarchy. You configure the scheduling and shaping on the scheduler nodes using traffic control profiles and scheduler maps for the queue level. The traffic control profile defines the following characteristics of a scheduler node:

- Scheduler-map
- Shaping rate
- Guaranteed rate

You attach traffic control profiles at the physical interface, logical interface set, and logical interface level. You define scheduling and shaping characteristics for the scheduler node using `shaping-rate` and `guaranteed-rate`. The following is a sample traffic control profile configuration:

```
[edit class-of-service traffic-control-profiles]
tcp-500m-shaping-rate {
    shaping-rate 500m;
}
tcp-vlan0 {
    shaping-rate 200m;
    guaranteed-rate 100m;
    scheduler-map tcp-map-vlan0; # Applies scheduler maps to customer VLANs.
}
tcp-remaining {
    shaping-rate 500m;
    guaranteed-rate 200m;
    scheduler-map map-remaining
}
```

Schedulers

A scheduler defines scheduling and queuing characteristics of a queue and holds the information about the queues, the last level of the hierarchy. The following is a sample scheduler configuration:

```
[edit class-of-service schedulers]
sched-vlan0-q0 {
    priority low;
    transmit-rate 20m;
    buffer-size temporal 100ms;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}
sched-vlanl-q1 {
    priority strict-high;
    shaping-rate 20m;
}
```

Drop Profiles

Drop profiles allow you to specify queue specific behavior to drop packets based on WRED profile under congestion. The following is a sample drop profile configuration:

```
[edit class-of-service drop-profiles]
dp-low {
    fill-level 80 drop-probability 0;
    fill-level 100 drop-probability 100;
}
dp-high {
    fill-level 60 drop-probability 0;
    fill-level 80 drop-probability 100;
}
```

NOTE: ACX routers support only two discrete fill levels per drop profile. The drop probability of the first fill level must be 0. ACX routers do not support interpolated drop profiles.

Scheduler Maps

A scheduler map is referenced by traffic control profiles to define queues. The scheduler map establishes the number of queues over a scheduler node, associating a forwarding-class with a scheduler. The following is a sample scheduler map configuration:

```
[edit class-of-service scheduler-maps]
tcp-map-vlan0 {
    forwarding-class voice scheduler sched-vlan0-q0;
    forwarding-class video scheduler sched-vlan0-q1;
    forwarding-class data scheduler sched-vlan0-q2;
}
tcp-map-vlan1 {
    forwarding-class voice scheduler sched-vlan1-q0;
    forwarding-class video scheduler sched-vlan1-q1;
    forwarding-class data scheduler sched-vlan1-q2;
}
```

Applying the Traffic Control Profiles

You can attach `output-traffic-control-profile` and `output-traffic-control-profile-remaining` at various levels of the scheduler hierarchy to achieve hierarchical class of service. On an interface set, `output-traffic-control-profile-remaining` is active if there is at least one member logical interface with an attached `output-traffic-control-profile` and at least one without.

NOTE: Although a shaping rate can be applied directly to the physical interface, hierarchical schedulers must use a traffic control profile to hold this parameter.

All logical interfaces that are part of an interface set inherit the traffic control profile of the interface set.

The following is a sample configuration to apply traffic control profiles:

```
[edit class-of-service interfaces]
et-1/0/0 {
    output-traffic-control-profile tcp-500m-shaping-rate;
    output-traffic-control-profile-remaining  tcp-500m-shaping-rate;
    unit 0 {
        output-traffic-control-profile tcp-cvlan0;
    }
    unit 1 {
        output-traffic-control-profile tcp-cvlan1;
    }
}
interface-set svlan0 {
    output-traffic-control-profile tcp-svlan0;
}
interface-set svlan1 {
    output-traffic-control-profile tcp-svlan1;
}
```

NOTE: You can apply traffic control profiles to physical, logical, and aggregated Ethernet interfaces.

HCoS on Aggregated Ethernet Interfaces

Configuring HCoS on aggregated Ethernet (AE) interfaces is exactly the same as configuring HCoS on physical interfaces. Apply all HCoS configuration to the AE interface.

NOTE: You cannot apply HCoS configuration to a physical interface that is a member of an AE interface. To apply HCoS to an AE interface, do not configure HCoS on a physical interface and then add the physical interface to the AE interface. First configure the AE interface, then configure HCoS on the AE interface.

HCoS configuration on an AE interface is automatically applied to each member interface. How the system applies the configuration to each member interfaces (that is, whether the configuration is normalized or not) depends on the AE interface configuration mode:

- Link protection mode — system applies configuration directly to each member interface *without* normalization.

- Non-link protection replica mode — system applies configuration directly to each member interface *without* normalization.
- Non-link protection scale mode (default mode) — system *does* normalize the configuraiton before applying to each member interface.

HCoS configuration has the following requirements when the configuration must be normalized across member interfaces:

- `transmit-rate` at scheduler/queue level — both percent and absolute configuration allowed
- `shaping-rate` at scheduler/queue level — both percent and absolute configuration allowed
- `guaranteed-rate` at traffic control profile level — only absolute configuration allowed
- `shaping-rate` at traffic control profile level — only absolute configuration allowed
- `shaping-rate` at port level — only absolute configuration allowed

If the configuration must be normalized, rates configured as a percent require no normalization. If the configuration must be normalized, rates configured as absolute must be normalized. You can calculated the normalized value for each interface by dividing the absolute configured value by the number of member interfaces in the AE interface.

Subscriber Services

Supported ACX routers support hierarchical class of service functionality for subscriber services such as Layer 3 VPN, Layer 2 VPN, Ethernet pseudowire (VPWS), and VPLS for logical interface instance on the AC (Access Port).

NOTE: Hierarchical class of service is not supported for Layer 2 bridging (bridge domain VLAN) service.

The following sections explain the hierarchical class of service configuration for subscriber services:

Configuring hierarchical class of service for Layer 3 VPN Service

Supported ACX routers can be configured to provide Layer 3 VPN services to subscribers by connecting the UNI port to a CE device. The physical port can be configured to provide Layer 3 VPN services to multiple subscribers. You can schedule traffic for different Layer 3 VPN instances based on the SLA parameters agreed with the subscriber.

The following is a sample UNI and NNI logical interface configuration on the PE router providing the Layer 3 VPN service:

```
[edit interfaces]
xe-0/0/1 {
    description “NNI IFL”;
    unit 0 {
        family inet {
            address 100.1.1.1/24;
        }
        family mpls;
    }
}
ge-0/0/1 {
    description “UNI IFL”;
    hierarchical-scheduler;
    flexible-vlan-tagging;
    unit 0 {
        vlan-id 100;
        family inet {
            address 20.20.0.1/24;
        }
    }
    unit 1 {
        vlan-id 101;
        family inet {
            address 20.20.1.1/24;
        }
    }
    unit 2 {
        vlan-id 2;
        family inet {
            address 20.20.2.1/24;
        }
    }
    unit 3 {
        vlan-id 3;
        family inet {
            address 20.20.3.1/24;
        }
    }
    unit 4 {
        vlan-id 4;
        family inet {
            address 20.20.4.1/24;
        }
    }
    ...
}
```

Scheduling can be enabled on the interfaces to achieve hierarchical class of service support for traffic flowing from NNI towards UNI direction.

Configuring hierarchical class of service for Layer 2 VPN (Ethernet Pseudowires) Service

Supported ACX routers can be configured to provide Layer 2 VPN services to subscribers based on Ethernet pseudowires where the UNI port is connected to a CE device. The physical port can be configured to provide Layer 2 VPN services to multiple subscribers. You can schedule traffic for different pseudowires based on the SLA parameters agreed with the subscriber. Hierarchical class of service can be enabled per UNI logical interface represented as the attachment point of the Ethernet pseudowire to achieve the functionality.

The following is a sample to configure the UNI logical interface on the PE router providing the Layer 2 VPN service based on Ethernet pseudowire:

```
[edit interfaces]
ge-0/0/1 {
    hierarchical-scheduler;
    vlan-tagging;
    unit 0 {
        encapsulation vlan-ccc;
        vlan-id 0;
    }
    unit 1 {
        encapsulation vlan-ccc;
        vlan-id 1;
    }
    unit 2 {
        encapsulation vlan-ccc;
        vlan-id 2;
    }
    unit 3 {
        encapsulation vlan-ccc;
        vlan-id 3;
    }
    unit 4 {
        encapsulation vlan-ccc;
        vlan-id 4;
    }
}
```

You can enable scheduling on the interfaces to achieve hierarchical class of service for traffic flowing from NNI towards UNI direction.

Configuring hierarchical class of service for VPLS Service

Supported ACX routers can be configured to provide Layer 2 VPN services to subscribers based on VPLS where the UNI port can be connected to a CE device. Subscriber network is attached to UNI logical interface at the PE router and have a VPLS instance. The same physical port can service multiple VPLS instances for various subscribers. The service provider can schedule traffic for different VPLS instances based on the SLA parameters agreed with the subscriber. You can enable hierarchical class of service per UNI logical interface representing the VPLS instance for the subscriber to achieve the functionality.

The following is a sample to configure the UNI logical interface on the PE router providing the VPLS service:

```
[edit interfaces]
ge-0/0/1 {
    hierarchical-scheduler;
    vlan-tagging;
    encapsulation vlan-vpls;
    unit 0 {
        encapsulation vlan-vpls;
        vlan-id 0;
    }
    unit 1 {
        encapsulation vlan-vpls;
        vlan-id 1;
    }
    unit 2 {
        encapsulation vlan-vpls;
        vlan-id 2;
    }
    unit 3 {
        encapsulation vlan-vpls;
        vlan-id 3;
    }
    unit 4 {
        encapsulation vlan-vpls;
        vlan-id 4;
    }
}
```

Scheduling can be enabled on the interfaces to achieve hierarchical class of service for the traffic flowing from NNI towards UNI direction.

Verifying the hierarchical class of service configurations

You can use the following CLI commands to verify the configuration:

- `show interfaces queue`—Shows physical interface aggregate, physical interface remaining, and logical interface traffic statistics to monitor the traffic received and transmitted. The following are some sample outputs of `show interfaces queue` CLI command:

```
user@host# run show interfaces queue ge-0/0/4.2
  Logical interface ge-0/0/4.2 (Index 555) (SNMP ifIndex 671)
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Burst size: 0
Queue: 0, Forwarding classes: 8q0
  Queued:
    Packets              :               1121476                  7642 pps
    Bytes                :             587885024              32237416 bps
  Transmitted:
    Packets              :                594964                  3160 pps
    Bytes                :             304621568              12946280 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                     0                     0 pps
    RL-dropped bytes     :                     0                     0 bps
    Total-dropped packets:                526512                  4482 pps
    Total-dropped bytes  :             283263456              19291136 bps
  Queue Buffer Usage:
    Reserved buffer      :                0 pkts                     0 bytes
    Shared buffer        :                0 pkts                     0 bytes
Queue: 1, Forwarding classes: 8q1
  Queued:
    Packets              :               1121476                  7642 pps
    Bytes                :           587885154             32237416 bps
  Transmitted:
    Packets              :              594959                 3160 pps
    Bytes                :           304619008             12946280 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                   0                    0 pps
    RL-dropped bytes     :                   0                    0 bps
    Total-dropped packets:              526517                 4482 pps
    Total-dropped bytes  :           283266146             19291136 bps
  Queue Buffer Usage:
    Reserved buffer      :              0 pkts                    0 bytes
    Shared buffer        :              0 pkts                    0 bytes
Queue: 2, Forwarding classes: 8q2
  Queued:
    Packets              :             1119456                 7321 pps
    Bytes                :           595127702             31122568 bps
  Transmitted:
    Packets              :              274601                 1500 pps
    Bytes                :           140595712              6144000 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                   0                    0 pps
    RL-dropped bytes     :                   0                    0 bps
    Total-dropped packets:              844855                 5821 pps
    Total-dropped bytes  :           454531990             24978568 bps
  Queue Buffer Usage:
    Reserved buffer      :              0 pkts                    0 bytes
    Shared buffer        :              0 pkts                    0 bytes
Queue: 3, Forwarding classes: 8q3
  Queued:
    Packets              :             1119464                 7303 pps
    Bytes                :           595131980             31122568 bps
  Transmitted:
    Packets              :              274602                 1500 pps
    Bytes                :           140596224              6144000 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                   0                    0 pps
    RL-dropped bytes     :                   0                    0 bps
    Total-dropped packets:              844862                 5803 pps
    Total-dropped bytes  :           454535756             24978568 bps
  Queue Buffer Usage:
    Reserved buffer      :              0 pkts                    0 bytes
    Shared buffer        :              0 pkts                    0 bytes
Queue: 4, Forwarding classes: 8q4
```

```
  Queued:
    Packets              :               1121476                  7642 pps
    Bytes                :             587885024              32237416 bps
  Transmitted:
    Packets              :                594964                  3160 pps
    Bytes                :             304621568              12946280 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                     0                     0 pps
    RL-dropped bytes     :                     0                     0 bps
    Total-dropped packets:                526512                  4482 pps
    Total-dropped bytes  :             283263456              19291136 bps
  Queue Buffer Usage:
    Reserved buffer      :                0 pkts                     0 bytes
    Shared buffer        :                0 pkts                     0 bytes
Queue: 5, Forwarding classes: 8q5
  Queued:
    Packets              :               1121476                  7660 pps
    Bytes                :             587885024              32310560 bps
  Transmitted:
    Packets              :                594964                  3178 pps
    Bytes                :             304621568              13019424 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                     0                     0 pps
    RL-dropped bytes     :                     0                     0 bps
    Total-dropped packets:                526512                  4482 pps
    Total-dropped bytes  :             283263456              19291136 bps
  Queue Buffer Usage:
    Reserved buffer      :                0 pkts                     0 bytes
    Shared buffer        :                0 pkts                     0 bytes
Queue: 6, Forwarding classes: 8q6
  Queued:
    Packets              :               1121476                  7017 pps
    Bytes                :             587190842              29535136 bps
  Transmitted:
    Packets              :                621684                  3589 pps
    Bytes                :             318302208              14701712 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                     0                     0 pps
    RL-dropped bytes     :                     0                     0 bps
    Total-dropped packets:                499792                  3428 pps
    Total-dropped bytes  :             268888634              14833424 bps
  Queue Buffer Usage:
    Reserved buffer      :                0 pkts                     0 bytes
    Shared buffer        :                0 pkts                      0 bytes
Queue: 7, Forwarding classes: 8q7
  Queued:
    Packets              :          1121477                    6481 pps
    Bytes                :        586036910                27137704 bps
  Transmitted:
    Packets              :           666066                    3660 pps
    Bytes                :        341025792                14994280 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                0                       0 pps
    RL-dropped bytes     :                0                       0 bps
    Total-dropped packets:           455411                    2821 pps
    Total-dropped bytes  :        245011118                12143424 bps
  Queue Buffer Usage:
    Reserved buffer      :                0 pkts                      0 bytes
    Shared buffer        :                0 pkts                      0 bytes
```

```
user@host# run show interfaces queue ge-0/0/4 aggregate
Physical interface: ge-0/0/4, Enabled, Physical link is Up
  Interface index: 648, SNMP ifIndex: 1763
  Description:  UNI side - connected to ixia 6/9
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Queue: 0, Forwarding classes: 8q0
  Queued:
    Packets              :         16762731                   33205 pps
    Bytes                :       8738362264               139058280 bps
  Transmitted:
    Packets              :          8779233                   13724 pps
    Bytes                :       4494967296                56220880 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                0                       0 pps
    RL-dropped bytes     :                0                       0 bps
    Total-dropped packets:          7983498                   19481 pps
    Total-dropped bytes  :       4243394968                82837400 bps
  Queue Buffer Usage:
    Reserved buffer      :               32 pkts                   6656 bytes
    Shared buffer        :               52 pkts                  10816 bytes
Queue: 1, Forwarding classes: 8q1
  Queued:
    Packets              :         16762732                   33237 pps
```

```
      Bytes               :           8738359966             139190408 bps
    Transmitted:
      Packets             :              8779363                 13756 pps
      Bytes               :           4495033856              56353008 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                   0                     0 pps
      RL-dropped bytes     :                   0                     0 bps
      Total-dropped packets:             7983369                 19481 pps
      Total-dropped bytes  :          4243326110              82837400 bps
    Queue Buffer Usage:
      Reserved buffer      :                 32 pkts                6656 bytes
      Shared buffer        :                 43 pkts                8944 bytes
  Queue: 2, Forwarding classes: 8q2
    Queued:
      Packets             :             16754769                 30221 pps
      Bytes               :           8826383526             127522040 bps
    Transmitted:
      Packets             :              4052168                  6369 pps
      Bytes               :           2074710016              26095472 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                   0                     0 pps
      RL-dropped bytes     :                   0                     0 bps
      Total-dropped packets:            12702601                 23852 pps
      Total-dropped bytes  :          6751673510             101426568 bps
    Queue Buffer Usage:
      Reserved buffer      :                 32 pkts                6656 bytes
      Shared buffer        :              24232 pkts             5040256 bytes
  Queue: 3, Forwarding classes: 8q3
    Queued:
      Packets             :             16754937                 30328 pps
      Bytes               :           8826406908             127965968 bps
    Transmitted:
      Packets             :              4052173                  6378 pps
      Bytes               :           2074646336              26134456 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                   0                     0 pps
      RL-dropped bytes     :                   0                     0 bps
      Total-dropped packets:            12702764                 23950 pps
      Total-dropped bytes  :          6751760572             101831512 bps
    Queue Buffer Usage:
      Reserved buffer      :                 32 pkts                6656 bytes
      Shared buffer        :              24205 pkts             5034640 bytes
  Queue: 4, Forwarding classes: 8q4
```

```
    Queued:
      Packets              :             16762735                 33406 pps
      Bytes                :           8738360722             139886136 bps
    Transmitted:
      Packets              :              8779404                 13828 pps
      Bytes                :           4495054848              56648672 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                    0                     0 pps
      RL-dropped bytes     :                    0                     0 bps
      Total-dropped packets:              7983331                 19578 pps
      Total-dropped bytes  :           4243305874              83237464 bps
    Queue Buffer Usage:
      Reserved buffer      :              32 pkts                  6656 bytes
      Shared buffer        :              46 pkts                  9568 bytes
  Queue: 5, Forwarding classes: 8q5
    Queued:
      Packets              :             16762734                 33285 pps
      Bytes                :           8738359924             139389112 bps
    Transmitted:
      Packets              :              8779416                 13788 pps
      Bytes                :           4495060992              56485136 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                    0                     0 pps
      RL-dropped bytes     :                    0                     0 bps
      Total-dropped packets:              7983318                 19497 pps
      Total-dropped bytes  :           4243298932              82903976 bps
    Queue Buffer Usage:
      Reserved buffer      :              32 pkts                  6656 bytes
      Shared buffer        :              52 pkts                 10816 bytes
  Queue: 6, Forwarding classes: 8q6
    Queued:
      Packets              :             16762732                 27688 pps
      Bytes                :           8721917186             115931832 bps
    Transmitted:
      Packets              :              9618678                 12111 pps
      Bytes                :           4924763136              49614432 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                    0                     0 pps
      RL-dropped bytes     :                    0                     0 bps
      Total-dropped packets:              7144054                 15577 pps
      Total-dropped bytes  :           3797154050              66317400 bps
    Queue Buffer Usage:
      Reserved buffer      :              24 pkts                  4992 bytes
      Shared buffer        :                0 pkts                        0 bytes
  Queue: 7, Forwarding classes: 8q7
    Queued:
      Packets              :             16762733                     26045 pps
      Bytes                :           8710804790                 108947832 bps
    Transmitted:
      Packets              :             10187546                     11805 pps
      Bytes                :           5216023552                  48359208 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                    0                         0 pps
      RL-dropped bytes     :                    0                         0 bps
      Total-dropped packets:              6575187                     14240 pps
      Total-dropped bytes  :           3494781238                  60588624 bps
    Queue Buffer Usage:
      Reserved buffer      :               21 pkts                     4368 bytes
      Shared buffer        :                0 pkts                        0 bytes
```

```
user@host# run show interfaces queue ge-0/0/4 remaining-traffic
Physical interface: ge-0/0/4, Enabled, Physical link is Up
  Interface index: 648, SNMP ifIndex: 1763
  Description:  UNI side - connected to ixia 6/9
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Queue: 0, Forwarding classes: 8q0
  Queued:
    Packets              :                77501                      6106 pps
    Bytes                :             41609646                  26235344 bps
  Transmitted:
    Packets              :                 3206                       243 pps
    Bytes                :              1641472                    999248 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                    0                         0 pps
    RL-dropped bytes     :                    0                         0 bps
    Total-dropped packets:                74295                      5863 pps
    Total-dropped bytes  :             39968174                  25236096 bps
  Queue Buffer Usage:
    Reserved buffer      :                8 pkts                     1664 bytes
    Shared buffer        :             1495 pkts                   310960 bytes
Queue: 1, Forwarding classes: 8q1
  Queued:
    Packets              :                77489                      6100 pps
    Bytes                :             41444318              26107008 bps
  Transmitted:
    Packets              :                 9330                   732 pps
    Bytes                :              4776960               3002344 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                    0                     0 pps
    RL-dropped bytes     :                    0                     0 bps
    Total-dropped packets:                68159                  5368 pps
    Total-dropped bytes  :             36667358              23104664 bps
  Queue Buffer Usage:
    Reserved buffer      :               8 pkts                  1664 bytes
    Shared buffer        :            1435 pkts                298480 bytes
Queue: 2, Forwarding classes: 8q2
  Queued:
    Packets              :                77485                  6099 pps
    Bytes                :             41362188              26054080 bps
  Transmitted:
    Packets              :                12417                   975 pps
    Bytes                :              6357504               3996992 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                    0                     0 pps
    RL-dropped bytes     :                    0                     0 bps
    Total-dropped packets:                65068                  5124 pps
    Total-dropped bytes  :             35004684              22057088 bps
  Queue Buffer Usage:
    Reserved buffer      :               8 pkts                  1664 bytes
    Shared buffer        :            1507 pkts                313456 bytes
Queue: 3, Forwarding classes: 8q3
  Queued:
    Packets              :                77547                  6114 pps
    Bytes                :             41609314              26271632 bps
  Transmitted:
    Packets              :                 3261                   243 pps
    Bytes                :              1646862                999248 bps
    Tail-dropped packets : Not Available
    RL-dropped packets   :                    0                     0 pps
    RL-dropped bytes     :                    0                     0 bps
    Total-dropped packets:                74286                  5871 pps
    Total-dropped bytes  :             39962452              25272384 bps
  Queue Buffer Usage:
    Reserved buffer      :               8 pkts                  1664 bytes
    Shared buffer        :            1381 pkts                287248 bytes
Queue: 4, Forwarding classes: 8q4
```

```
    Queued:
      Packets              :                 77502                  6105 pps
      Bytes                :              41450894              26131200 bps
    Transmitted:
      Packets              :                  9349                   732 pps
      Bytes                :               4786688               3002344 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                     0                     0 pps
      RL-dropped bytes     :                     0                     0 bps
      Total-dropped packets:                 68153                  5373 pps
      Total-dropped bytes  :              36664206              23128856 bps
    Queue Buffer Usage:
      Reserved buffer      :                8 pkts                  1664 bytes
      Shared buffer        :             1366 pkts                284128 bytes
  Queue: 5, Forwarding classes: 8q5
    Queued:
      Packets              :                 77480                  6094 pps
      Bytes                :              41358904              26032304 bps
    Transmitted:
      Packets              :                 12444                   975 pps
      Bytes                :               6371328               3996992 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                     0                     0 pps
      RL-dropped bytes     :                     0                     0 bps
      Total-dropped packets:                 65036                  5119 pps
      Total-dropped bytes  :              34987576              22035312 bps
    Queue Buffer Usage:
      Reserved buffer      :                8 pkts                  1664 bytes
      Shared buffer        :             1552 pkts                322816 bytes
  Queue: 6, Forwarding classes: 8q6
    Queued:
      Packets              :                 77970                  6099 pps
      Bytes                :              41151002              25749384 bps
    Transmitted:
      Packets              :                 30585                  2440 pps
      Bytes                :              15659520               9997088 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                     0                     0 pps
      RL-dropped bytes     :                     0                     0 bps
      Total-dropped packets:                 47385                  3659 pps
      Total-dropped bytes  :              25491482              15752296 bps
    Queue Buffer Usage:
      Reserved buffer      :                3 pkts                   624 bytes
    Shared buffer        :                 0 pkts                     0 bytes
  Queue: 7, Forwarding classes: 8q7
    Queued:
      Packets              :                 77971                    6099 pps
      Bytes                :              41151540                25749384 bps
    Transmitted:
      Packets              :                 30585                    2440 pps
      Bytes                :              15659520                 9997088 bps
      Tail-dropped packets : Not Available
      RL-dropped packets   :                     0                       0 pps
      RL-dropped bytes     :                     0                       0 bps
      Total-dropped packets:                 47386                    3659 pps
      Total-dropped bytes  :              25492020                15752296 bps
    Queue Buffer Usage:
      Reserved buffer      :                 3 pkts                   624 bytes
      Shared buffer        :                 0 pkts                     0 bytes
```

- `show class-of-service packet-buffer usage`—Shows the total buffer usage of the system. The following is a sample output of the `show class-of-service packet-buffer usage` CLI command:

```
user@host# run show class-of-service packet-buffer usage
            Egress:
            Total Buffer Bytes     :  10652.89 KB  in use out of 12480.00 KB
            Total Buffer Pkts      :  52445        in use out of 61440
            Dedicated Buffer Bytes :  48.14 KB     in use out of 738.16 KB
            Dedicated Buffer Pkts  :  237          in use out of 3634
            Shared Buffer Bytes    :  10604.75 KB  in use out of 11741.84 KB
            Shared Buffer Pkts     :  52208        in use out of 57806
```

- For interface sets, you can run the following commands:
  - `show interfaces interface-set voq`—Show VOQ statistics for the interface-set.
  - `show interfaces interface-set detail`—Display detailed output for the interface set.
  - `show interfaces interface-set voq remaining-traffic`—Show remaining traffic queue statistics for the interface set.

You can use the syslog to view the log messages and error reports.

Platform-Specific HCoS Behavior

Use the following table to review platform-specific behaviors for your platforms.

Table 68: Platform-Specific HCoS Behavior

Platform	Difference
ACX7000 Series Routers	• ACX7000 Series routers support 3 levels of hierarchical scheduling - physical interface, logical interface, and queues. • ACX7000 Series routers do not support the `show class-of-service packet-buffer usage` command.

Understanding SRv6 Classification and Rewrite

IN THIS SECTION

In Segment Routing over IPv6 (SRv6) networks, classification and rewrite is crucial for managing complex networking scenarios. You can classify and rewrite SRv6 traffic at encapsulation, transit, and decapsulation nodes to enforce CoS marking consistency for L3VPN and EVPN traffic. Configure policies to map VLAN PCP and DSCP/DSCP-IPv6 code points and to rewrite VLAN tags and IPv6 headers. Policies support reduced and non-reduced modes, SID depths, micro or classic SID, dynamic tunnels, PSP or USP, and TILFA. You can enable DSCP propagation to copy access-side DSCP into the SRv6 IP header. DSCP propagation disables DSCP, DSCP-IPv6, and inet-precedence rewrite rules. You cannot classify traffic based on inner payload code points at decapsulation.

Benefits of SRv6 Classification and Rewrite

- Enhances network performance by ensuring accurate handling of VLAN PCP bits and DSCP values, which is critical for maintaining QoS in complex networking scenarios such as L3VPN and EVPN.
- Facilitates consistent DSCP value propagation across the network, ensuring uniform treatment of packets and maintaining service quality.
- Supports a range of classification scenarios, allowing for precise traffic management and optimization tailored to specific network configurations and requirements.
- Improves network adaptability and precision in SRv6 environments, offering advanced solutions to meet the demands of dynamic and scalable network infrastructures.

Overview

SRv6 classification and rewrite feature provides a comprehensive framework for managing packet attributes across SRv6 nodes, including encapsulation, transit, and decapsulation points. By implementing this feature, you can classify packets based on VLAN PCP bits and DSCP values. This classification is essential for maintaining optimal QoS in complex networking scenarios like L3VPN and EVPN, where precise traffic management ensures that network performance is not compromised. The encapsulation process uses these attributes to define how packets should be treated as they enter the SRv6 network.

A core component of this functionality is the ability to propagate DSCP values from native IP packets to the SRv6 header. This ensures DSCP settings are consistently applied across the network, maintaining uniform treatment of packets. However, enabling DSCP propagation affects certain rewrite rules, which you must consider during configuration to avoid unexpected behavior. By understanding these interactions, you can configure SRv6 nodes to apply classification and rewriting rules effectively, optimizing traffic management according to your specific network requirements.

The feature outlines various classification scenarios that help you configure the network more efficiently. These scenarios specify how different SRv6 nodes handle the VLAN PCP and DSCP bits, guiding you in setting up the system to achieve desired traffic outcomes. While the feature supports many classification and rewrite scenarios, it does not support classification based on inner payload code points at decapsulation nodes. This limitation requires careful planning to ensure configurations are error-free and network operations are reliable.

Classification and Rewrite at SRv6 Encapsulation Node

Classification and rewrite at an SRv6 encapsulation node support the following scenarios:

Table 69: Classification and Rewrite at SRv6 Encapsulation Node

Scenario	Classification	Rewrite
Reduced mode	• Classification based on vlan pcp bits on access side interface • Classification based on dscp-ipv6 bits on access side interface • Classification based on dscp bits on access side interface	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag on core facing interface • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header on core facing interface
Non-reduced mode	• Classification based on vlan pcp bits on access side interface • Classification based on dscp-ipv6 bits on access side interface • Classification based on dscp bits on access side interface	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag on core facing interface • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header on core facing interface
Segment Identifier more than 4	• Classification based on vlan pcp bits on access side interface • Classification based on dscp-ipv6 bits on access side interface • Classification based on dscp bits on access side interface	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag on core facing interface • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header on core facing interface
Segment Identifier less than 4	• Classification based on vlan pcp bits on access side interface • Classification based on dscp-ipv6 bits on access side interface • Classification based on dscp bits on access side interface	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag on core facing interface • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header on core facing interface

Table 69: Classification and Rewrite at SRv6 Encapsulation Node *(Continued)*

Scenario	Classification	Rewrite
Best effort	• Classification based on vlan pcp bits on access side interface • Classification based on dscp-ipv6 bits on access side interface • Classification based on dscp bits on access side interface	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag on core facing interface • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header on core facing interface
Dynamic tunnel	• Classification based on vlan pcp bits on access side interface • Classification based on dscp-ipv6 bits on access side interface • Classification based on dscp bits on access side interface	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag on core facing interface • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header on core facing interface
L3VPN over SRv6	• Classification based on vlan pcp bits on access side interface • Classification based on dscp-ipv6 bits on access side interface • Classification based on dscp bits on access side interface	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag on core facing interface • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header on core facing interface • Preserve dscp/dscp-ipv6 bits in original packet (payload)
EVPN over SRv6 (ELAN, VPWS)	• Classification based on vlan pcp bits on access side interface • Classification based on dscp-ipv6 bits on access side interface • Classification based on dscp bits on access side interface	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag on core facing interface • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header on core facing interface • Preserve dscp/dscp-ipv6 bits in original packet (payload)

Classification and Rewrite at SRv6 Transit Node

Classification and rewrite at an SRv6 transit node support the following scenarios:

Table 70: Classification and Rewrite at SRv6 Transit Node

Scenario	Classification	Rewrite
Packet with one or more segment identifier (SID)	• Classification based on vlan pcp bits in SRv6 packet outer vlan tag • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header
Penultimate Segment Pop (PSP) when segment identifiers are less than 5	• Classification based on vlan pcp bits in SRv6 packet outer vlan tag • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header
Penultimate Segment Pop (PSP) when segment identifiers are more than 5	• Classification based on vlan pcp bits in SRv6 packet outer vlan tag • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header
Ultimate Segment Pop (USP)	• Classification based on vlan pcp bits in SRv6 packet outer vlan tag • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header
Reduced mode	• Classification based on vlan pcp bits in SRv6 packet outer vlan tag • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header
Binding SID	• Classification based on vlan pcp bits in SRv6 packet outer vlan tag • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header

Table 70: Classification and Rewrite at SRv6 Transit Node *(Continued)*

Scenario	Classification	Rewrite
TILFA non-insert mode	• Classification based on vlan pcp bits in SRv6 packet outer vlan tag • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header
TILFA insert mode	• Classification based on vlan pcp bits in SRv6 packet outer vlan tag • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in SRv6 packet outer vlan tag • Rewrite of dscp-ipv6 bits in SRv6 IPv6 header

Classification and Rewrite at SRv6 Decapsulation Node

Classification and rewrite at an SRv6 decapsulation node support the following scenarios:

Table 71: Classification and Rewrite at SRv6 Decapsulation Node

Scenario	Classification	Rewrite
Decapsulation with segment routing header	• Classification based on vlan pcp bits in SRv6 packet • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in original packet for EVPN-SRv6. Preserve payload DSCP. • Rewrite dscp-ipv6 bits in original packet in L3VPN-SRv6. • In EVPN-SRv6, preserve vlan pcp bits in original packet if vlan pcp rewrite is not configured on outgoing interface.

Table 71: Classification and Rewrite at SRv6 Decapsulation Node *(Continued)*

Scenario	Classification	Rewrite
Decapsulation without segment routing header	• Classification based on vlan pcp bits in SRv6 packet • Classification based on dscp-ipv6 bits in SRv6 IPv6 header	• Rewrite of vlan pcp bits in original packet for EVPN-SRv6. Preserve payload DSCP. • Rewrite dscp-ipv6 bits in original packet in L3VPN-SRv6. • In EVPN-SRv6, preserve vlan pcp bits in original packet if vlan pcp rewrite is not configured on outgoing interface. • In L3VPN-SRv6, preserve dscp-ipv6 bits in original packet if dscp-ipv6 rewrite not configured on outgoing interface
L3VPN over SRv6	• Classification based on vlan pcp bits in SRv6 packet on core interface • Classification based on dscp-ipv6 bits in SRv6 IPv6 header on core interface	• Rewrite dscp-ipv6 bits in original packet on access interface • In L3VPN-SRv6, Preserve dscp-ipv6 bits in original packet if dscp-ipv6 rewrite not configured on access interface
EVPN over SRv6 (ELAN, VPWS)	• Classification based on vlan pcp bits in SRv6 packet on core interface • Classification based on dscp-ipv6 bits in SRv6 IPv6 header on core interface	• Rewrite of vlan pcp bits in original packet on access interface • In EVPN-SRv6, Preserve vlan pcp bits in original packet if vlan pcp rewrite not configured on access interface

CoS Support for Pseudowire Subscriber Logical Tunnel Interfaces

SUMMARY

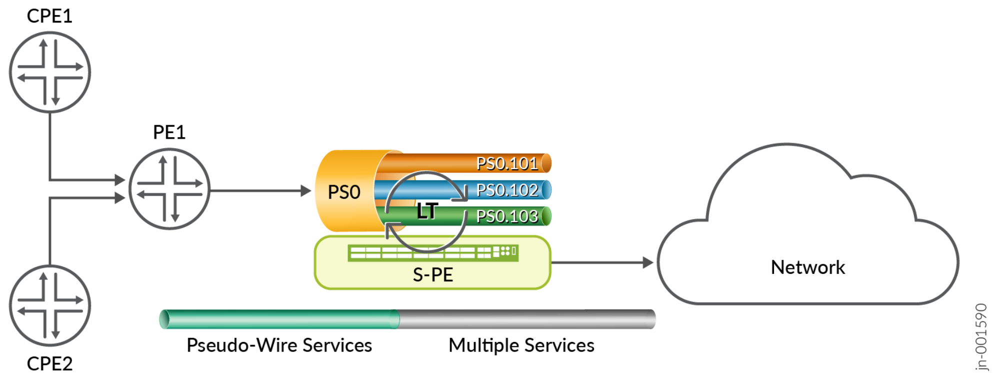
Junos provides CoS support for pseudowire subscriber logical tunnel (PSLT) interfaces, which is essential for optimizing traffic management within MPLS-based environments. Implementing CoS on pseudowire interfaces allows for efficient traffic prioritization and bandwidth allocation, especially in Layer 2 VPN or Layer 2 circuit services. This feature offers detailed traffic management through classification and rewriting functionalities, supporting traffic control profiles (TCPs) and hierarchical CoS configurations on logical tunnel interfaces.

IN THIS SECTION

In modern telecommunication networks, particularly those leveraging MPLS technology, PSLT interfaces play a crucial role in streamlining network operations and enhancing service delivery. Pseudo-wire subscriber (PS) interfaces are used to provide pseudo-wire connectivity for the subscribers/clients on access network to various services on the provider network. We can consider PS interfaces as a tunnel that is either an MPLS-based Layer 2 VPN or Layer 2 circuit. The PS interface is anchored over a logical tunnel (LT) interface. The LT interface is a recycle channel interface used to connect logical systems or VPN instances within the same router. A PS tunnel transports Ethernet encapsulated traffic from an access node to the router that hosts the subscriber management services. The termination of the PS tunnel on the router is like a physical Ethernet termination and is the point at which subscriber management functions are performed.

Figure 53 on page 744 illustrates how PSLT interfaces enable service providers to extend a pseudo-wire service from the access-aggregation network to the service edge. The stitching PE (S-PE) router terminates the pseudo wire services from access-aggregation network and initiates new services such as pseudo wire, L3-VPN etc. MPLS traffic from PE1 terminates at the PS interface PSn.0 logical interface of the S-PE router. After forwarding decision is made, packets are forwarded to designated static service logical interface (PSn.1, PSn.2, etc).

Figure 53: PSLT interfaces overview



In Figure 53 on page 744 above, upstream traffic, that is traffic from PE1 towards the core network, after recycling at LT interface, ingresses on service interfaces PS0.101, PS0.102, and PS0.103. You apply CoS classification to these service interfaces. Similarly in the downstream direction, that is traffic from the network to PE1, traffic egresses out over these same service interfaces, and here you can apply CoS rewrite. You can also apply traffic control profiles to these service interfaces.

Benefits of CoS Support for Pseudowire Subscriber Interfaces

- Enhances traffic management by enabling precise CoS classification and rewriting, optimizing traffic prioritization and efficient bandwidth allocation.
- Supports hierarchical CoS configurations, providing granular control over traffic flows and resource distribution across different service levels.
- Integrates traffic control profiles (TCP), facilitating advanced traffic shaping and control to meet specific service quality requirements.
- Increases network flexibility by allowing detailed management of data flow through logical tunnel interfaces, improving service delivery in complex network environments.

Understand CoS for PSLT Interfaces

The following are characteristics of CoS on PS interfaces:

- PS interfaces such as PS0, PS1, etc. are logical interface but have no representation in the CoS scheduling hierarchy. No separate bandwidth is allocated for a PS interface.
- Transport logical interfaces and client logical interfaces are part of the same LT interface, so both upstream and downstream traffic consume LT interface bandwidth.

- You can combine all the client logical interfaces of a PS interface in a logical interface set to have a common CoS scheduling configuration.
- Typical deployment has all client logical interfaces of a PS interface as part of a logical interface set with a TCP. Transport logical interfaces can have separate TCPs.
- If you apply no custom classifier or rewrite rule to a PS interface, then the default classifier and rewrite rule apply.
- An anchor point interface (physical interface) is specified for a PS interface. This identifies the logical tunnel interface that terminates the pseudo-wire tunnel at the access node.

Configuration of CoS on PS interfaces is similar to CoS on physical interfaces. However, behavioral differences due exist due to the fact that a PS interface is constituted on top of an LT recycle interface.

Table 72: CoS requirements for PS interfaces

CoS Feature	Target Interface	Direction
Classifier	PS0.0 transport logical interface	Downstream (network to PE router). From a classifier point of view, the transport logical interface is just any other logical interface.
	PS0.x – static client VLAN logical interface	Upstream (PE router to network). Client VLAN logical interfaces are just any other logical interface, so similar classifier functionality applies.
	LT.x	Point to point.
Rewrite rule	PS0.0 transport logical interface	Upstream. From a rewrite rule point of view, the transport logical interface is just any other logical interface.
	PS0.x – static client VLAN logical interface	Downstream.
	LT.x	Point to point.
Scheduler	PS0.0 transport logical interface	Upstream. You can attach a scheduler to shape the upstream traffic.
	PS0.x – static client VLAN logical interface	Downstream.

Table 72: CoS requirements for PS interfaces *(Continued)*

CoS Feature	Target Interface	Direction
	PS0.x-y – static client VLAN logical interface set	Downstream. You can attach a scheduler to shape downstream traffic for logical interfaces, PS0.x to PS0.y.
	LT interface	You can apply a scheduler to the LT interface. Be aware that LT port bandwidth depends on the recycle bandwidth allocation.

Configure CoS for PSLT Interfaces

You can enforce QoS on pseudowire subscriber interfaces to shape, prioritize, and mark traffic for L2/L3 VPN services by leveraging logical tunnel-anchored pseudowire client logical interfaces. Apply classifiers and rewrite rules per logical interface, and use hierarchical schedulers and traffic-control profiles on pseudowire logical interfaces or logical interface sets. You can use six priorities and weighted random early detection (WRED). Traffic rates are determined by the recycle bandwidth; use the logical tunnel port shaper to control upstream traffic.

The following shows sample CoS configuration over a PS interface:

1. Create an LT interface with a port speed of 100G:

```
[edit]
user@host# set chassis fpc 0 pfe 0 core 0 channel 0 tunnel-services bandwidth 100g
```

2. Put the LT interface into hierarchical scheduler mode:

```
[edit]
user@host# set groups ps_lt interfaces lt-0/0/0:0 hierarchical-scheduler
```

3. Create PS interfaces and logical interfaces:

```
[edit]
user@host# set interfaces ps0 anchor-point lt-0/0/0:0
user@host# set interfaces ps0 flexible-vlan-tagging
user@host# set interfaces ps0 unit 0 encapsulation ethernet-ccc
user@host# set interfaces ps0 unit 1 vlan-id 100
user@host# set interfaces ps0 unit 1 family inet address 70.70.1.1/24
```

```
user@host# set interfaces ps1 anchor-point lt-0/0/0:0
user@host# set interfaces ps1 flexible-vlan-tagging
user@host# set interfaces ps1 unit 0 encapsulation ethernet-ccc
user@host# set interfaces ps1 unit 1 vlan-id 100
user@host# set interfaces ps1 unit 1 family mpls
user@host# set interfaces ps1 unit 2 vlan-id 101
user@host# set interfaces ps1 unit 2 family inet address 24.1.1.1/24
```

4. Create a PS logical interface set of client logical interfaces:

```
[edit]
user@host# set interfaces interface-set psiflset1 interface ps1 unit 1
user@host# set interfaces interface-set psiflset1 interface ps1 unit 2
```

5. Apply a classifier to a client PS logical interface:

```
[edit]
user@host# set class-of-service interfaces ps0 unit 1 classifiers dscp dscp_cl
```

6. Apply a rewrite rule to a client PS logical interface:

```
[edit]
user@host# set class-of-service interfaces ps0 unit 1 rewrite-rules dscp rw_dscp
```

7. Apply a TCP to a client PS logical interface:

```
[edit]
user@host# set class-of-service interfaces ps1 unit 1 output-traffic-control-profile tcp_1
```

8. Apply a TCP to a PS logical interface set:

```
[edit]
user@host# set class-of-service interfaces interface-set psiflset1 output-traffic-control-
profile tcp_1
```

9. Use the `show interfaces voq` command to display statistics for PS logical interfaces:

```
user@host> show interfaces voq ps1.1
```

CHAPTER 12

Configuring Class of Service on MX Series 5G Universal Routing Platforms

IN THIS CHAPTER

Junos CoS on MX Series 5G Universal Routing Platforms Overview

The increased demand for sophisticated, media-rich services, the exponential growth of mobile sessions, and the emerging trend of cloud computing require a networking infrastructure that supports massive numbers of subscribers, service types and instances, and bandwidth. A number of features and methods have been developed to address these advanced network requirements, including Junos *class of service* (CoS). Junos CoS is a set of mechanisms that helps maintain specified service levels for your network by optimizing and prioritizing network traffic so that demand for resources can meet requirements. Use CoS mechanisms to control the allocation of network attributes such as available bandwidth, latency, *jitter*, packet drop, and bit rate errors so that resources are managed to levels acceptable to your network customers and applications.

CoS on Juniper Networks MX Series 5G Universal Routing Platforms

MX Series routers are available in a variety of configurations with robust features, including options that provide the level and granularity of the CoS support needed in your network. The MX Series hardware options include several models of Modular Port Concentrators (MPCs), using several different Modular Interface Cards (MICs), and several models of Dense Port Concentrators (DPCs). The MPCs and DPCs provide varying degrees of CoS support.

The MPCs are next-generation line modules for advanced Ethernet services edge and broadband edge networks using high capacity, modular Gigabit Ethernet, 10-Gigabit Ethernet, and 100-Gigabit Ethernet hardware. The MPCs house Packet Forwarding Engines that deliver comprehensive Layer 3 routing (IPv4 and IPv6), Layer 2 switching, inline services, and advanced hierarchical class of service (H-CoS) per MX Series slot. The MPCs can also take advantage of the high performance Junos Trio chipset.

Key CoS features provided by the MPCs include extensive queue management, scheduler hierarchy, shaping, intelligent oversubscription, weighted round robin (WRR), random early detection (RED), and weighted random early detection (WRED).

The DPCs (DPCE-X, DPCE-R, and DPCE-Q) each provide multiple physical interfaces and Packet Forwarding Engines on a single board that performs packet processing and forwarding. Each Packet Forwarding Engine consists of one I-chip for Layer 3 processing and one network processor for Layer 2. DPCE-Qs offer enhanced queuing capabilities and the CoS features of WRR, RED, and WRED.

RELATED DOCUMENTATION

CoS Features and Limitations on MX Series Routers | **750**

CoS Features and Limitations on MX Series Routers

IN THIS SECTION

All Layer 3 Junos OS CoS functions are supported on the MX Series routers. In addition, Layer 3 CoS capabilities, with the exception of traffic shaping, are supported on virtual LANs (VLANs) that span multiple ports.

MX Series routers can be equipped with Flexible PIC Concentrators (FPCs) and associated Physical Interface Cards (PICs), Dense Port Concentrators (DPCs), Modular Interface Cards (MICs), Modular Port Concentrators (MPCs), or MPCs with associated MICs. In all cases, the command-line interface (CLI) configuration syntax refers to FPCs, PICs, and ports (*type-fpc/pic/port*).

Fixed classification places all packets in the same forwarding class, or the usual multifield or behavior aggregate (BA) classifications can be used to treat packets differently. BA classification with firewall filters can be used for classification based on IP precedence, DSCP, IEEE, or other bits in the frame or packet header.

However, the MX Series routers can also employ multiple BA classifiers on the same *logical interface*. The logical interfaces do not have to employ the same type of BA classifier. For example, a single logical interface can use classifiers based on IP precedence as well as IEEE 802.1p. If the CoS bits of interest are on the inner VLAN tag of a dual-tagged VLAN interface, the classifier can examine either the inner or outer bits. (By default, the classification is done based on the outer VLAN tag.)

Internal fabric scheduling is based on only two queues: high and low priority. Strict-high priority queuing is also supported in the high-priority category.

Egress port scheduling supports up to eight queues per port using a form of round-robin queue servicing. The supported priority levels are strict-high, high, medium-high, medium-low, and low. The MX Series router architecture supports both early discard and tail drop on the queues.

All CoS features are supported at line rate.

MX Series routers allow fixed classification of traffic. All packets on a logical interface can be put into the same forwarding class. For example:

```
[edit class-of-service interfaces ge-1/0/0 unit 0]
user@host#set forwarding-class af
```

MX Series routers allow BA classification, the classifying of packets into different forwarding classes (up to eight) based on a value in the packet header. However, MX Series routers allow a mixture of BA classifiers (IEEE 802.1p and others) for logical interfaces on the same port. In the following example, the IEEE classifier is applied to Layer 2 traffic and the Internet precedence classifier is applied to Layer 3 (IP) traffic.

```
[edit class-of-service interfaces ge-0/0/0 unit 0]
user@host#set classifiers ieee-802.1 DOT1P-BA-1
user@host#set classifiers inet-precedence IPPRCE-BA-1
```

The IEEE classifier can also perform BA classification based on the bits of either the inner, outer, or transparent VLAN tag on a dual-tagged logical interface, as shown in the following example:

```
[edit class-of-service interfaces ge-0/0/0]
user@host#set unit 0 classifiers ieee-802.1 DOT1-BA-1 vlan-tag inner
user@host#set unit 1 classifiers ieee-802.1 DOT1-BA-1 vlan-tag outer
user@host#set unit 2 classifiers ieee-802.1 DOT1-BA-1 vlan-tag transparent
```

NOTE: The example above does not apply to single-tagged packets. The following example shows how to configure the classifier on single-tagged interfaces:

```
[edit class-of-service interfaces ge-0/0/0]
user@host#set unit 0 classifiers ieee-802.1 DOT1-BA-1
```

The default action is based on the outer VLAN tag's IEEE precedence bits.

The BA classification can be overridden with a multifield classifier in the action part of a firewall filter.

MX Series routers support classifications and rewrites for aggregated Ethernet (ae-) logical interfaces. MX Series routers also support the use of egress firewall filters for DSCP rewrites for IPv4 and IPv6 packets. For example:

```
[edit firewall family inet]
user@host# set term 1 from destination-address 198.51.100.100/32
user@host# set term 1 then dscp af21
user@host#  set term 2 then accept
```

On MX Series routers, the 64 classifier limit is a theoretical upper limit. In practice, you can configure 63 classifiers. Three values are used internally by the default IP precedence, IPv6, and EXP classifiers. Two other classifiers are used for forwarding class and queue operations. This leaves 58 classifiers for configuration purposes. If you configure Differentiated Services code point (DSCP) rewrites for MPLS, the maximum number of classifiers you can configure is less than 58.

On MX Series routers, IEEE 802.1 classifier bit rewrites are determined by forwarding class and packet priority, not by queue number and packet priority as on other routers.

The following scaling and performance parameters apply to MX Series routers:

- 48 classifiers of each type, when subscriber management is enabled
- 32 rewrite tables of each type, when subscriber management is enabled
- Eight queues per port
- 64 WRED profiles
- 100-ms queue buffering for interfaces 1 Gbps and above; 500 ms for all others
- Line-rate CoS features

For more information about MX Series router CoS capabilities, including software configuration, see "Configuring Hierarchical Schedulers for CoS" on page 446 and "Enhanced Queuing DPC CoS Properties" on page 916.

For Juniper Networks MX Series 5G Universal Routing Platforms, the following restrictions apply:

- You can only use multifield classifiers (but *not* BA classifiers) for IPv4 DSCP bits for virtual private LAN service (VPLS).
- You cannot use BA classifiers for IPv4 DSCP bits for Layer 2 VPNs.
- You cannot use BA classifiers for IPv6 DSCP bits for VPLS.
- You cannot use BA classifiers for IPv6 DSCP bits for Layer 2 VPNs.

On MX Series routers, you can apply classifiers or *rewrite rules* to an integrated bridging and routing (IRB) interface at the `[edit class-of-service interfaces irb unit logical-unit-number]` level of the hierarchy. All types of classifiers and rewrite rules are allowed. These classifiers and rewrite rules are independent of others configured on an MX Series router.

```
[edit class-of-service interfaces]
irb {
    unit logical-unit-number {
        classifiers {
            type (classifier-name | default) family (mpls | all);
        }
        rewrite-rules {
            dscp (rewrite-name | default);
            dscp-ipv6 (rewrite-name | default);
            exp (rewrite-name | default)protocol protocol-types;
            ieee-802.1 (rewrite-name | default) vlan-tag (outer | outer-and-inner);
            inet-precedence (rewrite-name | default);
        }
    }
}
```

The IRB classifiers and rewrite rules are applied only to the “routed” packets. For logical interfaces that are part of a bridge domain, only IEEE classifiers and IEEE rewrite rules are allowed. Only the listed options are available for rewrite rules on an IRB.

For dual-tagged bridge domain logical interfaces, you can configure classification based on the inner or outer VLAN tag’s IEEE 802.1p bits using the `vlan-tag` statement with the `inner` or `outer` option:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
classifiers {
    ieee-802.1 (classifier-name | default) vlan-tag (inner | outer);
}
```

Also, for dual-tagged bridge domain logical interfaces, you can configure rewrite rules to rewrite the outer or both outer and inner VLAN tag’s IEEE 802.1p bits using the `vlan-tag` statement with the `outer` or `outer-and-inner` option:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
rewrite-rules {
    ieee-802.1 (rewrite-rule-name | default) vlan-tag (outer | outer-and-inner);
}
```

Platform-Specific Behavior

Use the following table to review platform-specific details for your platforms.

Table 73: Platform-Specific Behavior

Platform	Difference
MX80	• The MX80 router is a single-board router with a built-in Routing Engine and one Packet Forwarding Engine, which can have up to four MICs attached to it. The Packet Forwarding Engine has two “pseudo” Flexible PIC Concentrators (FPC 0 and FPC1). Because there is no switching fabric, the single Packet Forwarding Engine takes care of both ingress and egress packet forwarding.

Configuring and Applying IEEE 802.1ad Classifiers

If you apply an IEEE 802.1 classifier to a logical interface, this classifier takes precedence and is not compatible with any other classifier type. You can set the forwarding class and loss priority for traffic on the basis of the three IEEE 802.1p bits (three bits in either the inner virtual LAN (VLAN) tag or the outer VLAN tag) and the drop eligible indicator (DEI) bit. You can apply the default map or customize one or more of the default values.

You then apply the classifier to the interface on which you configure IEEE 802.1ad frame formats.

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

1. Define the custom IEEE 802.1ad map:

    a. Create the classifier by specifying a name for it and defining it as an IEEE-802.1ad (DEI) classifier.

    ```
    [edit]
    user@host# edit class-of-service classifiers ieee-802.1ad dot1p_dei_class
    ```

    b. Assign the forwarding class and loss priority to the code-point alias.

    ```
    [edit class-of-service classifiers ieee-802.1ad dot1p_dei_class]
    user@host# set forwarding-class best-effort loss-priority low code-points [0000 1101]
    ```

2. Apply the classifier to the logical interface:

    a. Specify the interface to which you want to apply the classifier.

    ```
    [edit]
    user@host# edit class-of-service interfaces ge-2/0/0 unit 0
    ```

    b. Specify the name of the classifier you want to apply to the interface.

    ```
    [edit class-of-service interfaces ge-2/0/0 unit 0]
    user@host# set classifiers ieee-802.1ad dot1p_dei_class
    ```

3. Verify the custom IEEE 802.1ad map configuration:

```
[edit]
user@host# show
```

```
class-of-service {
    classifiers {
        ieee-802.1ad dot1p_dei_class {
            forwarding-class best-effort {
                loss-priority low code-points [ 0000 1101 ];
            }
        }
    }
}
```

```
class-of-service {
    interfaces {
        ge-2/0/0 {
            unit 0 {
                classifiers {
                    ieee-802.1ad dot1p_dei_class;
                }
            }
        }
    }
}
```

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 60

Apply Behavior Aggregate Classifiers to Interfaces | 83

Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels

IN THIS SECTION

Understanding Scheduling and Shaping of Traffic Routed to GRE Tunnels

You can manage CoS scheduling and shaping of traffic routed to generic route encapsulation (GRE) tunnel interfaces.

A single egress logical interface can be converted to multiple GRE tunnel interfaces. A GRE tunnel physical interface can support many logical interfaces, but one or more of those logical interfaces might not have an output traffic control profiles attached. If a GRE tunnel logical interface is not attached to an output traffic control profile, the router does not assign the interface a dedicated scheduler. Instead, the interface uses a reserved scheduler intended for all *unshaped tunnel traffic* (traffic entering a GRE tunnel logical interface that does not have an explicit traffic control profile configuration).

Configuration Overview

At GRE tunnel interfaces, the `output-traffic-control-profile` configuration statement can apply an output traffic scheduling and shaping profile at the physical or logical interface level, while the `output-traffic-control-profile-remaining` configuration statement can apply an output traffic scheduling and shaping profile for remaining traffic at the physical interface level only. Interface sets (sets of interfaces used to configure hierarchical CoS schedulers on supported Ethernet interfaces) are not supported on GRE tunnel interfaces.

By default—if you do not attach an output traffic control profile to the GRE tunnel physical interface—traffic entering the interface is scheduled and shaped using the default 95/5 scheduler with parameters as specified in the `tunnel-services` configuration.

If you use an output traffic control profile to configure the shaping rate at the GRE tunnel physical interface, the `shaping-rate` specified by the attached traffic control profile overrides the `bandwidth` specified as the tunnel services default value.

Configuration Caveats

When configuring hierarchical CoS scheduling and shaping of traffic routed to GRE tunnels, keep the following guidelines in mind:

- You must first configure and commit a hierarchical scheduler on the GRE tunnel physical interface, specifying a maximum of two hierarchical scheduling levels for node scaling. After you commit the `hierarchical-scheduler` configuration, you can configure scheduling and queuing parameters at the GRE tunnel physical or logical interfaces.
- GRE tunnel interfaces support eight egress queues.
- Queuing and scheduling calculations include Layer 3 fields. For GRE interfaces, Layer 3 fields include the delivery header (the outer IP header), the 4-byte GRE header, and the payload protocol header and data.

RELATED DOCUMENTATION

Example: Performing Output Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels

IN THIS SECTION

This example shows how to configure a generic routing encapsulation (GRE) tunnel device to perform CoS output scheduling and shaping of IPv4 traffic routed to GRE tunnels.

Requirements

This example uses the following Juniper Networks hardware and Junos OS software:

- Transport network—An IPv4 network running a supported Junos OS release.
- GRE tunnel device—One MX router installed as an ingress provider edge (PE) router.
- Input and output logical interfaces configurable on two ports of the MIC:
  - Input logical interface `ge-1/1/0.0` for receiving traffic that is to be transported across the network.
  - Output logical interfaces `ge-1/1/1.0`, `ge-1/1/1.1`, and `ge-1/1/1.2` to convert to GRE tunnel source interfaces `gr-1/1/10.1`, `gr-1/1/10.2`, and `gr-1/1/10.3`.

Overview

In this example, you configure the router with input and output logical interfaces for IPv4 traffic, and then you convert the output logical interface to four GRE tunnel source interfaces. You also install static routes in the routing table so that input traffic is routed to the four GRE tunnels.

NOTE: Before you apply a traffic control profile with a scheduler-map and shaping rate to a GRE tunnel interface, you must configure and commit a hierarchical scheduler on the GRE tunnel physical interface, specifying a maximum of two hierarchical scheduling levels for node scaling.

Configuration

IN THIS SECTION

To configure scheduling and shaping in hierarchical CoS queues for traffic routed to GRE tunnel interfaces, perform these tasks:

CLI Quick Configuration

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

Configuring Interfaces, Hierarchical Scheduling on the GRE Tunnel Physical Interface, and Static Routes

```
set chassis fpc 1 pic 1 tunnel-services bandwidth 1g
set interfaces ge-1/1/0 unit 0 family inet address 10.6.6.1/24
set interfaces ge-1/1/1 unit 0 family inet address 10.70.1.1/24 arp 10.70.1.3 mac
00:00:03:00:04:00
set interfaces ge-1/1/1 unit 0 family inet address 10.80.1.1/24 arp 10.80.1.3 mac
00:00:03:00:04:01
set interfaces ge-1/1/1 unit 0 family inet address 10.90.1.1/24 arp 10.90.1.3 mac
00:00:03:00:04:02
set interfaces ge-1/1/1 unit 0 family inet address 10.100.1.1/24 arp 10.100.1.3 mac
00:00:03:00:04:04
set interfaces gr-1/1/10 unit 1 family inet address 10.100.1.1/24
set interfaces gr-1/1/10 unit 1 tunnel source 10.70.1.1 destination 10.70.1.3
set interfaces gr-1/1/10 unit 2 family inet address 10.200.1.1/24
set interfaces gr-1/1/10 unit 2 tunnel source 10.80.1.1 destination 10.80.1.3
set interfaces gr-1/1/10 unit 3 family inet address 10.201.1.1/24
set interfaces gr-1/1/10 unit 3 tunnel source 10.90.1.1 destination 10.90.1.3
set interfaces gr-1/1/10 unit 4 family inet address 10.202.1.1/24
set interfaces gr-1/1/10 unit 4 tunnel source 10.100.1.1 destination 10.100.1.3
set interfaces gr-1/1/10 hierarchical-scheduler
set routing-options static route 10.2.2.0/24 next-hop gr-1/1/10.1
set routing-options static route 10.3.3.0/24 next-hop gr-1/1/10.2
set routing-options static route 10.4.4.0/24 next-hop gr-1/1/10.3
set routing-options static route 10.5.5.0/24 next-hop gr-1/1/10.4
```

Configuring Output Scheduling and Shaping at GRE Tunnel Physical and Logical Interfaces

```
set class-of-service forwarding-classes queue 0 be
set class-of-service forwarding-classes queue 1 ef
set class-of-service forwarding-classes queue 2 af
set class-of-service forwarding-classes queue 3 nc
set class-of-service forwarding-classes queue 4 be1
set class-of-service forwarding-classes queue 5 ef1
set class-of-service forwarding-classes queue 6 af1
set class-of-service forwarding-classes queue 7 nc1
set class-of-service classifiers inet-precedence gr-inet forwarding-class be loss-priority low
code-points 000
set class-of-service classifiers inet-precedence gr-inet forwarding-class ef loss-priority low
code-points 001
set class-of-service classifiers inet-precedence gr-inet forwarding-class af loss-priority low
code-points 010
set class-of-service classifiers inet-precedence gr-inet forwarding-class nc loss-priority low
code-points 011
set class-of-service classifiers inet-precedence gr-inet forwarding-class be1 loss-priority low
code-points 100
set class-of-service classifiers inet-precedence gr-inet forwarding-class ef1 loss-priority low
code-points 101
set class-of-service classifiers inet-precedence gr-inet forwarding-class af1 loss-priority low
code-points 110
set class-of-service classifiers inet-precedence gr-inet forwarding-class nc1 loss-priority low
code-points 111
set class-of-service interfaces ge-1/1/0 unit 0 classifiers inet-precedence gr-inet
set class-of-service schedulers be_sch transmit-rate percent 30
set class-of-service schedulers ef_sch transmit-rate percent 40
set class-of-service schedulers af_sch transmit-rate percent 25
set class-of-service schedulers nc_sch transmit-rate percent 5
set class-of-service schedulers be1_sch transmit-rate percent 60
set class-of-service schedulers be1_sch priority low
set class-of-service schedulers ef1_sch transmit-rate percent 40
set class-of-service schedulers ef1_sch priority medium-low
set class-of-service schedulers af1_sch transmit-rate percent 10
set class-of-service schedulers af1_sch priority strict-high
set class-of-service schedulers nc1_sch shaping-rate percent 10
set class-of-service schedulers nc1_sch priority high
set class-of-service scheduler-maps sch_map_1 forwarding-class be scheduler be_sch
set class-of-service scheduler-maps sch_map_1 forwarding-class ef scheduler ef_sch
set class-of-service scheduler-maps sch_map_1 forwarding-class af scheduler af_sch
set class-of-service scheduler-maps sch_map_1 forwarding-class nc scheduler nc_sch
set class-of-service scheduler-maps sch_map_2 forwarding-class be scheduler be1_sch
set class-of-service scheduler-maps sch_map_2 forwarding-class ef scheduler ef1_sch
set class-of-service scheduler-maps sch_map_3 forwarding-class af scheduler af_sch
set class-of-service scheduler-maps sch_map_3 forwarding-class nc scheduler nc_sch
set class-of-service traffic-control-profiles gr-ifl-tcp3 guaranteed-rate 5m
set class-of-service traffic-control-profiles gr-ifd-tcp shaping-rate 10m
set class-of-service traffic-control-profiles gr-ifd-tcp-remain shaping-rate 7m
set class-of-service traffic-control-profiles gr-ifd-tcp-remain guaranteed-rate 4m
set class-of-service traffic-control-profiles gr-ifl-tcp1 scheduler-map sch_map_1
set class-of-service traffic-control-profiles gr-ifl-tcp1 shaping-rate 8m
set class-of-service traffic-control-profiles gr-ifl-tcp1 guaranteed-rate 3m
set class-of-service traffic-control-profiles gr-ifl-tcp2 scheduler-map sch_map_2
set class-of-service traffic-control-profiles gr-ifl-tcp2 guaranteed-rate 2m
set class-of-service traffic-control-profiles gr-ifl-tcp3 scheduler-map sch_map_3
set class-of-service interfaces gr-1/1/10 output-traffic-control-profile gr-ifd-tcp
set class-of-service interfaces gr-1/1/10 output-traffic-control-profile-remaining gr-ifd-remain
set class-of-service interfaces gr-1/1/10 unit 1 output-traffic-control-profile gr-ifl-tcp1
set class-of-service interfaces gr-1/1/10 unit 2 output-traffic-control-profile gr-ifl-tcp2
set class-of-service interfaces gr-1/1/10 unit 3 output-traffic-control-profile gr-ifl-tcp3
```

Configuring Interfaces, Hierarchical Scheduling on the GRE Tunnel Physical Interface, and Static Routes

Step-by-Step Procedure

To configure GRE tunnel interfaces (including enabling hierarchical scheduling) and static routes:

1. Configure the amount of bandwidth for tunnel services on the physical interface.

   ```
   [edit]
   user@host# set chassis fpc 1 pic 1 tunnel-services bandwidth 1g
   ```

2. Configure the GRE tunnel device output logical interface.

   ```
   [edit]
   user@host# set interfaces ge-1/1/0 unit 0 family inet address 10.6.6.1/24
   ```

3. Configure the GRE tunnel device output logical interface.

   ```
   [edit]
   user@host# set interfaces ge-1/1/1 unit 0 family inet address 10.70.1.1/24 arp 10.70.1.3 mac
   00:00:03:00:04:00
   user@host# set interfaces ge-1/1/1 unit 0 family inet address 10.80.1.1/24 arp 10.80.1.3 mac
   00:00:03:00:04:01
   user@host# set interfaces ge-1/1/1 unit 0 family inet address 10.90.1.1/24 arp 10.90.1.3 mac
   00:00:03:00:04:02
   user@host# set interfaces ge-1/1/1 unit 0 family inet address 10.100.1.1/24 arp 10.100.1.3
   mac 00:00:03:00:04:04
   ```

4. Convert the output logical interface to four GRE tunnel interfaces.

```
[edit]
user@host# set interfaces gr-1/1/10 unit 1 family inet address 10.100.1.1/24
user@host# set interfaces gr-1/1/10 unit 1 tunnel source 10.70.1.1 destination 10.70.1.3
user@host# set interfaces gr-1/1/10 unit 2 family inet address 10.200.1.1/24
user@host# set interfaces gr-1/1/10 unit 2 tunnel source 10.80.1.1 destination 10.80.1.3
user@host# set interfaces gr-1/1/10 unit 3 family inet address 10.201.1.1/24
user@host# set interfaces gr-1/1/10 unit 3 tunnel source 10.90.1.1 destination 10.90.1.3
user@host# set interfaces gr-1/1/10 unit 4 family inet address 10.202.1.1/24
user@host# set interfaces gr-1/1/10 unit 4 tunnel source 10.100.1.1 destination 10.100.1.3
```

5. Enable the GRE tunnel interfaces to use hierarchical scheduling.

```
[edit]
user@host# set interfaces gr-1/1/10 hierarchical-scheduler
```

6. Install static routes in the routing table so that the device routes IPv4 traffic to the GRE tunnel source interfaces.

   Traffic destined to the subnets 10.2.2.0/24, 10.3.3.0/24, 10.4.4.0/24, and 10.5.5.0/24 is routed to the tunnel interfaces at IP addresses 10.70.1.1, 10.80.1.1, 10.90.1.1, and 10.100.1.1, respectively.

```
[edit]
user@host# set routing-options static route 10.2.2.0/24 next-hop gr-1/1/10.1
user@host# set routing-options static route 10.3.3.0/24 next-hop gr-1/1/10.2
user@host# set routing-options static route 10.4.4.0/24 next-hop gr-1/1/10.3
user@host# set routing-options static route 10.5.5.0/24 next-hop gr-1/1/10.4
```

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

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

Results

From configuration mode, confirm your configuration by entering the `show chassis fpc 1 pic 1`, `show interfaces ge-1/1/0`, `show interfaces ge-1/1/1`, `show interfaces gr-1/1/10`, and `show routing-options` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Confirm the configuration of interfaces, hierarchical scheduling on the GRE tunnel physical interface, and static routes.

```
user@host# show chassis  fpc 1 pic 1
tunnel-services {
    bandwidth 1g;
}

user@host# show interfaces ge-1/1/0
unit 0 {
    family inet {
        address 10.6.6.1/24;
    ]
}

user@host# show interfaces ge-1/1/1
unit 0 {
    family inet {
        address 10.70.1.1/24 (
            arp 10.70.1.3 mac 00:00:03:00:04:00;
        }
        address 10.80.1.1/24 {
            arp 10.80.1.3 mac 00:00:03:00:04:01;
        }
        address 10.90.1.1/24 {
            arp 10.90.1.3 mac 00:00:03:00:04:02;
        }
        address 10.100.1.1/24 {
            arp 10.100.1.3 mac 00:00:03:00:04:04;
        }
    ]
}

user@host# show interfaces gr-1/1/10
hierarchical-scheduler;
unit 1 {
    tunnel {
        destination 10.70.1.3;
        source 10.70.1.1;
    }
    family inet {
        address 10.100.1.1/24;
    }
}
unit 2 {
    tunnel {
        destination 10.80.1.3;
        source 10.80.1.1;
    }
    family inet {
        address 10.200.1.1/24;
    }
}
unit 3 {
    tunnel {
        destination 10.90.1.3;
        source 10.90.1.1;
    }
    family inet {
        address 10.201.1.1/24;
    }
}
unit 4 {
    tunnel {
        destination 10.100.1.3;
        source 10.100.1.1;
    }
    family inet {
        address 10.202.1.1/24;
    }
}

user@host# show routing-options
static {
    route 10.2.2.0/24 next-hop gr-1/1/10.1;
    route 10.3.3.0/24 next-hop gr-1/1/10.2;
    route 10.4.4.0/24 next-hop gr-1/1/10.3;
    route 10.5.5.0/24 next-hop gr-1/1/10.4;
}
```

Measuring GRE Tunnel Transmission Rates Without Shaping Applied

Step-by-Step Procedure

To establish a baseline measurement, note the transmission rates at each GRE tunnel source.

1. Pass traffic through the GRE tunnel at logical interfaces `gr-1/1/10.1`, `gr-1/1/10.2`, and `gr-1/1/10.3`.

2. To display the traffic rates at each GRE tunnel source, use the `show interfaces queue` operational mode command.

   The following example command output shows detailed CoS queue statistics for logical interface gr-1/1/10.1 (the GRE tunnel from source IP address 10.70.1.1 to destination IP address 10.70.1.3).

```
user@host> show interfaces queue gr-1/1/10.1
Logical interface gr-1/1/10.1 (Index 331) (SNMP ifIndex 4045)
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Burst size: 0
Queue: 0, Forwarding classes: be
  Queued:
    Packets              :             31818312                102494 pps
    Bytes                :           6522753960             168091936 bps
  Transmitted:
    Packets              :              1515307                  4879 pps
    Bytes                :            310637935               8001632 bps
    Tail-dropped packets :             21013826                 68228 pps
    RED-dropped packets  :              9289179                 29387 pps
     Low                 :              9289179                 29387 pps
     Medium-low          :                    0                     0 pps
     Medium-high         :                    0                     0 pps
     High                :                    0                     0 pps
    RED-dropped bytes    :           1904281695              48194816 bps
     Low                 :           1904281695              48194816 bps
     Medium-low          :                    0                     0 bps
     Medium-high         :                    0                     0 bps
     High                :                    0                     0 bps
...
```

NOTE: This step shows command output for queue `0` (forwarding class `be`) only.

The command output shows that the GRE tunnel device transmits traffic from queue `0` at a rate of 4879 pps. Allowing for 182 bytes per Layer 3 packet, preceded by 24 bytes of GRE overhead (a 20-byte delivery header consisting of the IPv4 packet header followed by 4 bytes for GRE flags plus encapsulated protocol type), the traffic rate received at the tunnel destination device is 8,040,592 bps:

The command output shows that the GRE tunnel device transmits traffic from queue `0` at a rate of 4879 pps. Allowing for 182 bytes per Layer 3 packet, preceded by 24 bytes of GRE overhead (a 20-byte delivery header consisting of the IPv4 packet header followed by 4 bytes for GRE flags plus encapsulated protocol type), the traffic rate received at the tunnel destination device is 8,040,592 bps:

```
4879 packets/second X 206 bytes/packet X 8 bits/byte = 8,040,592 bits/second
```

Configuring Output Scheduling and Shaping at GRE Tunnel Physical and Logical Interfaces

Step-by-Step Procedure

To configure the GRE tunnel device with scheduling and shaping at GRE tunnel physical and logical interfaces:

1. Define eight transmission queues.

```
[edit]
user@host# set class-of-service forwarding-classes queue 0 be
user@host# set class-of-service forwarding-classes queue 1 ef
user@host# set class-of-service forwarding-classes queue 2 af
user@host# set class-of-service forwarding-classes queue 3 nc
user@host# set class-of-service forwarding-classes queue 4 be1
user@host# set class-of-service forwarding-classes queue 5 ef1
user@host# set class-of-service forwarding-classes queue 6 af1
user@host# set class-of-service forwarding-classes queue 7 nc1
```

NOTE: To configure up to eight forwarding classes with one-to-one mapping to output queues, use the `queue` statement at the `[edit class-of-service forwarding-classes]` hierarchy level.

If you need to configure up to 16 forwarding classes with multiple forwarding classes mapped to single queues, use the `class` statement instead.

2. Configure BA classifier `gr-inet` that, based on IPv4 precedence bits set in an incoming packet, sets the forwarding class, loss-priority value, and DSCP bits of the packet.

```
[edit]
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class be loss-
priority low code-points 000
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class ef loss-
priority low code-points 001
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class af loss-
priority low code-points 010
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class nc loss-
priority low code-points 011
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class be1 loss-
priority low code-points 100
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class ef1 loss-
priority low code-points 101
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class af1 loss-
priority low code-points 110
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class nc1 loss-
priority low code-points 111
```

3. Apply BA classifier `gr-inet` to the GRE tunnel device input at logical interface ge-1/1/0.0.

```
[edit]
user@host# set class-of-service interfaces ge-1/1/0 unit 0 classifiers inet-precedence gr-inet
```

4. Define a scheduler for each forwarding class.

```
[edit]
user@host# set class-of-service schedulers be_sch transmit-rate percent 30
user@host# set class-of-service schedulers ef_sch transmit-rate percent 40
user@host# set class-of-service schedulers af_sch transmit-rate percent 25
user@host# set class-of-service schedulers nc_sch transmit-rate percent 5
user@host# set class-of-service schedulers be1_sch transmit-rate percent 60
user@host# set class-of-service schedulers be1_sch priority low
user@host# set class-of-service schedulers ef1_sch transmit-rate percent 40
user@host# set class-of-service schedulers ef1_sch priority medium-low
user@host# set class-of-service schedulers af1_sch transmit-rate percent 10
user@host# set class-of-service schedulers af1_sch priority strict-high
```

```
user@host# set class-of-service schedulers nc1_sch shaping-rate percent 10
user@host# set class-of-service schedulers nc1_sch priority high
```

5. Define a scheduler map for each of three GRE tunnels.

```
[edit]
user@host# set class-of-service scheduler-maps sch_map_1 forwarding-class be scheduler be_sch
user@host# set class-of-service scheduler-maps sch_map_1 forwarding-class ef scheduler ef_sch
user@host# set class-of-service scheduler-maps sch_map_1 forwarding-class af scheduler af_sch
user@host# set class-of-service scheduler-maps sch_map_1 forwarding-class nc scheduler nc_sch
user@host# set class-of-service scheduler-maps sch_map_2 forwarding-class be scheduler be1_sch
user@host# set class-of-service scheduler-maps sch_map_2 forwarding-class ef scheduler ef1_sch
user@host# set class-of-service scheduler-maps sch_map_3 forwarding-class af scheduler af_sch
user@host# set class-of-service scheduler-maps sch_map_3 forwarding-class nc scheduler nc_sch
```

6. Define traffic control profiles for three GRE tunnel interfaces.

```
[edit]
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp1 scheduler-map sch_map_1
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp1 shaping-rate 8m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp1 guaranteed-rate 3m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp2 scheduler-map sch_map_2
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp2 guaranteed-rate 2m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp3 scheduler-map sch_map_3
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp3 guaranteed-rate 5m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp shaping-rate 10m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp-remain shaping-rate 7m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp-remain guaranteed-rate 4m
```

7. Apply CoS scheduling and shaping to the output traffic at the physical interface and logical interfaces.

```
[edit]
user@host# set class-of-service interfaces gr-1/1/10 output-traffic-control-profile gr-ifd-tcp
user@host# set class-of-service interfaces gr-1/1/10 output-traffic-control-profile-remaining
gr-ifd-remain
user@host# set class-of-service interfaces gr-1/1/10 unit 1 output-traffic-control-profile gr-
ifl-tcp1
user@host# set class-of-service interfaces gr-1/1/10 unit 2 output-traffic-control-profile gr-
ifl-tcp2
```

```
user@host# set class-of-service interfaces gr-1/1/10 unit 2 output-traffic-control-profile gr-
ifl-tcp3
```

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

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

Results

From configuration mode, confirm your configuration by entering the `show class-of-service forwarding-classes`, `show class-of-service classifiers`, `show class-of-service interfaces ge-1/1/0`, `show class-of-service schedulers`, `show class-of-service scheduler-maps`, `show class-of-service traffic-control-profiles`, and `show class-of-service interfaces gr-1/1/10` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Confirm the configuration of output scheduling and shaping at the GRE tunnel physical and logical interfaces.

```
user@host# show class-of-service forwarding-classes
queue 0 be;
queue 1 ef;
queue 2 af;
queue 3 nc;
queue 4 be1;
queue 5 ef1;
queue 6 af1;
queue 7 nc1;

user@host# show class-of-service classifiers
inet-precedence gr-inet {
    forwarding-class be {
        loss-priority low code-points 000;
    }
    forwarding-class ef {
        loss-priority low code-points 001;
    }
    forwarding-class af {
        loss-priority low code-points 010;
    }
    forwarding-class nc {
        loss-priority low code-points 011;
    }
    forwarding-class be1 {
        loss-priority low code-points 100;
    }
    forwarding-class ef1 {
        loss-priority low code-points 101;
    }
    forwarding-class af1 {
        loss-priority low code-points 110;
    }
    forwarding-class nc1 {
        loss-priority low code-points 111;
    }
}

user@host# show class-of-service interfaces ge-1/1/0
unit 0 {
    classifiers {
        inet-precedence gr-inet;
    }
}

user@host# show class-of-service schedulers
be_sch {
    transmit-rate percent 30;
}
ef_sch {
    transmit-rate percent 40;
}
af_sch {
    transmit-rate percent 25;
}
nc_sch {
    transmit-rate percent 5;
}
be1_sch {
    transmit-rate percent 60;
    priority low;
}
ef1_sch {
    transmit-rate percent 40;
    priority medium-low;
```

```
}
af1_sch {
    transmit-rate percent 10;
    priority strict-high;
}
nc1_sch {
    shaping-rate percent 10;
    priority high;
}

user@host# show class-of-service scheduler-maps
sch_map_1 {
    forwarding-class be scheduler be_sch;
    forwarding-class ef scheduler ef_sch;
    forwarding-class af scheduler af_sch;
    forwarding-class nc scheduler nc_sch;
}
sch_map_2 {
    forwarding-class be scheduler be1_sch;
    forwarding-class ef scheduler ef1_sch;
}
sch_map_3 {
    forwarding-class af scheduler af_sch;
    forwarding-class nc scheduler nc_sch;
}

user@host# show class-of-service traffic-control-profiles
gr-ifl-tcp1 {
    scheduler-map sch_map_1;
    shaping-rate 8m;
    guaranteed-rate 3m;
}
gr-ifl-tcp2 {
    scheduler-map sch_map_2;
    guaranteed-rate 2m;
}
gr-ifl-tcp3 {
    scheduler-map sch_map_3;
    guaranteed-rate 5m;
}
gr-ifd-remain {
    shaping-rate 7m;
    guaranteed-rate 4m;
```

```
}
gr-ifd-tcp {
    shaping-rate 10m;
}

user@host# show class-of-service interfaces gr-1/1/10
gr-1/1/10 {
    output-traffic-control-profile gr-ifd-tcp;
    output-traffic-control-profile-remaining gr-ifd-remain;
    unit 1 {
        output-traffic-control-profile gr-ifl-tcp1;
    }
    unit 2 {
        output-traffic-control-profile gr-ifl-tcp2;
    }
    unit 3 {
        output-traffic-control-profile gr-ifl-tcp3;
    }
}
```

Verification

IN THIS SECTION

Confirm that the configurations are working properly.

Verifying That Scheduling and Shaping Are Attached to the GRE Tunnel Interfaces

Purpose

Verify the association of traffic control profiles with GRE tunnel interfaces.

Action

Verify the traffic control profile attached to the GRE tunnel physical interface by using the **show class-of-service interface gr-1/1/10 detail** operational mode command.

-
  ```
  user@host> show class-of-service interface gr-1/1/10 detail
  Physical interface: gr-1/1/10, Enabled, Physical link is Up
    Type: GRE, Link-level type: GRE, MTU: Unlimited, Speed: 1000mbps
    Device flags   : Present Running
    Interface flags: Point-To-Point SNMP-Traps

  Physical interface: gr-1/1/10, Index: 220
  Queues supported: 8, Queues in use: 8
    Output traffic control profile: gr-ifd-tcp, Index: 17721
    Output traffic control profile remaining: gr-ifd-remain, Index: 58414
    Congestion-notification: Disabled

    Logical interface gr-1/1/10.1
      Flags: Point-To-Point SNMP-Traps 0x4000 IP-Header
  10.70.1.3:10.70.1.1:47:df:64:0000000000000000 Encapsulation: GRE-NULL
      Gre keepalives configured: Off, Gre keepalives adjacency state: down
      inet  10.100.1.1/24
    Logical interface: gr-1/1/10.1, Index: 331
  Object                  Name                   Type                    Index
  Traffic-control-profile gr-ifl-tcp1            Output                  17849
  Classifier              ipprec-compatibility   ip                         13

    Logical interface gr-1/1/10.2
      Flags: Point-To-Point SNMP-Traps 0x4000 IP-Header
  10.80.1.3:10.80.1.1:47:df:64:0000000000000000 Encapsulation: GRE-NULL
      Gre keepalives configured: Off, Gre keepalives adjacency state: down
      inet  10.200.1.1/24
    Logical interface: gr-1/1/10.2, Index: 332
  Object                  Name                   Type                    Index
  Traffic-control-profile gr-ifl-tcp2            Output                  17856
  Classifier              ipprec-compatibility   ip                         13

    Logical interface gr-1/1/10.3
      Flags: Point-To-Point SNMP-Traps 0x4000 IP-Header
  10.90.1.3:10.90.1.1:47:df:64:0000000000000000 Encapsulation: GRE-NULL
      Gre keepalives configured: Off, Gre keepalives adjacency state: down
      inet  10.201.1.1/24
  ```

```
  Logical interface: gr-1/1/10.3, Index: 333
Object                 Name                  Type                   Index
Traffic-control-profile gr-ifl-tcp3          Output                 17863
Classifier             ipprec-compatibility   ip                       13
```

Meaning

Ingress IPv4 traffic routed to GRE tunnels on the device is subject to CoS output scheduling and shaping.

Verifying That Scheduling and Shaping Are Functioning at the GRE Tunnel Interfaces

Purpose

Verify the traffic rate shaping at the GRE tunnel interfaces.

Action

1. Pass traffic through the GRE tunnel at logical interfaces `gr-1/1/10.1`, `gr-1/1/10.2`, and `gr-1/1/10.3`.
2. To verify the rate shaping at each GRE tunnel source, use the `show interfaces queue` operational mode command.

   The following example command output shows detailed CoS queue statistics for logical interface gr-1/1/10.1 (the GRE tunnel from source IP address 10.70.1.1 to destination IP address 10.70.1.3):

```
user@host> show interfaces queue gr-1/1/10.1
Logical interface gr-1/1/10.1 (Index 331) (SNMP ifIndex 4045)
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Burst size: 0
Queue: 0, Forwarding classes: be
  Queued:
    Packets              :             59613061                 51294 pps
    Bytes                :          12220677505              84125792 bps
  Transmitted:
    Packets              :              2230632                  3039 pps
    Bytes                :            457279560               4985440 bps
    Tail-dropped packets :              4471146                  2202 pps
    RED-dropped packets  :             52911283                 46053 pps
     Low                 :             49602496                 46053 pps
     Medium-low          :               0                     0 pps
     Medium-high         :               0                     0 pps
     High                :         3308787                     0 pps
    RED-dropped bytes    :     10846813015              75528000 bps
     Low                 :     10168511680              75528000 bps
     Medium-low          :               0                     0 bps
     Medium-high         :               0                     0 bps
     High                :       678301335                     0 bps
Queue: 1, Forwarding classes: ef
  Queued:
    Packets              :        15344874                 51295 pps
    Bytes                :      3145699170              84125760 bps
  Transmitted:
    Packets              :          366115                  1218 pps
    Bytes                :        75053575               1997792 bps
    Tail-dropped packets :          364489                  1132 pps
    RED-dropped packets  :        14614270                 48945 pps
     Low                 :        14614270                 48945 pps
     Medium-low          :               0                     0 pps
     Medium-high         :               0                     0 pps
     High                :               0                     0 pps
    RED-dropped bytes    :      2995925350              80270528 bps
     Low                 :      2995925350              80270528 bps
     Medium-low          :               0                     0 bps
     Medium-high         :               0                     0 bps
     High                :               0                     0 bps
...
```

NOTE: This step shows command output for queue `0` (forwarding class `be`) and queue `1` (forwarding class `ef`) only.

Meaning

Now that traffic shaping is attached to the GRE tunnel interfaces, the command output shows that traffic shaping specified for the tunnel at logical interface gr-1/1/10.1 (`shaping-rate 8m` and `guaranteed-rate 3m`) is honored.

- For queue `0`, the GRE tunnel device transmits traffic at a rate of 3039 pps. The traffic rate received at the tunnel destination device is 5,008,272 bps:

```
3039 packets/second X 206 bytes/packet X 8 bits/byte = 5,008,272 bits/second
```

- For queue `0`, the GRE tunnel device transmits traffic at a rate of 1218 pps. The traffic rate received at the tunnel destination device is 2,007,264 bps:

```
1218 packets/second X 206 bytes/packet X 8 bits/byte = 2,007,264 bits/second
```

Compare these statistics to the baseline measurements taken without traffic shaping, as described in "Measuring GRE Tunnel Transmission Rates Without Shaping Applied" on page 766.

RELATED DOCUMENTATION

Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | **757**

Configuring Traffic Control Profiles for Shared Scheduling and Shaping | **350**

CoS-Based Interface Counters for IPv4 or IPv6 Aggregate on Layer 2

Layer 2 CoS-based traffic metering is available for MX series routers. It can be used in greenfield deployments or as a replacement for term-matching counters configured via firewall filters. (Term-matching counters configured via firewall filters can have several drawbacks, including a one-filter-per-family limit, the inclusion of overhead bytes, and less operational efficiency than CoS-based counters). With CoS-based counters, a single aggregate counter per forwarding class can be used for inet and inet6 flows. Both bytes and packet total are counted. Note that flow rates are not measured, and forwarding-class accounting for host-bound traffic is not supported.

You can configure the counters with any or all of the following parameters:

- Logical | physical interfaces
- IPv4 | IPv6 traffic
- Unicast | multicast traffic
- Ingress | egress flows

CoS-based interface counters are highly accurate and can be configured to exclude overhead bytes (such as protocol encapsulations) so end-customer packets can be differentiated from other traffic. At ingress only packets forwarded to the fabric are counted, and at egress only packets forwarded to the WAN are counted. In other words, `forwarding-class accounting` applies to transit traffic only, not host-generated or host-bound traffic. Non-relevant network protocols such as ARP, BFD, and EOAM, as well as dropped packets, are not counted.

To support native interface counters, a new CLI option, `enhanced`, is introduced under `forwarding-class accounting` at both the physical and logical interface levels:

```
interfaces {
    interface-name{
        forwarding-class-accounting {
            enhanced {
                overhead-bytes overhead-value(;
                traffic-type (ucast | mcast);
                family (ipv4 | ipv6 | both );
                direction (ingress | egress | both);
            }
        }
    }
}
```

To view additional counter details, run the following `show` commands:

- `show interfaces forwarding-class-counters` *`interface-name`*
- `show class-of-service interface` *`interface-name`* `comprehensive`
- `show class-of-service interface` *`interface-name`* `detail`

The `comprehensive` option shows both forwarding-class accounting parameters and the forwarding-class counter, whereas `detail` shows only the forwarding-class accounting parameters. It does not display counters for each forwarding class.

RELATED DOCUMENTATION

forwarding-class-accounting

enhanced

show class-of-service interface

Enabling a Timestamp for Ingress and Egress Queue Packets

You can enable a packet timestamp feature to record the time at which the last packet is enqueued for CoS ingress and egress queues. Timestamps are enabled and reported per FPC. When the feature is enabled, the Packet Forwarding Engine begins collection the timestamp for all ingress and egress queue counters on the FPC. By default, packet timestamp information is not collected.

To activate packet timestamp collection for CoS ingress and egress queues:

- Enable the timestamp on the desired FPC.

```
[edit chassis fpc slot-number traffic-manager]
user@host# set packet-timestamp enable
```

NOTE: When you enable or disable the packet timestamp for an FPC that is already up, the FPC is automatically rebooted when you commit the changes. The action takes effect when the FPC is back up.

NOTE: For aggregated Ethernet interfaces, enable the packet timestamp on all FPCs that have an aggregated Ethernet leg. When you display the queue statistics for the interface, the timestamps for all the legs are shown.

The following commands display the collected timestamps in the Last-packet enqueued field:

- `show interfaces queue both-ingress-egress` *`interface-name`*
- `show interfaces queue` *`interface-name`*
- `show interfaces queue` *`interface-name`* `aggregate`

RELATED DOCUMENTATION

CoS-Based Interface Counters for IPv4 or IPv6 Aggregate on Layer 2 | 777

CHAPTER 13

Configuring Class of Service on PTX Series Packet Transport Routers

IN THIS CHAPTER

CoS Features and Limitations on PTX Series Routers

Table 74 on page 781 summarizes CoS features and limitations on PTX Series Packet Transport Routers.

The following table lists the CoS features supported on the PTX Series router, as well as the limitations relevant to the PTX Series router. Note that this list is a subset of the overall CoS feature set.

Table 74: CoS Features and Limitations on PTX Series Routers

CoS Feature	Capacity	Comments
Classifiers		
Maximum number per PFE	64	L2 classifiers (sum of IEEE-802.1p and IEEE-802.1ad cannot exceed 32) DSCP and IP precedence classifiers (sum of DSCP and IP precedence classifiers cannot exceed 32) DSCP IPv6 classifiers MPLS EXP classifiers
`dscp`	Yes	DSCP and IP precedence classifiers cannot be configured on the same logical interface.
`dscp-ipv6`	Yes	Separate classifiers can be applied for IPv4 and IPv6 packets per logical interface.
`ieee-802.1p` and `ieee-802.1ad`	Yes	You can associate IEEE-802.1p and IEEE-802.1ad classifiers with any other type of classifier on the same logical interface. For L3 packets, an L3 classifier takes precedence over an IEEE classifier.
`inet-precedence`	Yes	
`mpls-exp`	Yes	**NOTE**: MPLS EXP classifiers are not supported on PTX10001-20C routers.
Loss priorities based on the Frame Relay discard eligible (DE) bit	No	
Drop Profiles		
Maximum number	32	You can configure up to 32 *drop-profiles* in the PTX chassis.

Table 74: CoS Features and Limitations on PTX Series Routers *(Continued)*

CoS Feature	Capacity	Comments
Per queue	Yes	
Per loss priority	Yes	
Per Transmission Control Protocol (TCP) bit	No	
`protocol` option	`protocol any` only	PTX Series PFEs not support drop-profile assignments for a queue based on the protocol. As a consequence, the `protocol` option for the `drop-profile-map` configuration statement is treated as `protocol any`.

Table 74: CoS Features and Limitations on PTX Series Routers *(Continued)*

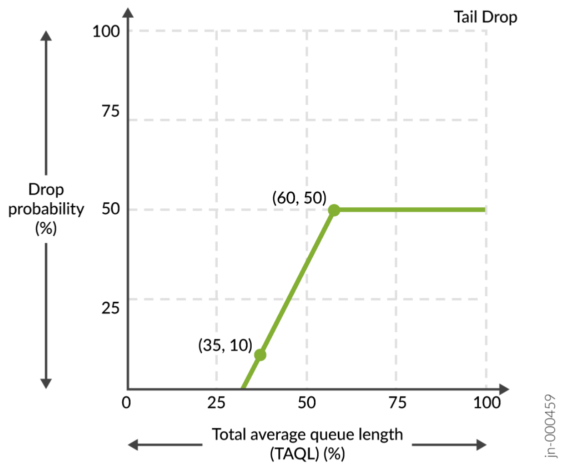
CoS Feature	Capacity	Comments

NOTE: PTX10K-LC1201 and PTX10K-LC1202 line cards on PTX10001-36MR, PTX10004, PTX10008, and PTX10016 routers support one or two drop profile points (fill level/drop probability pairs) per drop profile:

- Discrete and interpolated drop profiles are equivalent on these platforms; the hardware ignores the interpolated option. You cannot configure both discrete and interpolated pairs on a single drop profile.
- Point one drop probability must be less than or equal to 25%.
- You must configure fill level and drop probabilities in pairs and in ascending order.

Figure 54 on page 783 shows how two point RED curves generate a single line.

Figure 54: Two Drop Profile Points



Although the first point is (35,10), we see the drop profile points starting earlier where the line intersects the X axis.

At the second point (60,50), the drop probability is maximum and remains at point 50 from TAQL 60% through 100%, after which the tail drop happens.

Policing		

Table 74: CoS Features and Limitations on PTX Series Routers *(Continued)*

CoS Feature	Capacity	Comments
Traffic policing	Yes	
Two-rate tricolor marking (TCM)	Yes	
Queuing		
Priority	Yes (4)	On PTX Series routers, strict-high-priority queues and high-priority queues are assigned different hardware priorities. Strict-high-priority can starve other queues if a rate limiter is not applied on PTX Series routers.
Per-queue output statistics	Yes	Red-dropped counters are not maintained per drop precedence. Also tail drop counters always show zero because packets are always dropped by the RED algorithm. On PTX Series routers, transmitted byte counters are computed using the full Layer 2 overhead (including all L2 encapsulation and CRC) plus 12 for the inter-packet gap plus 8 for the preamble. On Junos OS Evolved PTX Series routers , the tail-dropped counters and the RED-dropped counters are displayed separately in the output of the `show interfaces queue` command. On Junos OS PTX Series routers, tail-dropped counters are always zero. All the packet drops are shown as RED-dropped in the `show interfaces voq` output.
`transmit-rate percent`	Yes	Percentage transmit rate for a scheduler has the range 1 through 100 percent. Unconfigured interfaces are equivalent to `percent 0`.
Excess bandwidth sharing	Yes	On PTX Series routers, excess bandwidth is shared based on the ratio of the configured transfer rate. Therefore, all priority queues get a share of excess bandwidth.

Table 74: CoS Features and Limitations on PTX Series Routers *(Continued)*

CoS Feature	Capacity	Comments
Virtual Output Queueing	Yes	PTX Series routers support a virtual output queuing (VOQ) architecture and the fabric schedulers utilize the CoS scheduling parameters to configure the fabric schedulers. There is not a separate configuration for the fabric schedulers on PTX Series routers. With the VOQ architecture, packets are queued and dropped on ingress during congestion.
Buffer latency	Yes	On PTX Series routers, `buffer-size percent 100` requests the platform's full buffer latency at the port or shaping rate, that is, a nominal queue limit of (`productFullLatency × portRate`) bytes. For all PTX routers, the `productFullLatency` is 100 ms except for the PTX10001, and line cards LC1201 and LC1202, which have a 25 ms buffer.
Rewrite Markers		
Maximum number per PFE	64	The sum of L2 and L3 rewrite rules cannot exceed 64.
`dscp`	Yes	
`dscp-ipv6`	Yes	
`ieee-802.1p` and `ieee-802.1ad`	Yes	PTX Series routers support Layer 2 rewrite of 802.1p and 802.1ad, to either the outer VLAN tag, or both outer and inner VLAN tags. L2 and L3 rewrites can be applied to the same packet simultaneously.
`inet-precedence`	No	
`mpls-exp`	Yes	PTX Series routers support rewrite of both DSCP and DSCP IPv6 for protocol MPLS.

For PTX Series routers running Junos OS Evolved:

- PTX Series routers running Junos OS Evolved support Layer 3 VPNs.

- Junos OS Evolved does not support applying classifier and rewrite rules to IRB interfaces on PTX Series routers.

Understanding Scheduling on PTX Series Routers

IN THIS SECTION

This topic covers class of service packet scheduling for interfaces on PTX Series routers:

Output Queue Priorities Supported by the Junos OS CLI on PTX Series Routers

Output queues on the PTX Series interface hardware support these values for the queue priority—high, medium, low, and excess. The Junos OS supports five queue priority levels: `strict-high`, `high`, `medium-high`, `medium-low`, and `low`.

NOTE: If a strict-high-priority queue is constantly loaded to 100 percent of traffic capacity, other queues are starved. Queue starvation can cause the interface hardware to generate interrupts.
This starvation can be alleviated by using a rate-limiter on the strict-high queues.

Scheduling Processes on PTX Series Routers

Physical interfaces on PTX Series routers support two mutually exclusive scheduling processes:

- *normal scheduling* (default mode) — A queue's transmit-rate is used to determine whether it is operating within the *guaranteed region* or in the *excess region*.

  Within the *guaranteed region* (transmit-rates credits are positive), the scheduler uses the transmit rates to decide the bandwidth allocation. Queues that are at priority-level `low` or higher and have transmit-rate credits are serviced first by priority order and then within a priority level using packet round robin algorithm.

Within the *excess region* (for all queues in which `transmit-rate` credits are negative), CoS queues are selected based on the *weighted round-robin* (WRR) algorithm. If a queue does not have `excess-rate` configured, then its weight is set to 1.

If a queue is not configured with a `transmit-rate` statement (to specify the transmit rate or percentage of transmission capacity), then it will not be scheduled at `priority` level and only at `excess` level (unless priority is `strict-high`).

A queue must have a `transmit-rate` assigned to it in order to be scheduled at priority. `excess-rate` is used only to determine its weight when scheduled in the excess-region.

If multiple queues are in the excess region (queue priority `low`) and the `excess-rate` statement is used, then those queues are selected using the WRR algorithm.

The `remainder` keyword can be applied to transmit-rate as well as buffer size statements. This will cause the remaining portion of the specified resource to be assigned to the queue. The remaining resources for transmit-rate are the sum of all queues that specified a transmit-rate subtracted from the total transmit-rate available. If there are multiple queues assigned with the `remainder` keyword, the remainder of the resource is evenly divided amongst those queues.

There are two keywords that can be applied to the `transmit-rate` statement in order to limit the rate of a queue: `rate-limit` and `exact`, where `rate-limit` can be applied only to the `strict-high` queue and `exact` can be used for all other queues. The two keywords behave identically in that queues can transmit only up to the specified rate. All of their transmission will be scheduled at their configured priority level and they will never be scheduled at `excess-priority` level.

BEST PRACTICE: The `rate-limit` option of the `transmit-rate` configuration statement is allowed only on the strict-high queue. We recommend that you configure rate limit on strict-high queues because the other queues might not meet their guaranteed bandwidths.

Non strict-high queues can use the option `exact` to place a maximum `transmit-rate` limit on them, which is equivalent to `rate-limit`.

- *strict-priority scheduling* — Queues are processed in strict-priority order. There is no concept of guaranteed region and excess region. The packet scheduler is always operating in the guaranteed region, with the exception of priority `low`, which is always assigned to the excess priority level. The configured `transmit-rate` does not affect how the queue is serviced because packets are processed in order of queue priority. Among queues that are configured `low` priority, if `excess-rate` weights are configured, they are used by the hardware to perform WRR. Queues that are mapped to the same hardware priority or that have the same configured priority other than `low` are serviced in a packet round-robin fashion.

  The queues are serviced in strict-priority order until they have reached their transmit-rate (that is, their guaranteed rate) and then are demoted to the excess level.

NOTE: The `rate-limit` and `exact` options of the `transmit-rate` configuration statement have no effect when strict-priority scheduling is configured.

To configure *strict-priority scheduling* for a physical interface on a PTX Series router, include the `strict-priority-scheduler` and `scheduler-map` *`map-name`* configuration statements in the traffic control profile you associate with an output interface.

Strict-Priority and Scheduling Processes on PTX Series Routers

Table 75 on page 788 shows the different configurations available for Junos Priority Scheduler Modes, including those for Strict-Priority and Enhanced-Priority Modes. Table 75 on page 788 also shows how the output queue priority values in the Junos OS map to the output queue priorities supported by physical interfaces on PTX Series routers, and the scheduling action taken. The table shows differences for normal scheduling when `strict-high` is not configured, and for strict-priority scheduling.

Table 75: Strict-Priority and Scheduling Processes on PTX Series Routers

Junos Priority	Scheduler Mode	Normal			Strict Priority Scheduler
	Chassis Knob	enhanced-priority-mode (Junos OS only)		no-enhanced-priority-mode	*
	Strict-High Config	No	Yes	*	*
`strict-high`	—	—	High	High	High
`high`	—	High	Medium	High	High
`medium-high`	—	Medium	Low	Medium	Medium
`medium-low`	—	Medium	Low	Medium	Low
`low`	—	Low	Low	Low	Excess

NOTE: Packet scheduling is strict priority round-robin while the virtual output queues are in the guaranteed region.

After the virtual output queues consume their guaranteed credits, they are demoted to excess-priority scheduling, which is weighted round-robin.

The only exception is the strict-high priority, which is always scheduled as strict-high-priority.

PTX Series routers running Junos OS support `enhanced-priority-mode` and `no-enhanced-priority-mode` options under the `[edit chassis fpc` *`fpc-slot`* `traffic-manager]` configuration hierarchy.

PTX Series routers running Junos OS Evolved do not support `enhanced-priority-mode` mode.

NOTE: For PTX10K-LC1201 and PTX10K-LC1202 line cards on PTX10001-36MR, PTX10004, PTX10008, and PTX10016 routers running Junos OS Evolved:

- In strict priority scheduler mode, schedulers for queue 6 and queue 7 *must* have priority `low` (for `Excess` in hardware), and queues 0 through 5 should have higher priorities. These PTX models allow queues 6 and 7 to emulate two strict `Excess` priorities by setting large and small `excess-rate` values.

- When an FPC starts, if the system has any `traffic-control-profiles` *`profile-name`* `strict-priority-scheduler` configured, then all egress interfaces (`et-*`) of that FPC treat queues 6 and 7 specially. For proper transmit scheduling, configure `traffic-control-profiles` *`profile-name`* `strict-priority-scheduler` for *all* port interfaces if *any* traffic control profile has `strict-priority` scheduler.

RELATED DOCUMENTATION

Understanding CoS CLI Configuration Statements on PTX Series Routers | **855**

How Schedulers Define Output Queue Properties | **342**

Understanding Scheduling on PTX Series Routers | **786**

Example: Configuring Excess Rate for PTX Series Packet Transport Routers | **805**

show interfaces voq

strict-priority-scheduler

traffic-control-profiles

transmit-rate

Understand Virtual Output Queues

IN THIS SECTION

This section describes the virtual output queue (VOQ) architecture.

Refer to Feature Explorer for platform and release support information for VOQs.

Introduction to Virtual Output Queues

IN THIS SECTION

This topic introduces the VOQ architecture and how it operates with configurable class-of-service (CoS) components.

On supported platforms, Junos hardware CoS features use *virtual output queues* on the *ingress* to buffer and queue traffic for each egress output queue. Most platforms support up to eight egress output queues per output port (physical interface).

The traditional method of forwarding traffic through a router is based on buffering ingress traffic in input queues on ingress interfaces, forwarding the traffic across the fabric to output queues on egress interfaces, and then buffering traffic again on the output queues before transmitting the traffic to the next hop. The traditional method of queueing packets on an ingress port is storing traffic destined for different egress ports in the same input queue (buffer).

During periods of congestion, the router might drop packets at the egress port, so the router might spend resources transporting traffic across the switch fabric to an egress port, only to drop that traffic instead of forwarding it. And because input queues store traffic destined for different egress ports, congestion on one egress port could affect traffic on a different egress port, a condition called head-of-line blocking (HOLB).

VOQ architecture takes a different approach:

- Instead of separate physical buffers for input and output queues, the Junos device uses the physical buffers on the ingress pipeline of each Packet Forwarding Engine (PFE) to store traffic for every egress port. Every output queue on an egress port has buffer storage space on every ingress pipeline on all of the PFEs on the router. The mapping of ingress pipeline storage space to output queues is 1-to-1, so each output queue receives buffer space on each ingress pipeline.

- Instead of one input queue containing traffic destined for multiple different output queues (a one-to-many mapping), each output queue has a dedicated VOQ comprised of the input buffers on each PFE that are dedicated to that output queue (a 1-to-1 mapping). This architecture prevents communication between any two ports from affecting another port.

- Instead of storing traffic on a physical output queue until it can be forwarded, a VOQ does not transmit traffic from the ingress port across the fabric to the egress port until the egress port has the resources to forward the traffic. A VOQ is a collection of input queues (buffers) that receive and store traffic destined for one output queue on one egress port. Each output queue on each egress port has its own dedicated VOQ, which consists of all of the input queues that are sending traffic to that output queue.

A VOQ is a collection of input queues (buffers) that receive and store traffic destined for one output queue on one egress port. Each output queue on each egress port has its own dedicated VOQ, which consists of all of the input queues that are sending traffic to that output queue.

VOQ Architecture

A VOQ represents the ingress buffering for a particular output queue. Each of the PFEs uses a specific output queue. The traffic stored on the PFEs comprises the traffic destined for one particular output queue on one port, and is the VOQ for that output queue.

A VOQ is distributed across all of the PFEs in the router that are actively sending traffic to that output queue. Each output queue is the sum of the total buffers assigned to that output queue across all of the

PFEs in the router. So the output queue itself is virtual, not physical, although the output queue is comprised of physical input queues.

Round-Trip Time Buffering

Although there is no output queue buffering during periods of congestion (no long-term storage), there is a small physical output queue buffer on egress line cards to accommodate the round-trip time for traffic to traverse the fabric from ingress to egress. The round-trip time consists of the time it takes the ingress port to request egress port resources, receive a grant from the egress port for resources, and transmit the data across the fabric.

That means if a packet is not dropped at the router ingress, and the router forwards the packet across the fabric to the egress port, the packet will not be dropped and will be forwarded to the next hop. All packet drops take place in the ingress pipeline.

VOQ Advantages

VOQ architecture provides two major advantages:

Eliminate Head-of-Line Blocking

VOQ architecture eliminates head-of-line blocking (HOLB) issues. On non-VOQ devices, HOLB occurs when congestion at an egress port affects a different egress port that is not congested. HOLB occurs when the congested port and the non-congested port share the same input queue on an ingress interface.

VOQ architecture avoids HOLB by creating a different dedicated virtual queue for each output queue on each interface.

Because different egress queues do not share the same input queue, a congested egress queue on one port cannot affect an egress queue on a different port. For the same reason, a congested egress queue on one port cannot affect another egress queue on the same port—each output queue has its own dedicated VOQ composed of ingress interface input queues.

Performing queue buffering at the ingress interface ensures that the router only sends traffic across the fabric to an egress queue if that egress queue is ready to receive that traffic. If the egress queue is not ready to receive traffic, the traffic remains buffered at the ingress interface.

Increase Fabric Efficiency and Utilization

Traditional output queue architecture has some inherent inefficiencies that VOQ architecture addresses.

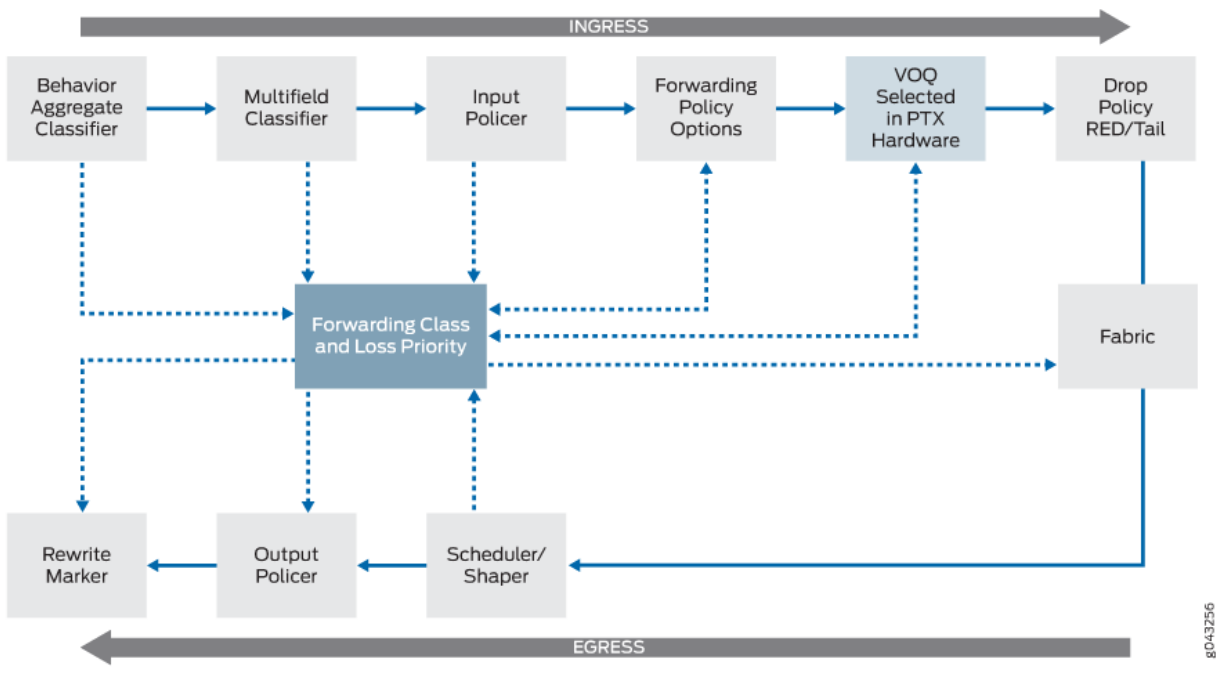
- Packet buffering—Traditional queueing architecture buffers each packet twice in long-term DRAM storage, once at the ingress interface and once at the egress interface. VOQ architecture buffers each packet only once in long-term DRAM storage, at the ingress interface. The fabric is fast enough to be transparent to egress CoS policies, so instead of buffering packets a second time at the egress interface, the router can forward traffic at a rate that does not require deep egress buffers, without affecting the configured egress CoS policies (scheduling).

- Consumption of resources—Traditional queueing architecture sends packets from the ingress interface input queue (buffer), across the fabric, to the egress interface output queue (buffer). At the egress interface, packets might be dropped, even though the router has expended resources transporting the packets across the fabric and storing them in the egress queue. VOQ architecture does not send packets across the fabric to the egress interface until the egress interface is ready to transmit the traffic. This increases system utilization because no resources are wasted transporting and storing packets that are dropped later.

Does VOQ Change How I Configure CoS?

There are no changes to the way you configure the CoS features. Figure 55 on page 793 shows the Junoś CoS components and VOQ selection, illustrating the sequence in which they interact.

Figure 55: Packet Flow Through CoS Components with VOQs



The VOQ selection process is performed by ASICs that use either the behavior aggregate (BA) classifier or the multifield classifier, depending on your configuration, to select one of the eight possible VOQs for an egress port. The VOQs on the ingress buffer data for the egress port based on your CoS configuration.

Although the CoS features do not change, there are some operational differences with VOQ:

- Random early detection (RED) occurs on the ingress PFEs. With devices that support only egress output queuing, RED and associated congestion drops occur on the egress. Performing RED on the ingress saves valuable resources and increases router performance.

  Although RED occurs on the ingress with VOQ, there is no change to how you configure the drop profiles.

- Fabric scheduling is controlled through request and grant control messages. Packets are buffered in ingress VOQs until the egress PFE sends a grant message to the ingress PFE indicating it is ready to receive them. For details on fabric scheduling, see "Fabric Scheduling and Virtual Output Queues on PTX Series Routers" on page 797.

Understand How VOQs Work

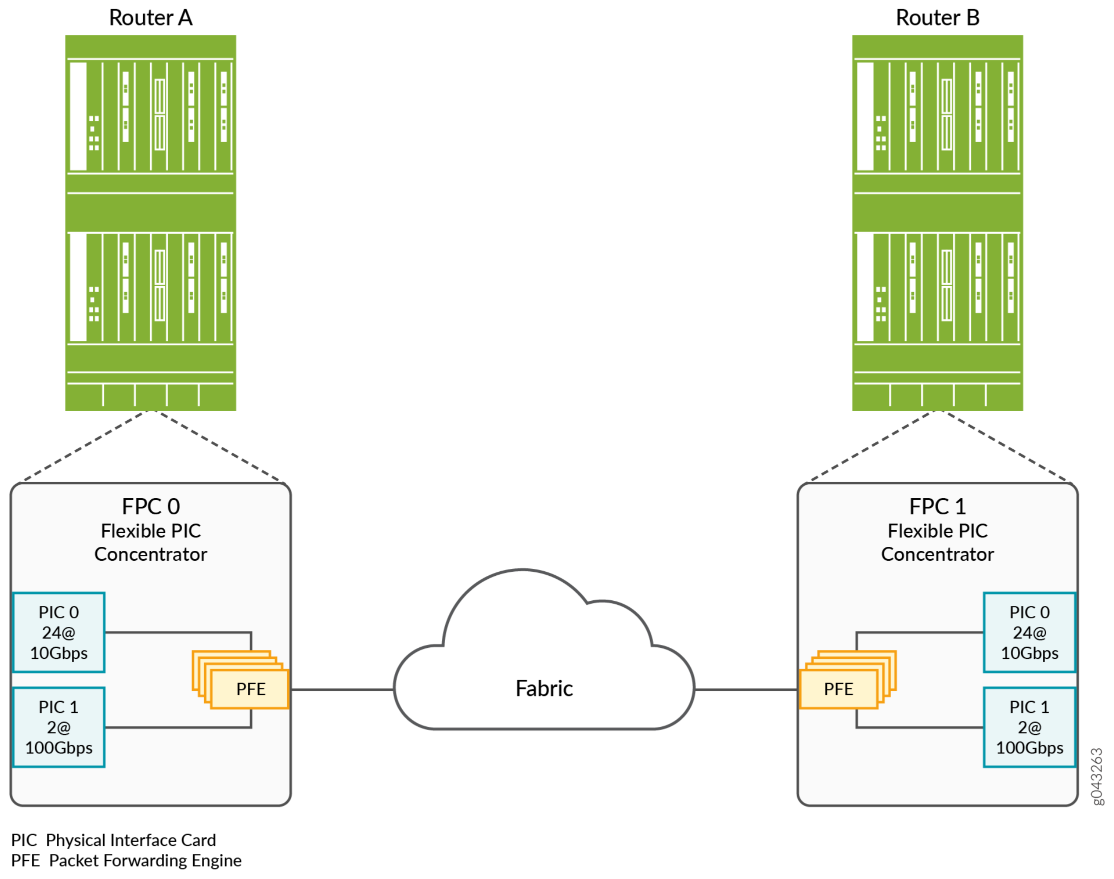
IN THIS SECTION

This topic describes how the VOQ process works on supported Junos devices.

Understanding the Components of the VOQ Process

Figure 56 on page 795 shows the hardware components of the Junos device involved in the VOQ process.

Figure 56: VOQ Components on a Junos Device



These components perform the following functions:

- **Physical Interface Card (PIC)**—Provides the physical connection to various network media types, receiving incoming packets from the network and transmitting outgoing packets to the network.
- **Flexible PIC Concentrator (FPC)**—Connects the PICs installed in it to the other packet transport router components.
- **Packet Forwarding Engine (PFE)**—Provides L2 and L3 packet switching and encapsulation and de-encapsulation. The PFE also provides forwarding, route lookup functions, and manages packet buffering and the queuing of notifications. The PFE receives incoming packets from the PICs installed on the FPC and forwards the packets through the device planes to the appropriate destination port.
- **Output queues**—(Not shown) These output queues are controlled by the CoS scheduler configuration, which establishes how to handle the traffic within the output queues for transmission onto the device fabric. In addition, these output queues control when packets are sent from the VOQs on the ingress to the egress output queues.

Understanding the VOQ Process

Output queues are controlled by the CoS scheduler configuration, which establishes how to handle the traffic within the output queues for transmission onto the fabric. In addition, these output queues control when packets are sent from the VOQs on the ingress to the egress output queues.

For every egress output queue, the VOQ architecture provides *virtual* queues on each and every ingress PFE. These queues are referred to as virtual because the queues physically exist on the ingress PFE *only* when the line card actually has packets enqueued to it.

Figure 57 on page 796 shows three ingress PFEs—PFE0, PFE1, and PFE2. Each ingress PFE provides up to eight VOQs (PFE*n*.e0.q0 through PFE*n*.e0.q7) for the single egress port 0. The egress PFE (PFE*n*) distributes the bandwidth to each ingress VOQ in a round-robin fashion.

For example, egress PFE n's VOQ e0.q0 has 10 Gbps of bandwidth available to it. PFE 0 has an offered load 10 Gbps to e0.qo, PFE1 and PFE2 have an offered load of 1 Gbps to e0.q0. The result is that PFE1 And PFE2 will get 100 percent of their traffic through, while PFE0 will only get 80 percent of its traffic through.

Figure 57: Virtual Output Queues

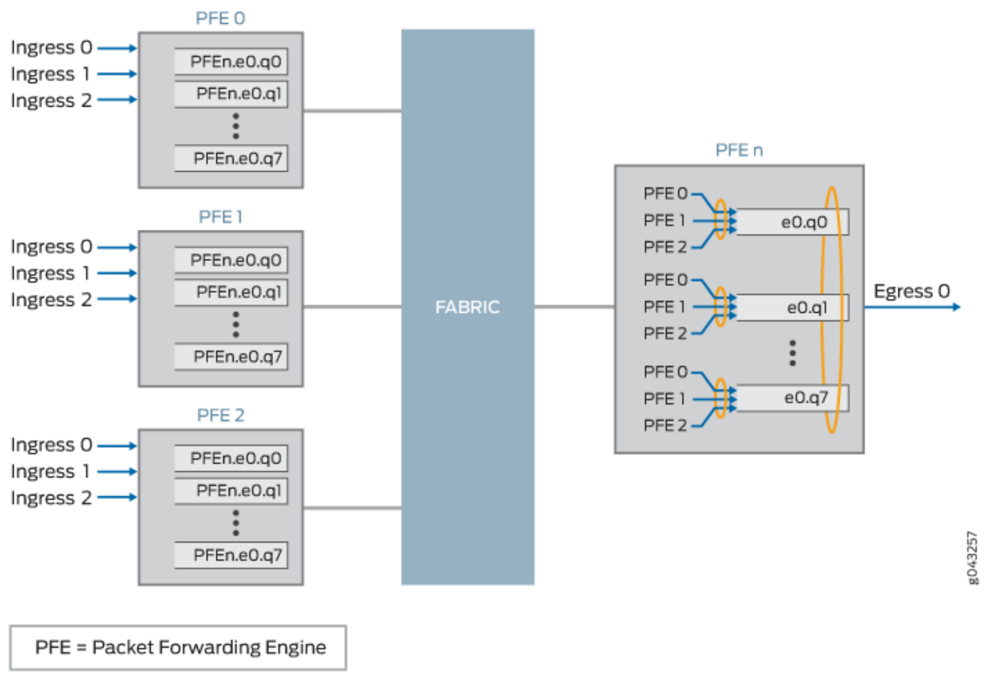
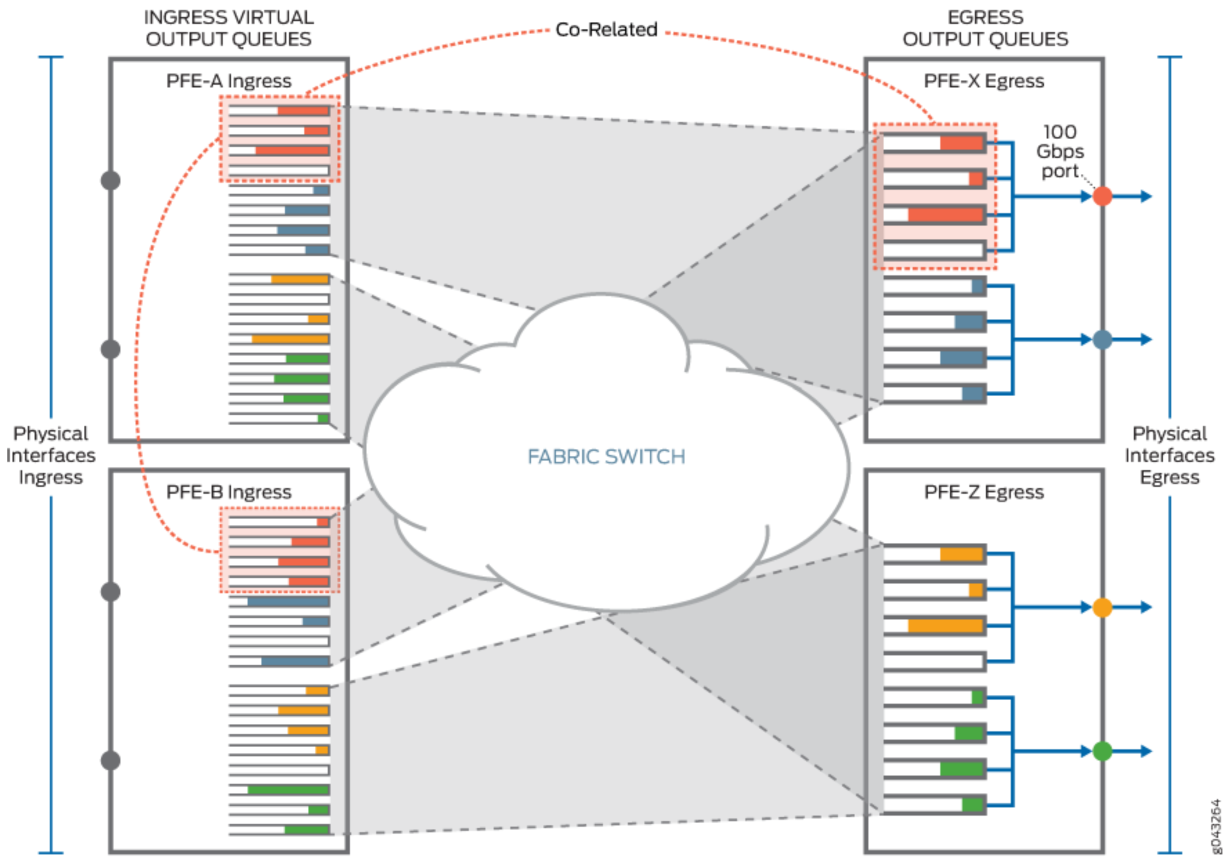


Figure 58 on page 797 illustrates an example of the correlation between the egress output queues and the ingress virtual output queues. On the egress side, PFE-X has a 100 Gbps port, which is configured with four different forwarding classes. As a result, the 100 Gbps egress output port on PFE-X uses four out of eight available egress output queues (as denoted by the four queues highlighted with dashed-orange lines on PFE-X), and the VOQ architecture provides four corresponding *virtual* output queues on *each* ingress PFE (as denoted by the four virtual queues on PFE-A and PFE-B highlighted with dashed-orange lines). The virtual queues on PFE-A and PFE-B exist only when there is traffic to be sent.

Figure 58: Example of VOQ



Fabric Scheduling and VOQs

This topic describes the fabric scheduling process on Junos devices that use VOQs.

VOQs use request and grant messages to control fabric scheduling on Junos devices. The egress Packet Forwarding Engines control data delivery from the ingress VOQs by using request and grant messages. The virtual queues buffer packets on the ingress until the egress Packet Forwarding Engine confirms that it is ready to receive the packets by sending a grant message to the ingress Packet Forwarding Engine.

Figure 59: Fabric Scheduling and VOQs Process

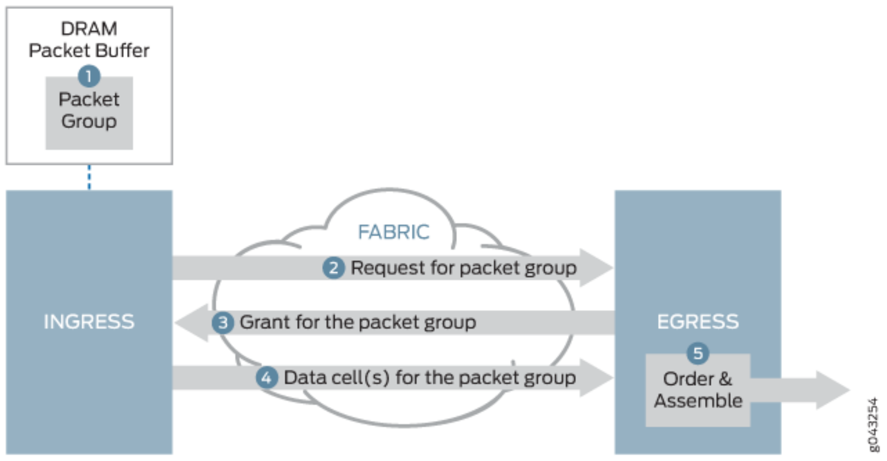

DRAM
Packet Buffer
1
Packet Group
FABRIC
2 Request for packet group
INGRESS
3 Grant for the packet group
EGRESS
5
Order & Assemble
4 Data cell(s) for the packet group
g043254

Figure 59 on page 798 illustrates the fabric scheduling process used by Junos devices with VOQs. When packets arrive at an ingress port, the ingress pipeline stores the packet in the ingress queue associated with the destination output queue. The router makes the buffering decision after performing the packet lookup. If the packet belongs to a forwarding class for which the maximum traffic threshold has been exceeded, the packet might not be buffered and might be dropped. The scheduling process works as follows:

1. An ingress Packet Forwarding Engine receives a packet and buffers it in virtual queues, then groups the packet with other packets destined for the same egress interface and data output queue.

2. The ingress line card Packet Forwarding Engine sends a request, which contains a reference to the packet group, over the fabric to the egress Packet Forwarding Engine.

3. When there is available egress bandwidth, the egress line card grant scheduler responds by sending a bandwidth grant to the ingress line card Packet Forwarding Engine. .

4. When the ingress line card Packet Forwarding Engine receives the grant from the egress line card Packet Forwarding Engine, the ingress Packet Forwarding Engine segments the packet group and sends all of the pieces over the fabric to the egress Packet Forwarding Engine.

5. The egress Packet Forwarding Engine receives the pieces, reassembles the pieces into the packet group, and enqueues individual packets to a data output queue corresponding to the VOQ.

Ingress packets remain in the VOQ on the ingress port input queues until the output queue is ready to accept and forward more traffic.

Under most conditions, the fabric is fast enough to be transparent to egress CoS policies. Therefore the process of forwarding traffic from the ingress pipeline, across the fabric, to egress ports, does not affect the configured CoS policies for the traffic. The fabric only affects CoS policy if there is a fabric failure or an issue of port fairness.

When a packet ingresses and egresses the same Packet Forwarding Engine (local switching), the packet does not traverse the fabric. However, the router uses the same request and grant mechanism to receive egress bandwidth as packets that cross the fabric, so locally routed packets and packets that arrive at a Packet Forwarding Engine after crossing the fabric are treated fairly when the traffic is vying for the same output queue.

Understanding the Packet Forwarding Engine Fairness and VOQ Process

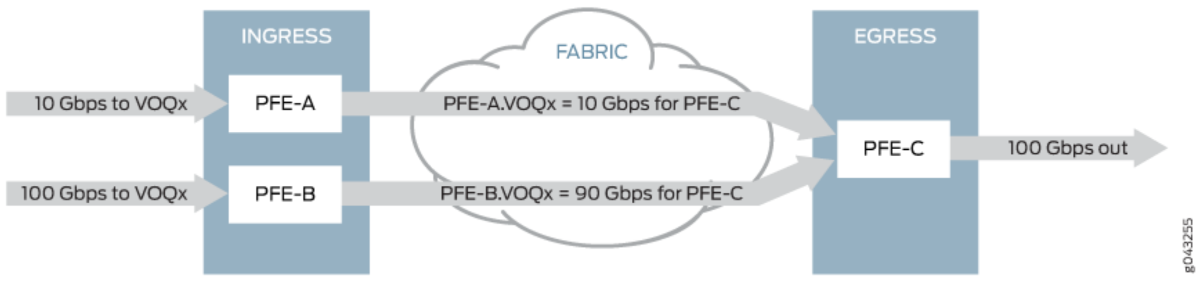
IN THIS SECTION

This topic describes the Packet Forwarding Engine fairness scheme used with VOQ on Junos devices.

Packet Forwarding Engine fairness means that all Packet Forwarding Engines are treated equally from a egress perspective. If multiple egress Packet Forwarding Engines need to transmit data from the same VOQ, the Packet Forwarding Engines are serviced in round-robin fashion. Servicing of VOQs is *not* dependent upon the load that is present at each of the source Packet Forwarding Engines.

Figure 60 on page 799 illustrates the Packet Forwarding Engine fairness scheme used with VOQ in a simple example with three Packet Forwarding Engines. Ingress PFE-A has a single stream of 10 Gbps data destined for VOQ*x* on PFE-C. PFE-B has a single stream of 100 Gbps data also destined for VOQ*x* on PFE-C. On PFE-C, VOQ*x* is serviced by a 100 Gbps interface, and that is the only active VOQ on that interface.

Figure 60: Packet Forwarding Engine Fairness with VOQ Process



In Figure 60 on page 799, we have a total of 110 Gbps of source data destined for a 100 Gbps output interface. As a result, we need to drop 10 Gbps of data. Where does the drop occur and how does this drop affect traffic from PFE-A versus PFE-B?

Because PFE-A and PFE-B are serviced in round-robin fashion by egress PFE-C, all 10 Gbps of traffic from PFE-A makes it through to the egress output port. However, 10 Gbps of data is dropped on PFE-B, allowing only 90 Gbps of data from PFE-B to be sent to PFE-C. So, the 10 Gbps stream has a 0% drop and the 100 Gbps stream has only a 10% drop.

However, if PFE-A and PFE-B were each sourcing 100 Gbps of data, then they would each drop 50 Gbps of data. This is because the egress PFE-C actually controls the servicing and drain rate on the ingress virtual queues using the round-robin algorithm. With the round-robin algorithm, higher bandwidth sources are always penalized when multiple sources are present. The algorithm attempts to make the two sources equal in bandwidth; however, because it cannot raise the bandwidth of the slower source, it drops the bandwidth of the higher source. The round robin algorithm continues this sequence until the sources have equal egress bandwidth.

Each ingress Packet Forwarding Engine provides up to eight VOQs for a single egress port. The egress Packet Forwarding Engine distributes the bandwidth to each ingress VOQ; therefore the VOQs receive equal treatment regardless of their presented load. The drain-rate of a queue is the rate at which a queue is draining. The egress Packet Forwarding Engine divides its bandwidth for each output queue equally across the ingress Packet Forwarding Engines. So, the drain-rate of each ingress Packet Forwarding Engine=Drain-rate of output queue/Number of ingress Packet Forwarding Engines.

Handling Congestion

There are two main types of congestion that can occur:

- Ingress congestion — Occurs when the ingress Packet Forwarding Engine has more offered load than the egress can handle. The ingress congestion case is very similarly to a traditional router in that the queues build-up and once the queues cross their configured threshold, packets are dropped.

- Egress congestion — Occurs when the sum of all the ingress Packet Forwarding Engines exceeds the capability of the egress router. All drops are performed on the ingress Packet Forwarding Engines. However, the size of the ingress queue is attenuated by the queue’s drain-rate (how fast the egress Packet Forwarding Engine is requesting packets). This rate is essentially determined by the rate that requests are being converted in to grants by the egress Packet Forwarding Engine. The egress Packet Forwarding Engine services the request-to-grant conversion in round-robin fashion; it is not dependent on the ingress Packet Forwarding Engines offered load. For instance, if the ingress Packet Forwarding Engine’s drain-rate is half of what it expects it to be (as is the case when 2 ingress Packet Forwarding Engines are presenting an oversubscribed load for the target output queue), then the ingress Packet Forwarding Engine’s reduce the size of this queue to be half of its original size (when it was getting its full drain rate).

VOQ Queue-depth Monitoring

IN THIS SECTION

VOQ queue-depth monitoring, or latency monitoring, measures peak queue occupancy of a VOQ. This feature enables the reporting of peak queue length for a given physical interface for each individual Packet Forwarding Engine.

NOTE: In addition to the peak queue-length data, each queue also maintains drop statistics and time-averaged queue length on the ingress data path. Also, each queue maintains queue transmission statistics on the egress data path.

In a typical deployment scenario that uses strict-priority scheduling, a `HIGH` priority queue can starve `LOW` priority queues. Thus the packets in such `LOW` priority queues can remain longer than desired. You can use this VOQ queue-depth monitoring feature, along with queue transmission statistics, to detect such stalled conditions.

NOTE: You can only enable VOQ queue-depth monitoring on transit WAN interfaces.

To enable VOQ queue-depth monitoring on an interface, you first create a monitoring profile, and then attach that profile to the interface. If you attach a monitoring profile to an aggregated Ethernet (ae-) interface, each member interface has its own dedicated hardware VOQ monitor, unless you also apply the `shared` option to the monitoring profile attached to the ae- interface.

The monitoring profile reports virtual output queue (VOQ) depth individually on each interface. However, given finite number of hardware monitoring profile IDs on a system, this process can quickly consume the maximum supported hardware monitoring profile IDs on large systems. By default, a monitoring profile that you assign to an ae- interface replicates across all members of the ae- interface. Therefore, to conserve monitoring profile IDs, include the `shared` option at the `[set class-of-service interfaces` *`ae-interface`* `monitoring-profile` *`profile-name`*`]` hierarchy level. The configured `shared` option creates only one monitoring profile ID to share across all member interfaces. The option also reports the largest peak on a member interface as the common peak for the ae- interface.

NOTE: You cannot enable the `shared` option on mixed mode ae- interfaces.

Each monitoring profile consists of one or more export filters. An export filter defines a peak queue-length percentage threshold for one or more queues on the physical interface. Once the defined peak queue-length percentage threshold is met for any queue in the export filter, Junos exports the VOQ telemetry data for all queues in the export filter.

NOTE: The queue-depth monitoring data goes out *only* through a telemetry channel. In addition to configuring a monitoring profile (as shown below) you *must* initiate a regular sensor subscription in order for the data to go out. There is no CLI display option.

Configure VOQ Queue-depth Monitoring

Configure VOQ queue-depth monitoring to export queue utilization data. You can use this data to monitor micro-bursts and also assist in identifying stalled transit output queues. To configure VOQ queue-depth monitoring:

1. Configure the monitoring profile.

2. Attach the monitoring profile to an interface.

To configure the monitoring profile:

1. Name the monitoring profile. For example:

```
[edit class-of-service]
set monitoring-profile mp1
```

2. Name an export filter for the montoring profile. For example:

```
[edit class-of-service monitoring-profile mp1]
set export-filters ef1
```

3. Define which queues (0 through 7) belong to the export filter. For example:

```
[edit class-of-service monitoring-profile mp1 export-filters ef1]
set queue [0 1]
```

4. (Optional) Define the threshold peak queue length percentage to export VOQ telemetry data. The default percentage is 0. For example:

```
[edit class-of-service monitoring-profile mp1 export-filters ef1]
set peak-queue-length percent 50
```

5. (Optional) Define one or more other export filters for the monitoring profile. For example:

```
[edit class-of-service monitoring-profile mp1]
set export-filters ef2 queue [2 3]
```

6. Commit your changes.

To attach the monitoring profile to an interface:

1. Attach the monitoring profile to an interface. For example:

```
[edit class-of-service]
set interfaces et-0/0/1 monitoring-profile mp1
set interfaces ae0 monitoring-profile mp1 shared
```

2. Commit your changes.

Check your configuration. For example:

```
[edit class-of-service]
user@host# show
monitoring-profile mp1 {
    export-filters ef1 {
        peak-queue-length {
            percent 50;
        }
        queue [ 0 1 ];
    }
    export-filters ef2 {
        queue [ 2 3 ];
    }
}
interfaces {
    ae0 {
        monitoring-profile mp1 shared;
    }
    et-0/0/1 {
        monitoring-profile mp1;
    }
}
```

Run these show commands to verify your configuration:

```
user@host> show class-of-service interface et-0/0/1
Physical interface: et-0/0/1, Index: 1098
Maximum usable queues: 8, Queues in use: 4
Exclude aggregate overhead bytes: disabled
Logical interface aggregate statistics: disabled
  Scheduler map: default, Index: 0
  Congestion-notification: Disabled
  Monitoring Profile Name: mp1

  Logical interface: et-0/0/1.16386, Index: 1057

user@host> show class-of-service interface ae0
Physical interface: ae0, Index: 7860
Maximum usable queues: 8, Queues in use: 4
Exclude aggregate overhead bytes: disabled
Logical interface aggregate statistics: disabled
  Scheduler map: default, Index: 0
  Congestion-notification: Disabled
  Monitoring Profile Name: mp1 [shared]


user@host> show class-of-service monitoring-profile
Monitoring profile: mp1
  Export filter         Queue Number                    Peak Queue Length
  ef1                    0                                 50%
  ef1                    1                                 50%
  ef2                    2                                 0%
  ef2                    3                                 0%
```

NOTE: As you can see from this example, not setting a `peak-queue-length percent` for an export filter defaults the percentage to 0 percent, as export filter `ef2` shows. This example shows different queues on the physical interface having different peak queue length thresholds for exporting VOQ telemetry data.

RELATED DOCUMENTATION

show interfaces voq

Junos CoS Components Used to Manage Congestion and Control Service Levels | **6**

Packet Flow Through the Junos CoS Process | **16**

Example: Configuring Excess Rate for PTX Series Packet Transport Routers

IN THIS SECTION

You can configure excess rate to customize the distribution of available excess bandwidth among the queues for PTX Series Packet Transport Routers. When excess rate is not configured, the excess bandwidth available is distributed in proportion to the transmit rates allocated to the queues.

Requirements

This example uses the following hardware and software components:

- One PTX Series Packet Transport Router
- Junos OS Release 12.1X48R2 or later

Overview

This set of examples illustrates how you configure schedulers for the PTX Series Packet Transport Router to distribute the remaining bandwidth (excess rate) among the configured queues.

When you configure excess rate, use the following guidelines:

- The `transmit-rate` statements of the configured schedulers can add up to at most 100 percent.
- All queues on the PTX Series Packet Transport Router have the same excess priority. Excess priority configuration is not supported.
- If a strict-high-priority queue is configured and is rate-limited, this queue gets the rate-limited bandwidth first. Then the configured `transmit-rate` value of other queues is met (regardless of queue priority), and finally the excess bandwidth is distributed in proportion to the configured `excess-rate` values.

BEST PRACTICE: We recommend that you configure rate limit on strict-high queues because the other queues might not meet their guaranteed bandwidths. See *transmit-rate*.

Configuration

IN THIS SECTION

To configure excess rate, perform one or more of these tasks:

Configuring Schedulers Without Specifying Excess Rate

CLI Quick Configuration

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

```
set class-of-service schedulers sched_queue_0 transmit-rate percent 20
set class-of-service schedulers sched_queue_1 transmit-rate percent 40
set class-of-service schedulers sched_queue_2 transmit-rate percent 10
set class-of-service schedulers sched_queue_3 transmit-rate percent 10
```

Step-by-Step Procedure

In this example, four queues are configured and each associated scheduler is assigned the indicated transmit rate. Across the four queues, the transmit rate totals to 80 percent. No excess rate is configured. Assuming that each queue has loads greater than or equal to the configured transmit rate, the remaining 20 percent of the bandwidth is distributed in proportion to the configured transmit rates (20:40:10:10):

- sched_queue_0—5% (20% of the guaranteed rate plus 5% of the remaining bandwidth is 25%)
- sched_queue_1—10% (40% of the guaranteed rate plus 10% of the remaining bandwidth is 50%)
- sched_queue_2—2.5% (10% of the guaranteed rate plus 2.5% of the remaining bandwidth is 12.5%)
- sched_queue_3—2.5% (10% of the guaranteed rate plus 2.5% of the remaining bandwidth is 12.5%)

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure the schedulers:

1. Create the scheduler for queue 0:

   ```
   [edit class-of-service]
   user@host# set schedulers sched_queue_0 transmit-rate percent 20
   ```

2. Create the scheduler for queue 1:

```
[edit class-of-service]
user@host# set schedulers sched_queue_1 transmit-rate percent 40
```

3. Create the scheduler for queue 2:

```
[edit class-of-service]
user@host# set schedulers sched_queue_2 transmit-rate percent 10
```

4. Create the scheduler for queue 3:

```
[edit class-of-service]
user@host# set schedulers sched_queue_3 transmit-rate percent 10
```

Results

From configuration mode, confirm your configuration by entering the `show class-of-service schedulers` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
    sched_queue_0 {
        transmit-rate percent 20;
    }
    sched_queue_1 {
        transmit-rate percent 40;
    }
    sched_queue_2 {
        transmit-rate percent 10;
    }
    sched_queue_3 {
        transmit-rate percent 10;
    }
```

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

Configuring Schedulers by Specifying Excess Rate

CLI Quick Configuration

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

```
set class-of-service schedulers sched_queue_0 transmit-rate percent 25
set class-of-service schedulers sched_queue_0 excess-rate percent 25
set class-of-service schedulers sched_queue_1 transmit-rate percent 25
set class-of-service schedulers sched_queue_1 excess-rate percent 50
set class-of-service schedulers sched_queue_2 transmit-rate percent 25
set class-of-service schedulers sched_queue_3 transmit-rate percent 25
```

Step-by-Step Procedure

In this example, four schedulers are configured and each is assigned a transmit rate of 25 percent. Queue 0 is configured with 25 percent and queue 1 with 50 percent of the excess rate. If the offered load through queue 2 is only 10 percent, the remaining bandwidth is distributed as: queue excess rate / total excess rate * remaining bandwidth percentage. If a queue has transmit rate configured but not excess rate, the excess rate for that queue is 1. In this example, the excess rate ratio is 25:50:1:1, which yields the following distribution of the 15 percent remaining bandwidth from queue 2:

- sched_queue_0—4.93% (25 / 76 * 15%)
- sched_queue_1—9.87% (50 / 76 * 15%)
- sched_queue_3—0.197% (1 / 76 * 15%)

When the offered load on queue 2 increases to 25 percent or greater, the other queues get only their configured transmit rates.

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure the schedulers:

1. Create the scheduler for queue 0:

   ```
   [edit class-of-service]
   user@host# set schedulers sched_queue_0 transmit-rate percent 25
   user@host# set schedulers sched_queue_0 excess-rate percent 25
   ```

2. Create the scheduler for queue 1:

   ```
   [edit class-of-service]
   user@host# set schedulers sched_queue_1 transmit-rate percent 25
   user@host# set schedulers sched_queue_1 excess-rate percent 50
   ```

3. Create the scheduler for queue 2:

   ```
   [edit class-of-service]
   user@host# set schedulers sched_queue_2 transmit-rate percent 25
   ```

4. Create the scheduler for queue 3:

   ```
   [edit class-of-service]
   user@host# set schedulers sched_queue_3 transmit-rate percent 25
   ```

Results

From configuration mode, confirm your configuration by entering the `show class-of-service schedulers` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
    sched_queue_0 {
        transmit-rate percent 25;
        excess-rate percent 25;
    }
    sched_queue_1 {
        transmit-rate percent 25;
        excess-rate percent 50;
    }
    sched_queue_2 {
        transmit-rate percent 25;
```

```
    }
    sched_queue_3 {
        transmit-rate percent 25;
    }
```

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

Configuring Schedulers to Control Excess Rate for Non-High-Priority Queues

CLI Quick Configuration

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

```
set class-of-service schedulers sched_queue_0 transmit-rate percent 90
set class-of-service schedulers sched_queue_0 priority high
set class-of-service schedulers sched_queue_1 transmit-rate percent 10
set class-of-service schedulers sched_queue_1 priority low
set class-of-service schedulers sched_queue_2 excess-rate percent 10
set class-of-service schedulers sched_queue_3 excess-rate percent 30
```

Step-by-Step Procedure

In this example, the scheduler for queue 0 is configured to transmit up to 90 percent of traffic if there is enough offered load. When the traffic to queue 0 is less than 90 percent, excess rate is configured to distribute the remaining bandwidth in the ratio 1:1:10:30 (when the offered load on queue 1 is greater than 10 percent), which yields the following distribution of the remaining bandwidth from queue 0:

- sched_queue_1—0.0244 * x% (1 / 41 * remaining bandwidth (x)%)
- sched_queue_2—0.244 * x% (10 / 41 * remaining bandwidth (x)%)
- sched_queue_3—0.732 * x% (30 / 41 * remaining bandwidth (x)%)

NOTE: Although the `transmit-rate` values on queues can add up to at most 100 percent, the `excess-rate` value does not have this restriction because it is a ratio.

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the Junos OS CLI User Guide.

To configure the schedulers:

1. Create the scheduler for queue 0:

```
[edit class-of-service]
user@host# set schedulers sched_queue_0 transmit-rate percent 90
user@host# set schedulers sched_queue_0 priority high
```

2. Create the scheduler for queue 1:

```
[edit class-of-service]
user@host# set schedulers sched_queue_1 transmit-rate percent 10
user@host# set schedulers sched_queue_1 priority low
```

3. Create the scheduler for queue 2:

```
[edit class-of-service]
user@host# set schedulers sched_queue_2 excess-rate percent 10
```

4. Create the scheduler for queue 3:

```
[edit class-of-service]
user@host# set schedulers sched_queue_3 excess-rate percent 30
```

Results

From configuration mode, confirm your configuration by entering the `show class-of-service schedulers` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
sched_queue_0 {
    transmit-rate percent 90;
    priority high;
}
sched_queue_1 {
    transmit-rate percent 10;
    priority low;
}
```

```
    sched_queue_2 {
        excess-rate percent 10;
    }
    sched_queue_3 {
        excess-rate percent 30;
    }
```

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

Verification

IN THIS SECTION

Verifying the Excess Rate Configuration

Purpose

Verify that the excess rate configuration is producing the results you expect.

Action

From operational mode, enter the `show interfaces queue` *`interface`* command for the physical interface to verify.

Meaning

The show command output lists the traffic by queue and forwarding class names. Verify that the Bytes field for active queues on the specified physical interface match the proportions you expect from the excess rate configuration.

RELATED DOCUMENTATION

How Schedulers Define Output Queue Properties | 342

excess-rate

CoS Features and Limitations on PTX Series Routers | 780

Identifying the Source of RED Dropped Packets on PTX Series Routers

This topic describes how to identify the source of random early detection (RED) dropped packets.

Junos OS and PTX Series hardware CoS features use virtual output queues (VOQs) on the ingress to buffer and queue traffic for each egress output queue.

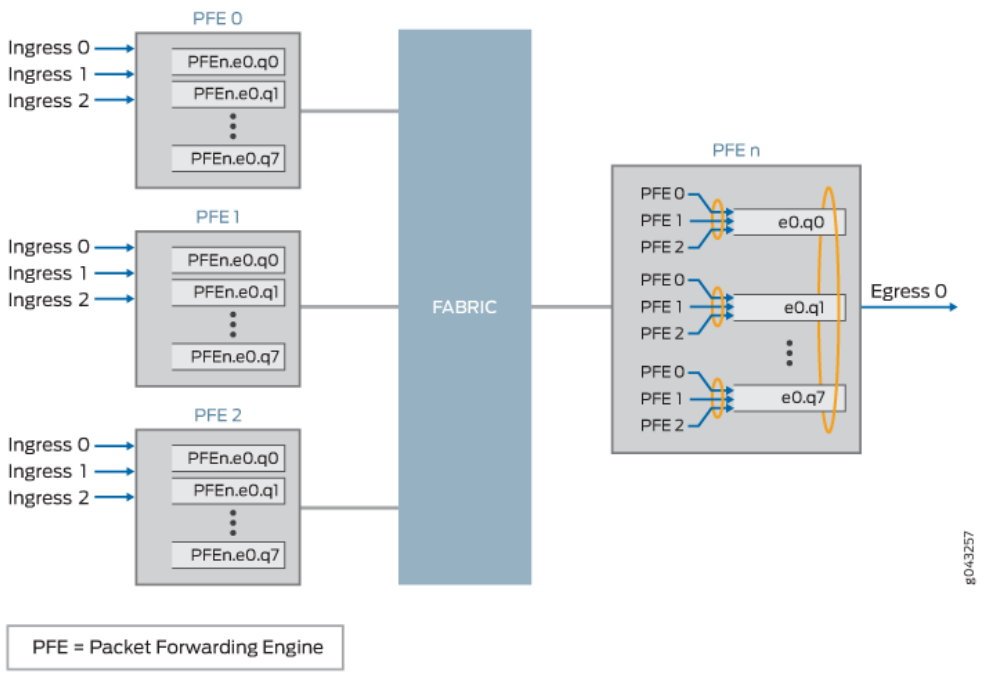
VOQ is a queuing strategy that eliminates congestion drops on the egress and alleviates head-of-line blocking. Head-of-line blocking is a condition in which a queue of packets is blocked from making progress because the packet at the head of the queue is waiting for resources to become available, while other packets behind this packet could be serviced. For example, if the ingress has a single queue for an egress Packet Forwarding Engine, then packets destined for a slow, congested interface can block packets destined for a fast, uncongested interface attached to the same egress Packet Forwarding Engine.

With VOQ, *virtual* queues are maintained on the ingress Packet Forwarding Engines, instead of on the egress Packet Forwarding Engine. However, the scheduling of the ingress virtual output queues is controlled by the egress Packet Forwarding Engine. For every egress output queue (shallow buffer), the VOQ architecture provides *virtual* queues on each and every ingress Packet Forwarding Engine. These queues are referred to as virtual because the queues physically exist on the ingress Packet Forwarding Engine *only* when the line card actually has packets enqueued to it.

Figure 61 on page 815 shows three ingress Packet Forwarding Engines—PFE0, PFE1, and PFE2. Each ingress Packet Forwarding Engine provides up to eight virtual output queues (PFE*n*.e0.q0 through PFE*n*.e0.q7) for the single egress port 0. The egress Packet Forwarding Engine PFE*n* distributes the bandwidth to each ingress VOQ in a round-robin fashion; therefore they will receive equal treatment regardless of their presented load.

For example, egress PFE*n*'s VOQ e0.q0 has 10 Gbps of bandwidth available to it. PFE0 has an offered load of 10 Gbps to e0.qo, whereas PFE1 and PFE2 have an offered load of 1Gbps to e0.q0. The result is that PFE1 and PFE2 get 100 percent of their traffic through, wheras PFE0 gets only 80 percent of its traffic through.

Figure 61: Virtual Output Queuing on PTX Series Routers



When congestion occurs because of the load on the egress output queue, the ingress VOQs corresponding to the egress output queue contain RED dropped packets.

NOTE: For more information about VOQ, see "Understanding Virtual Output Queues on PTX Series Packet Transport Routers" on page 790.

Using the following procedure, you can identify the ingress Packet Forward Engine (in terms of ingress traffic) that is contributing to the egress congestion.

To determine which ingress Packet Forwarding Engine is contributing to the RED dropped packets:

1. Determine whether there are RED dropped packets on the egress link.

   a. Run the `show interfaces queue` *`interface-name`* command on the egress interface.

   ```
   user@host> show interfaces queue et-7/0/0
   ```

   b. In the `show` output, determine whether the interface is experiencing RED dropped packets, by locating the RED-dropped packets field and checking whether its value is greater than zero.

The following example shows RED-dropped statistics for the egress Ethernet interface configured on port 0 of PIC 0, located on the FPC in slot 7.

```
user@host> show interfaces queue et-7/0/0
Physical interface: et-7/0/0, Enabled, Physical link is Up
  Interface index: 206, SNMP ifIndex: 790
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Queue: 0, Forwarding classes: fc0
  Queued:
    Packets              :             539433200              14896082 pps
    Bytes                :           38302319880            8461137824 bps
  Transmitted:
    Packets              :              67108815               1859497 pps
    Bytes                :            4294964160             952062464 bps
    Tail-dropped packets :                     0                     0 pps
    RL-dropped packets   :                     0                     0 pps
    RL-dropped bytes     :                     0                     0 bps
    RED-dropped packets  :             472324385              13036585 pps
    RED-dropped bytes    :           34007355720            7509075360 bps
Queue: 1, Forwarding classes: fc1
  Queued:
    Packets              :             539433555              14877096 pps
    Bytes                :           38302345472            8450201072 bps
  Transmitted:
    Packets              :              67108811               1859498 pps
    Bytes                :            4294963904             952062976 bps
    Tail-dropped packets :                     0                     0 pps
    RL-dropped packets   :                     0                     0 pps
    RL-dropped bytes     :                     0                     0 bps
    RED-dropped packets  :             472324744              13017598 pps
    RED-dropped bytes    :           34007381568            7498138096 bps
Queue: 2, Forwarding classes: fc2
  Queued:
    Packets              :             539433811              14892745 pps
    Bytes                :           38302363728            8459214984 bps
  Transmitted:
    Packets              :              67108833               1859501 pps
    Bytes                :            4294965312             952064512 bps
    Tail-dropped packets :                     0                     0 pps
    RL-dropped packets   :                     0                     0 pps
    RL-dropped bytes     :                     0                     0 bps
    RED-dropped packets  :            472324978             13033244 pps
    RED-dropped bytes    :          34007398416           7507150472 bps
Queue: 3, Forwarding classes: fc3
  Queued:
    Packets              :            539433461             14879323 pps
    Bytes                :          38302338584           8451484208 bps
  Transmitted:
    Packets              :             67108826              1859498 pps
    Bytes                :           4294964864            952062976 bps
    Tail-dropped packets :                    0                    0 pps
    RL-dropped packets   :                    0                    0 pps
    RL-dropped bytes     :                    0                    0 bps
    RED-dropped packets  :            472324635             13019825 pps
    RED-dropped bytes    :          34007373720           7499421232 bps
Queue: 4, Forwarding classes: fc4
  Queued:
    Packets              :            539433755             14884190 pps
    Bytes                :          38302359616           8454286816 bps
  Transmitted:
    Packets              :             67108843              1859508 pps
    Bytes                :           4294965952            952068096 bps
    Tail-dropped packets :                    0                    0 pps
    RL-dropped packets   :                    0                    0 pps
    RL-dropped bytes     :                    0                    0 bps
    RED-dropped packets  :            472324912             13024682 pps
    RED-dropped bytes    :          34007393664           7502218720 bps
Queue: 5, Forwarding classes: fc5
  Queued:
    Packets              :            539433849             14892950 pps
    Bytes                :          38302366384           8459333176 bps
  Transmitted:
    Packets              :             67108843              1859497 pps
    Bytes                :           4294965952            952062464 bps
    Tail-dropped packets :                    0                    0 pps
    RL-dropped packets   :                    0                    0 pps
    RL-dropped bytes     :                    0                    0 bps
    RED-dropped packets  :            472325006             13033453 pps
    RED-dropped bytes    :          34007400432           7507270712 bps
Queue: 6, Forwarding classes: fc6
  Queued:
    Packets              :            539434160             14879808 pps
    Bytes                :          38302388632           8451762856 bps
  Transmitted:
    Packets              :            67108861            1859514 pps
    Bytes                :          4294967104          952071168 bps
    Tail-dropped packets :                   0                  0 pps
    RL-dropped packets   :                   0                  0 pps
    RL-dropped bytes     :                   0                  0 bps
    RED-dropped packets  :           472325299           13020294 pps
    RED-dropped bytes    :         34007421528         7499691688 bps
Queue: 7, Forwarding classes: fc7
  Queued:
    Packets              :           539434364           14900946 pps
    Bytes                :         38302403328         8463940000 bps
  Transmitted:
    Packets              :            67108860            1859496 pps
    Bytes                :          4294967040          952061952 bps
    Tail-dropped packets :                   0                  0 pps
    RL-dropped packets   :                   0                  0 pps
    RL-dropped bytes     :                   0                  0 bps
    RED-dropped packets  :           472325504           13041450 pps
    RED-dropped bytes    :         34007436288         7511878048 bps
```

2. If the interface is experiencing RED dropped packets, run the `show interface voq` *`interface-name`* command on the egress interface that is experiencing the RED dropped packets.

```
user@host> show interfaces voq et-7/0/0 non-zero
```

TIP: When using the `show interfaces voq` command, you can use command filters to help locate the exact queue. For command usage, see *show interfaces voq*.

3. In the `show` output, determine whether the interface is experiencing RED dropped packets.

   The following example shows the count of the ingress RED-dropped packets for the egress Ethernet interface configured on port 0 of PIC 0, located on the FPC in slot 7.

   The sample output shows that the cause of the congestion is the ingress Packet Forwarding Engine PFE 0, on FPC number 4, and the ingress Packet Forwarding Engine PFE 0 on FPC number 6, as denoted by the count of RED-dropped packets.

```
user@host> show interfaces voq et-7/0/0 non-zero
Physical interface: et-7/0/0, Enabled, Physical link is Up
  Interface index: 156, SNMP ifIndex: 699
```

```
Queue: 0, Forwarding classes: q00

  FPC number: 4
    PFE: 0
      RED-dropped packets  :               6834995                96929 pps
      RED-dropped bytes    :            5249276160            595537368 bps

  FPC number: 6
    PFE: 0
      RED-dropped packets  :               6835203                96964 pps
      RED-dropped bytes    :            5249435904            595749256 bps

Queue: 1, Forwarding classes: q01

  FPC number: 4
    PFE: 0
      RED-dropped packets  :               6834998                96967 pps
      RED-dropped bytes    :            5249278464            595766280 bps

  FPC number: 6
    PFE: 0
      RED-dropped packets  :               6835201                96627 pps
      RED-dropped bytes    :            5249434368            593677664 bps

Queue: 2, Forwarding classes: q02

  FPC number: 4
    PFE: 0
      RED-dropped packets  :               6834997                96921 pps
      RED-dropped bytes    :            5249277696            595482712 bps

  FPC number: 6
    PFE: 0
      RED-dropped packets  :               6835205                96827 pps
      RED-dropped bytes    :            5249437440            594907344 bps

Queue: 3, Forwarding classes: q03

  FPC number: 4
    PFE: 0
      RED-dropped packets  :               6834997                96961 pps
      RED-dropped bytes    :            5249277696            595731736 bps
  FPC number: 6
    PFE: 0
      RED-dropped packets  :              6835202                96522 pps
      RED-dropped bytes    :           5249435136            593031808 bps

Queue: 4, Forwarding classes: q04

  FPC number: 4
    PFE: 0
      RED-dropped packets  :              6834995                97021 pps
      RED-dropped bytes    :           5249276160            596099296 bps

  FPC number: 6
    PFE: 0
      RED-dropped packets  :              6835199                96935 pps
      RED-dropped bytes    :           5249432832            595572304 bps

Queue: 5, Forwarding classes: q05

  FPC number: 4
    PFE: 0
      RED-dropped packets  :              6834996                96949 pps
      RED-dropped bytes    :           5249276928            595656872 bps

  FPC number: 6
    PFE: 0
      RED-dropped packets  :              6835204                96899 pps
      RED-dropped bytes    :           5249436672            595348960 bps

Queue: 6, Forwarding classes: q06

  FPC number: 4
    PFE: 0
      RED-dropped packets  :              6835000                97019 pps
      RED-dropped bytes    :           5249280000            596088832 bps

  FPC number: 6
    PFE: 0
      RED-dropped packets  :              6835201                96916 pps
      RED-dropped bytes    :           5249434368            595455624 bps

Queue: 7, Forwarding classes: q07
```

```
FPC number: 4
  PFE: 0
    RED-dropped packets  :          6834999           96929 pps
    RED-dropped bytes    :       5249279232       595536704 bps

FPC number: 6
  PFE: 0
    RED-dropped packets  :          6835202           96941 pps
    RED-dropped bytes    :       5249435136       595609968 bps
```

NOTE: For an aggregate interface, follow the same steps, but you must run the `show interface queue` command on each child link of the aggregate interface to determine which child egress link is experiencing the congestion. Then run the `show interface voq` command on that child link to determine which ingress Packet Forward Engine is contributing to the congestion.

RELATED DOCUMENTATION

Understand Virtual Output Queues | 790

show interfaces voq

show interfaces queue

Configuring Queuing and Shaping on Logical Interfaces on PTX Series Routers

You can enable per-logical interface queuing (sometimes called per-unit scheduling) on on PTX Series Packet Transport Routers and specify a traffic-shaping rate for each logical interface. In conjunction, you can also configure other class-of-service (CoS) features, including classifiers, schedulers, and rewrite rules.

- On PTX5000 Series routers:
  - Only 100-Gigabit Ethernet interfaces are supported.
  - You can configure a maximum of 10 logical interfaces on each physical interface.
  - The maximum shaping rate cannot exceed 100 Gbps for all logical interfaces configured on an interface.

  - You cannot configure per-logical interface queuing on Aggregated Ethernet interfaces.

- On PTX10000 Series routers:

  - Only 400-Gigabit Ethernet interfaces are supported.

  - You can configure a maximum of 6 logical interfaces with per-unit scheduling enabled on each interface (maximum of 2 logical interfaces per interface on PTX10016 routers).

  - The maximum shaping rate cannot exceed 100 Gbps combined for all logical interfaces configured on an interface.

- You cannot configure per-logical interface queuing if you have configured strict priority scheduling on any interface on the router.

- The interface wildcard character (*) is not supported—for example, the following configuration is not supported and should not be used:

```
set class-of-service interface et-* unit * shaping-rate 1g
set class-of-service interface et-* unit * scheduler-map sch0
```

To configure per-logical interface queuing and traffic shaping on PTX Series routers:

1. Enable per-logical interface queuing on the interface:

```
[edit interfaces et-fpc/pic/port]
user@host# set per-unit-scheduler
```

2. Enable the reception and transmission of 8021.q VLAN-tagged frames on the interface:

```
[edit interfaces et-fpc/pic/port]
user@host# set vlan-tagging
```

3. Configure logical interface properties.

   a. Specify a VLAN identifier for each logical interface:

```
[edit interfaces et-fpc/pic/port unit logical-unit-number]
user@host# set vlan-id number
```

b. Specify a protocol family and IP address for each logical interface:

```
[edit interfaces et-fpc/pic/port unit logical-unit-number]
user@host# set family (inet | inet6 | mpls) address ip-address
```

4. Configure per-logical interface traffic shaping by specifying the amount of bandwidth to be allocated to each logical interface:

```
[edit class-of-service interfaces et-fpc/pic/port unit logical-unit-number]
user@host# set shaping-rate rate
```

NOTE: If you do not configure a shaping rate, then the line rate of the physical interface is evenly distributed across all logical interfaces, with each logical interface's portion considered its shaping rate.

5. (Optional) Configure one or more classifiers and apply them to the logical interface.

a. Define one or more behavior aggregate (BA) classifiers:

```
[edit class-of-service classifiers]
user@host# set classifier-type classifier-name
```

b. Define one or more forwarding classes for each classifier:

```
[edit class-of-service classifiers classifier-type classifier-name]
user@host# set forwarding-class forwarding-class-name loss-priority level code-points
[ aliases ] [ bit-patterns ]
```

c. Apply one or more classifiers to the logical interface:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
user@host# set classifiers classifier-type classifier-name
```

6. (Optional) Configure one or more rewrite rules to set CoS bits on outgoing packets.

a. Define one or more rewrite rules:

```
[edit class-of-service rewrite-rules]
user@host# set traffic-type rewrite-rule-name
```

b. Define one or more forwarding classes for each rewrite rule:

```
[edit class-of-service rewrite-rules traffic-type rewrite-rule-name]
user@host# set forwarding-class forwarding-class-name loss-priority level code-points
[ aliases ] [ bit-patterns ]
```

c. Apply one or more rewrite rules to the logical interface for outgoing traffic:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
user@host# set rewrite-rules traffic-type rewrite-rule-name
```

7. (Optional) Configure one or more scheduler maps and apply them to the logical interface. Use a scheduler map to associate the properties of the output queues you define in schedulers with forwarding classes.

a. Specify the name of a scheduler map:

```
[edit class-of-service]
user@host# set scheduler-maps scheduler-map-name
```

b. Specify the name of a forwarding class to associate with the scheduler map:

```
[edit class-of-service scheduler-maps scheduler-map-name]
user@host set forwarding-class forwarding-class-name
```

c. Specify the name of a scheduler configured at the `[edit class-of-service schedulers scheduler-name]` hierarchy level to associate with the scheduler map:

```
[edit class-of-service scheduler-maps scheduler-map-name]
user@host# set schedulers scheduler-name
```

d. Apply the scheduler map to the logical interface:

```
[edit class-of-service]
user@host# set interfaces et-fpc/pic/port unit logical-unit-number scheduler-map scheduler-
map-name
```

8. (Optional) Configure one or more traffical control profiles and apply them to the logical interface. Use a traffic control profile to configure traffic shaping and scheduling profiles for the logical interface.

   a. Specify the name of a traffic control profile:

```
[edit class-of-service]
user@host# set traffic-control-profiles profile-name
```

   b. Specify a shaping rate for the traffic control profile:

```
[edit class-of-service traffic-control-profiles profile-name]
user@host set shaping-rate (percent percentage | rate)
```

   c. Specify a scheduler map for the traffic control profile:

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set scheduler-map map-name
```

   d. Apply the traffic control profile to the logical interface:

```
[edit class-of-service]
user@host# set interfaces et-fpc/pic/port unit logical-unit-number output-traffic-control-
profile profile-name
```

RELATED DOCUMENTATION

Junos CoS Components Used to Manage Congestion and Control Service Levels | 6

Example: Configuring Queuing and Shaping on Logical Interfaces in PTX Series Packet Transport Routers | 826

per-unit-scheduler

shaping-rate

Example: Configuring Queuing and Shaping on Logical Interfaces in PTX Series Packet Transport Routers

IN THIS SECTION

You can enable per-logical interface queuing on interfaces on PTX Series Packet Transport Routers and specify a scheduler map and traffic-shaping rate for each logical interface.

Requirements

This example uses the following hardware and software components:

- Junos OS Evolved Release 24.2R1 or later.
- One PTX10004 router with LC1201 or LC1202 line card.

Overview

This example shows how to configure six VLANs on six logical interfaces, enable per-logical interface queuing, and specify a scheduler map and traffic-shaping rate for each logical interface. The total traffic-shaping rate combined for all logical interfaces cannot exceed 100 Gbps.

Configuration

IN THIS SECTION

CLI Quick Configuration

To configure logical interface queuing and traffic shaping on the PTX10004 router, copy the following commands and paste them into the terminal window of the router:

```
[edit]
set interfaces et-2/0/0 vlan-tagging
set interfaces et-2/0/0 per-unit-scheduler
set interfaces et-2/0/0 unit 0 vlan-id 0
set interfaces et-2/0/0 unit 1 vlan-id 1
set interfaces et-2/0/0 unit 2 vlan-id 2
set interfaces et-2/0/0 unit 3 vlan-id 3
set interfaces et-2/0/0 unit 4 vlan-id 4
set interfaces et-2/0/0 unit 5 vlan-id 5

set class-of-service classifiers dscp dscp_v4 forwarding-class GOLD loss-priority low code-
points 000000
set class-of-service classifiers dscp dscp_v4 forwarding-class SILVER loss-priority low code-
points 000001
set class-of-service classifiers dscp dscp_v4 forwarding-class BRONZE loss-priority low code-
points 000010
set class-of-service classifiers dscp dscp_v4 forwarding-class BE loss-priority low code-points
000011
set class-of-service classifiers dscp dscp_v4 forwarding-class BE1 loss-priority low code-points
000100
set class-of-service classifiers dscp dscp_v4 forwarding-class BE2 loss-priority low code-points
000101
set class-of-service classifiers dscp dscp_v4 forwarding-class BE3 loss-priority low code-points
000110
set class-of-service classifiers dscp dscp_v4 forwarding-class BE4 loss-priority low code-points
000111

set class-of-service forwarding-classes class GOLD queue-num 0
set class-of-service forwarding-classes class SILVER queue-num 1
set class-of-service forwarding-classes class BRONZE queue-num 2
set class-of-service forwarding-classes class BE queue-num 3
set class-of-service forwarding-classes class BE1 queue-num 4
set class-of-service forwarding-classes class BE2 queue-num 5
set class-of-service forwarding-classes class BE3 queue-num 6
set class-of-service forwarding-classes class BE4 queue-num 7

set class-of-service schedulers GOLD transmit-rate percent 20
set class-of-service schedulers SILVER transmit-rate percent 20
set class-of-service schedulers BRONZE transmit-rate percent 10
set class-of-service schedulers BE transmit-rate percent 10

set class-of-service schedulers GOLD priority high
set class-of-service schedulers SILVER priority medium-high
set class-of-service schedulers BRONZE priority medium-low
set class-of-service schedulers BE priority low

set class-of-service scheduler-maps OUT forwarding-class GOLD scheduler GOLD
set class-of-service scheduler-maps OUT forwarding-class SILVER scheduler SILVER
set class-of-service scheduler-maps OUT forwarding-class BRONZE scheduler BRONZE
set class-of-service scheduler-maps OUT forwarding-class BE scheduler BE
set class-of-service scheduler-maps OUT forwarding-class BE1 scheduler BE
set class-of-service scheduler-maps OUT forwarding-class BE2 scheduler BE
set class-of-service scheduler-maps OUT forwarding-class BE3 scheduler BE
set class-of-service scheduler-maps OUT forwarding-class BE4 scheduler BE

set class-of-service interfaces et-2/0/0 unit 0 scheduler-map OUT
set class-of-service interfaces et-2/0/0 unit 1 scheduler-map OUT
set class-of-service interfaces et-2/0/0 unit 2 scheduler-map OUT
set class-of-service interfaces et-2/0/0 unit 3 scheduler-map OUT
set class-of-service interfaces et-2/0/0 unit 4 scheduler-map OUT
set class-of-service interfaces et-2/0/0 unit 5 scheduler-map OUT

set class-of-service interfaces et-2/0/0 unit 0 shaping-rate 20g
set class-of-service interfaces et-2/0/0 unit 1 shaping-rate 16g
set class-of-service interfaces et-2/0/0 unit 2 shaping-rate 16g
set class-of-service interfaces et-2/0/0 unit 3 shaping-rate 16g
set class-of-service interfaces et-2/0/0 unit 4 shaping-rate 16g
set class-of-service interfaces et-2/0/0 unit 5 shaping-rate 16g
```

Procedure

Step-by-Step Procedure

To configure the PTX10004 router:

1. Enable the reception and transmission of 8021.q VLAN-tagged frames on the interface:

```
[edit interfaces]
user@host# set et-2/0/0 vlan-tagging
```

2. Enable per-logical interface scheduling on the interface:

```
[edit interfaces]
user@host# set et-2/0/0 per-unit-scheduling
```

3. Specify a VLAN identifier for each logical interface:

```
[edit interfaces]
user@host# set et-2/0/0 unit 0 vlan-id 0
user@host# set et-2/0/0 unit 1 vlan-id 1
user@host# set et-2/0/0 unit 2 vlan-id 2
user@host# set et-2/0/0 unit 3 vlan-id 3
user@host# set et-2/0/0 unit 4 vlan-id 4
user@host# set et-2/0/0 unit 5 vlan-id 5
```

4. Define classifiers to assign incoming packets to forwarding classes based on each packet's code point value and loss priority:

```
[edit class-of-service classifiers]
user@host# set dscp dscp_v4 forwarding-class GOLD loss-priority low code-points 000000
user@host# set dscp dscp_v4 forwarding-class SILVER loss-priority low code-points 000001
user@host# set dscp dscp_v4 forwarding-class BRONZE loss-priority low code-points 000010
user@host# set dscp dscp_v4 forwarding-class BE loss-priority low code-points 000011
user@host# set dscp dscp_v4 forwarding-class BE1 loss-priority low code-points 000100
user@host# set dscp dscp_v4 forwarding-class BE2 loss-priority low code-points 000101
user@host# set dscp dscp_v4 forwarding-class BE3 loss-priority low code-points 000110
user@host# set dscp dscp_v4 forwarding-class BE4 loss-priority low code-points 000111
```

5. Assign each forwarding class to a queue:

```
[edit class-of-service forwarding-classes]
user@host# set class GOLD queue-num 0
user@host# set class SILVER queue-num 1
```

```
user@host# set class BRONZE queue-num 2
user@host# set class BE queue-num 3
user@host# set class BE1 queue-num 4
user@host# set class BE2 queue-num 5
user@host# set class BE3 queue-num 6
user@host# set class BE4 queue-num 7
```

6. Define schedulers, each with their own transmit rate and priority:

```
[edit class-of-service schedulers]
user@host# set GOLD transmit-rate percent 20
user@host# set SILVER transmit-rate percent 20
user@host# set BRONZE transmit-rate percent 10
user@host# set BE transmit-rate percent 10

user@host# set GOLD priority high
user@host# set SILVER priority medium-high
user@host# set BRONZE priority medium-low
user@host# set BE priority low
```

7. Define a scheduler map that maps a scheduler to each forwarding class (and therefore queue):

```
[edit class-of-service scheduler-maps]
user@host# set OUT forwarding-class GOLD scheduler GOLD
user@host# set OUT forwarding-class SILVER scheduler SILVER
user@host# set OUT forwarding-class BRONZE scheduler BRONZE
user@host# set OUT forwarding-class BE scheduler BE
user@host# set OUT forwarding-class BE1 scheduler BE
user@host# set OUT forwarding-class BE2 scheduler BE
user@host# set OUT forwarding-class BE3 scheduler BE
user@host# set OUT forwarding-class BE4 scheduler BE
```

8. Assign the scheduler map to each logical interface:

```
[edit class-of-service interfaces]
user@host# set et-2/0/0 unit 0 scheduler-map OUT
user@host# set et-2/0/0 unit 1 scheduler-map OUT
user@host# set et-2/0/0 unit 2 scheduler-map OUT
user@host# set et-2/0/0 unit 3 scheduler-map OUT
```

```
user@host# set et-2/0/0 unit 4 scheduler-map OUT
user@host# set et-2/0/0 unit 5 scheduler-map OUT
```

NOTE: Keep in mind, with per-logical interface queueing, each logical interface has eight queues and can have its own scheduler map for its own service profile.

9. Specify the amount of bandwidth to allocate to each logical interface:

```
[edit class-of-service interfaces]
user@host# set et-2/0/0 unit 0 shaping-rate 20g
user@host# set et-2/0/0 unit 1 shaping-rate 16g
user@host# set et-2/0/0 unit 2 shaping-rate 16g
user@host# set et-2/0/0 unit 3 shaping-rate 16g
user@host# set et-2/0/0 unit 4 shaping-rate 16g
user@host# set et-2/0/0 unit 5 shaping-rate 16g
```

Results

Confirm your results by entering the `show interfaces` and `show class-of-service` commands:

```
user@host# show interfaces
et-2/0/0  {
    per-unit-scheduling
    vlan-tagging;
    unit 0 {
        vlan-id 0;
    }
    unit 1 {
        vlan-id 1;
    }
    unit 2 {
        vlan-id 2;
    }
    unit 3 {
        vlan-id 3;
    }
    unit 4 {
        vlan-id 4;
    }
    unit 5 {
        vlan-id 5;
    }
}
```

```
user@host# **show class-of-service**
classifiers {
    dscp dscp_v4 {
        forwarding-class BE {
            loss-priority low code-points 000011;
        }
        forwarding-class BE1 {
            loss-priority low code-points 000100;
        }
        forwarding-class BE2 {
            loss-priority low code-points 000101;
        }
        forwarding-class BE3 {
            loss-priority low code-points 000110;
        }
        forwarding-class BE4 {
            loss-priority low code-points 000111;
        }
        forwarding-class BRONZE {
            loss-priority low code-points 000010;
        }
        forwarding-class GOLD {
            loss-priority low code-points 000000;
        }
        forwarding-class SILVER {
            loss-priority low code-points 000001;
        }
    }
}
forwarding-classes {
    class BE queue-num 3;
    class BE1 queue-num 4;
    class BE2 queue-num 5;
    class BE3 queue-num 6;
    class BE4 queue-num 7;
    class BRONZE queue-num 2;
    class GOLD queue-num 0;
    class SILVER queue-num 1;
}interfaces {
    et-2/0/0 {
        unit 0 {
            scheduler-map OUT;
            shaping-rate 20g;
        }
        unit 1 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
        unit 2 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
        unit 3 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
        unit 4 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
        unit 5 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
    }
}
scheduler-maps {
    OUT {
        forwarding-class BE scheduler BE;
        forwarding-class BE1 scheduler BE;
        forwarding-class BE2 scheduler BE;
        forwarding-class BE3 scheduler BE;
        forwarding-class BE4 scheduler BE;
        forwarding-class BRONZE scheduler BRONZE;
        forwarding-class GOLD scheduler GOLD;
        forwarding-class SILVER scheduler SILVER;
    }
}
schedulers {
    BE {
        transmit-rate percent 10;
        priority low;
    }
    BRONZE {
        transmit-rate percent 10;
        priority medium-low;
    }
    GOLD {
        transmit-rate percent 20;
        priority high;
    }
    SILVER {
        transmit-rate percent 20;
        priority medium-high;
    }
}
```

RELATED DOCUMENTATION

Configuring Queuing and Shaping on Logical Interfaces on PTX Series Routers | 821

per-unit-scheduler

shaping-rate

Example: Configuring Strict-Priority Scheduling on a PTX Series Router

IN THIS SECTION

This example shows how to configure *strict-priority scheduling* for a physical interface on a PTX Series router.

Requirements

This example uses the following hardware and software components:

- One PTX Series Packet Transport Router
- One or more routers that provide input packets and receive output packets
- Any supported Junos release

Overview

IN THIS SECTION

This example illustrates how you configure strict-priority scheduling for a physical interface on a PTX Series router to perform processing of queues in strict-priority order. Queues in the guaranteed region with the same priority are processed in round-robin fashion. Queues in the excess region are processed based on the WRR algorithm.

When you configure strict-priority scheduling, use the following guidelines:

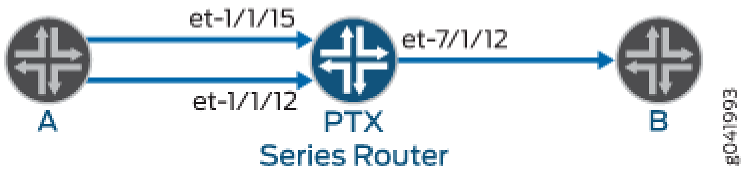
- The configured `transmit-rate` does not affect the queue drain rate because packets are processed in order of queue priority.
- You can configure only one queue with `strict-high` priority at the `[edit class-of-service schedulers` *`scheduler-name`* `priority]` hierarchy level.
- You cannot configure both `transmit-rate exact` and `strict-high` priority at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level.
- You cannot configure `scheduler-map` or `shaping-rate` on an interface where you configure an output traffic control profile.
- You cannot configure `transmit-rate` on a queue with `low` priority or the commit will fail.

NOTE: If a strict-high priority queue is constantly loaded to 100 percent of traffic capacity, other queues are starved. Queue starvation can cause the interface hardware to generate critical interrupts.

Topology

In Figure 62 on page 836, the PTX Series router has inputs from Router A, et-1/1/15 and et-1/1/12, and an output to Router B, et-7/1/12. This example configures classification on the two ingress Interfaces and configures strict-priority scheduling on the egress interface.

Figure 62: Topology for Configuring Strict-Priority Scheduling on a PTX Series Router



Configuration

IN THIS SECTION

Configuring Strict-Priority Scheduling

CLI Quick Configuration

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

```
set class-of-service classifiers dscp cls forwarding-class be loss-priority high code-points
000000
set class-of-service classifiers dscp cls forwarding-class ef loss-priority medium-high code-
```

```
points 000001
set class-of-service classifiers dscp cls forwarding-class af loss-priority medium-low code-
points 000010
set class-of-service classifiers dscp cls forwarding-class nc loss-priority low code-points
000011
set class-of-service classifiers dscp cls forwarding-class af11 loss-priority low code-points
000100
set class-of-service classifiers dscp cls forwarding-class af12 loss-priority low code-points
000101
set class-of-service classifiers dscp cls forwarding-class af13 loss-priority low code-points
000110
set class-of-service classifiers dscp cls forwarding-class nc2 loss-priority low code-points
000111
set class-of-service forwarding-classes queue 0 be
set class-of-service forwarding-classes queue 1 ef
set class-of-service forwarding-classes queue 2 af
set class-of-service forwarding-classes queue 3 nc
set class-of-service forwarding-classes queue 4 af11
set class-of-service forwarding-classes queue 5 af12
set class-of-service forwarding-classes queue 6 af13
set class-of-service forwarding-classes queue 7 nc2
set class-of-service traffic-control-profiles tcp1 scheduler-map sch0
set class-of-service traffic-control-profiles tcp1 strict-priority-scheduler
set class-of-service interfaces et-1/1/12 unit 0 classifiers dscp cls
set class-of-service interfaces et-1/1/15 unit 0 classifiers dscp cls
set class-of-service interfaces et-7/1/12 output-traffic-control-profile tcp1
set class-of-service scheduler-maps sch0 forwarding-class be scheduler be_sch
set class-of-service scheduler-maps sch0 forwarding-class ef scheduler ef_sch
set class-of-service scheduler-maps sch0 forwarding-class af scheduler af_sch
set class-of-service scheduler-maps sch0 forwarding-class nc scheduler nc_sch
set class-of-service scheduler-maps sch0 forwarding-class af11 scheduler af11_sch
set class-of-service scheduler-maps sch0 forwarding-class af12 scheduler af12_sch
set class-of-service scheduler-maps sch0 forwarding-class af13 scheduler af13_sch
set class-of-service scheduler-maps sch0 forwarding-class nc2 scheduler nc2_sch
set class-of-service schedulers be_sch transmit-rate percent 60
set class-of-service schedulers be_sch priority high
set class-of-service schedulers ef_sch transmit-rate percent 5
set class-of-service schedulers ef_sch priority medium-high
set class-of-service schedulers af_sch transmit-rate percent 5
set class-of-service schedulers af_sch priority high
set class-of-service schedulers nc_sch transmit-rate percent 5
set class-of-service schedulers nc_sch priority strict-high
set class-of-service schedulers af11_sch transmit-rate percent 5
set class-of-service schedulers af11_sch priority high
set class-of-service schedulers af12_sch transmit-rate percent 5
set class-of-service schedulers af12_sch priority medium-high
set class-of-service schedulers af13_sch transmit-rate percent 5
set class-of-service schedulers af13_sch priority medium-low
set class-of-service schedulers nc2_sch priority low
```

Step-by-Step Procedure

In this example, eight schedulers are configured based on eight DSCP classifier configurations. Each associated scheduler is assigned a priority and transmit rate, although the transmit rate is ignored by the strict-priority scheduler. The scheduler map sch0 is configured with the mapping of forwarding classes to schedulers. Within the traffic control profile tcp1, the scheduler map and the strict-priority scheduler feature are configured. Two input interfaces on the PTX Series router, et-1/1/12 and et-1/1/15, are configured with the DSCP classifiers. The output traffic control profile on et-7/1/12 is configured with the traffic control profile tcp1.

The following example requires you to navigate various levels in the configuration hierarchy. For instructions on how to do that, see *Using the CLI Editor in Configuration Mode* in the CLI User Guide.

To configure strict-priority scheduling:

1. Configure the DSCP forwarding classes.

```
[edit class-of-service dscp cls]
user@host# set forwarding-class be loss-priority high code-points 000000
user@host# set forwarding-class ef loss-priority medium-high code-points 000001
user@host# set forwarding-class af loss-priority medium-low code-points 000010
user@host# set forwarding-class nc loss-priority low code-points 000011
user@host# set forwarding-class af11 loss-priority low code-points 000100
user@host# set forwarding-class af12 loss-priority low code-points 000101
user@host# set forwarding-class af13 loss-priority low code-points 000110
user@host# set forwarding-class nc2 loss-priority low code-points 000111
```

2. Configure the mapping of queues to forwarding classes.

```
[edit class-of-service forwarding-classes]
user@host# set queue 0 be
user@host# set queue 1 ef
user@host# set queue 2 af
user@host# set queue 3 nc
```

```
user@host# set queue 4 af11
user@host# set queue 5 af12
user@host# set queue 6 af13
user@host# set queue 7 nc2
```

3. Configure the transmit rate and priority for each scheduler.

   Although you can configure a transmit rate, the value that you configure is overridden by the strict-priority scheduler.

```
[edit class-of-service]
user@host# set schedulers be_sch transmit-rate percent 60
user@host# set schedulers be_sch priority high
user@host# set schedulers ef_sch transmit-rate percent 5
user@host# set schedulers ef_sch priority medium-high
user@host# set schedulers af_sch transmit-rate percent 5
user@host# set schedulers af_sch priority high
user@host# set schedulers nc_sch transmit-rate percent 5
user@host# set schedulers nc_sch priority strict-high
user@host# set schedulers af11_sch transmit-rate percent 5
user@host# set schedulers af11_sch priority high
user@host# set schedulers af12_sch transmit-rate percent 5
user@host# set schedulers af12_sch priority medium-high
user@host# set schedulers af13_sch transmit-rate percent 5
user@host# set schedulers af13_sch priority medium-low
user@host# set schedulers nc2_sch priority low
```

4. Configure the scheduler map with the mapping of forwarding classes to schedulers.

```
[edit class-of-service scheduler-maps sch0]
user@host# set forwarding-class be scheduler be_sch
user@host# set forwarding-class ef scheduler ef_sch
user@host# set forwarding-class af scheduler af_sch
user@host# set forwarding-class nc scheduler nc_sch
user@host# set forwarding-class af11 scheduler af11_sch
user@host# set forwarding-class af12 scheduler af12_sch
user@host# set forwarding-class af13 scheduler af13_sch
user@host# set forwarding-class nc2 scheduler nc2_sch
```

5. Configure the traffic control profile to do strict-priority scheduling and define the scheduler map to use.

```
[edit class-of-service traffic-control-profiles tcp1]
user@host# set scheduler-map sch0
user@host# set strict-priority-scheduler
```

6. Apply the classifiers to the input interfaces, and the traffic control profile to the output interface.

```
[edit class-of-service interfaces]
user@host# set et-1/1/12 unit 0 classifiers dscp cls
user@host# set et-1/1/15 unit 0 classifiers dscp cls
user@host# set et-7/1/12 output-traffic-control-profile tcp1
```

Results

From configuration mode, confirm your configuration by entering the `show class-of-service` command. If the output does not display the intended configuration, repeat the configuration instructions in this example to correct it.

```
[edit]
user@host# show class-of-service
forwarding-classes {
    queue 0 be;
    queue 1 ef;
    queue 2 af;
    queue 3 nc;
    queue 4 af11;
    queue 5 af12;
    queue 6 af13;
    queue 7 nc2;
}
interfaces {
    et-1/1/12 {
        unit 0 {
            classifiers {
                dscp cls;
            }
        }
    }
    et-1/1/15 {
        unit 0 {
            classifiers {
                dscp cls;
            }
        }
    }
    et-7/1/12 {
        output-traffic-control-profile tcp1;
    }
}
scheduler-maps {
    sch0 {
        forwarding-class be scheduler be_sch;
        forwarding-class ef scheduler ef_sch;
        forwarding-class af scheduler af_sch;
        forwarding-class nc scheduler nc_sch;
        forwarding-class af11 scheduler af11_sch;
        forwarding-class af12 scheduler af12_sch;
        forwarding-class af13 scheduler af13_sch;
        forwarding-class nc2 scheduler nc2_sch;
    }
}
schedulers {
    be_sch {
        transmit-rate percent 60;
        priority high;
    }
    ef_sch {
        transmit-rate percent 5;
        priority medium-high;
    }
    af_sch {
        transmit-rate percent 5;
        priority high;
    }
    nc_sch {
        transmit-rate percent 5;
        priority strict-high;
    }
    af11_sch {
        transmit-rate percent 5;
        priority high;
    }
    af12_sch {
        transmit-rate percent 5;
        priority medium-high;
    }
    af13_sch {
        transmit-rate percent 5;
        priority medium-low;
    }
    nc2_sch {
        priority low;
    }
}
traffic-control-profiles {
    tcp1 {
        scheduler-map sch0;
        strict-priority-scheduler;
    }
}
```

Verification

IN THIS SECTION

Verifying Strict-Priority Scheduling

Purpose

Verify that the strict-priority scheduling configuration is producing the results you expect.

Action

From operational mode, enter the `show interfaces queue interface-name` *`interface-name`* command and select the output physical interface to verify.

```
user@host> show interfaces queue interface-name et-7/1/12

Physical interface: et-7/1/12, Enabled, Physical link is Up
  Interface index: 231, SNMP ifIndex: 612
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Queue: 0, Forwarding classes: be
  Queued:
    Packets              :               394488                131507 pps
    Bytes                :            591732000            1578084848 bps
  Transmitted:
    Packets              :               394488                131507 pps
    Bytes                :            591732000            1578084848 bps
    Tail-dropped packets :                    0                     0 pps
    RL-dropped packets   :                    0                     0 pps
    RL-dropped bytes     :                    0                     0 bps
    RED-dropped packets  :                    0                     0 pps
    RED-dropped bytes    :                    0                     0 bps
Queue: 1, Forwarding classes: ef
  Queued:
    Packets              :               234498                 82115 pps
    Bytes                :            352963584             988886784 bps
  Transmitted:
    Packets              :                82425                 27551 pps
    Bytes                :            123637500             330618176 bps
    Tail-dropped packets :                    0                     0 pps
    RL-dropped packets   :                    0                     0 pps
    RL-dropped bytes     :                    0                     0 bps
    RED-dropped packets  :               152073                 54564 pps
    RED-dropped bytes    :            229326084             658268608 bps
Queue: 2, Forwarding classes: af
  Queued:
    Packets              :               345175                115068 pps
    Bytes                :            517762500            1380824240 bps
  Transmitted:
    Packets              :               345175                115068 pps
    Bytes                :            517762500            1380824240 bps
    Tail-dropped packets :                    0                     0 pps
    RL-dropped packets   :                    0                     0 pps
    RL-dropped bytes     :                    0                     0 bps
    RED-dropped packets  :                    0                     0 pps
    RED-dropped bytes    :                    0                     0 bps
Queue: 3, Forwarding classes: nc
  Queued:
    Packets              :               986224                328769 pps
    Bytes                :           1479336000            3945236360 bps
  Transmitted:
    Packets              :               986224                328769 pps
    Bytes                :           1479336000            3945236360 bps
    Tail-dropped packets :                    0                     0 pps
    RL-dropped packets   :                    0                     0 pps
    RL-dropped bytes     :                    0                     0 bps
    RED-dropped packets  :                    0                     0 pps
    RED-dropped bytes    :                    0                     0 bps
Queue: 4, Forwarding classes: af11
  Queued:
    Packets              :               493110                164383 pps
    Bytes                :            739665000            1972606056 bps
  Transmitted:
    Packets              :               493110                164383 pps
    Bytes                :            739665000            1972606056 bps
    Tail-dropped packets :                    0                     0 pps
    RL-dropped packets   :                    0                     0 pps
    RL-dropped bytes     :                    0                     0 bps
    RED-dropped packets  :                    0                     0 pps
    RED-dropped bytes    :                    0                     0 bps
Queue: 5, Forwarding classes: af12
  Queued:
    Packets              :               461830                164375 pps
    Bytes                :            695777416            1981272728 bps
  Transmitted:
    Packets              :                82778                 27543 pps
    Bytes                :            124167000             330521208 bps
    Tail-dropped packets :                    0                     0 pps
    RL-dropped packets   :                    0                     0 pps
    RL-dropped bytes     :                    0                     0 bps
    RED-dropped packets  :               379052                136832 pps
    RED-dropped bytes    :            571610416            1650751520 bps
Queue: 6, Forwarding classes: af13
  Queued:
    Packets              :                 462258               164556 pps
    Bytes                :              696421280           1983445256 bps
  Transmitted:
    Packets              :                  82973                27637 pps
    Bytes                :              124459500            331648480 bps
    Tail-dropped packets :                      0                    0 pps
    RL-dropped packets   :                      0                    0 pps
    RL-dropped bytes     :                      0                    0 bps
    RED-dropped packets  :                 379285               136919 pps
    RED-dropped bytes    :              571961780           1651796776 bps
Queue: 7, Forwarding classes: nc2
  Queued:
    Packets              :                 227750                82215 pps
    Bytes                :              343447000            991843712 bps
  Transmitted:
    Packets              :                      0                    0 pps
    Bytes                :                      0                    0 bps
    Tail-dropped packets :                      0                    0 pps
    RL-dropped packets   :                      0                    0 pps
    RL-dropped bytes     :                      0                    0 bps
    RED-dropped packets  :                 227750                82215 pps
    RED-dropped bytes    :              343447000            991843712 bps
```

Meaning

The `show` command output lists the traffic by queue and forwarding class names. The Bytes field under the Transmitted field for each queue shows the actual bytes transmitted.

From the sample output, you can see that the strict-high queue gets the highest priority and transmits without drops. The high-priority queues are then transmitted. The medium-high and medium-low priority queues are processed in a round-robin fashion. The low-priority queue is starved.

Keep in mind the following conditions that apply to strict-priority scheduling:

- If the traffic on the output interface is undersubscribed, no queue should show dropped traffic.
- The strict-high queue is processed first, followed by the high-priority queues (in a round-robin fashion), and finally all remaining queues in the guaranteed region (in a round-robin fashion).
- If the ingress traffic exceeds the capacity of the output interface, the queues are processed in strict-priority order.
- Queues in the excess region are processed based on the WRR algorithm.

RELATED DOCUMENTATION

Understanding Scheduling on PTX Series Routers | 786

Example: Configuring Excess Rate for PTX Series Packet Transport Routers | 805

Understanding CoS CLI Configuration Statements on PTX Series Routers | 855

How Schedulers Define Output Queue Properties | 342

Configuring Schedulers | 348

excess-rate

strict-priority-scheduler

traffic-control-profiles

CoS Support on EVPN VXLANs

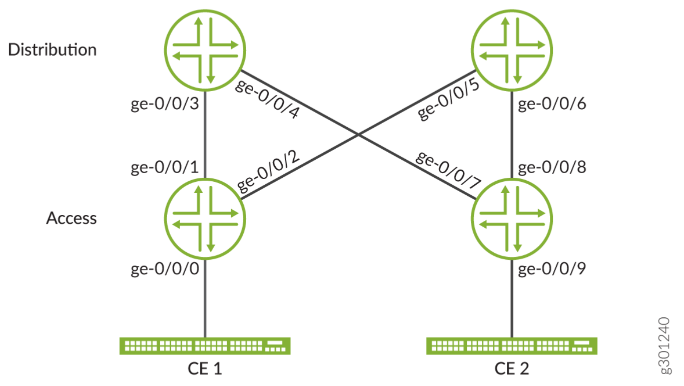
IN THIS SECTION

You can configure class of service (CoS) features on VXLAN interfaces. VXLAN traffic from different tenants traverses network boundaries over the same physical underlay network. To ensure fairness in the treatment of traffic for all tenants in the VXLAN, and to prioritize higher priority traffic, apply CoS features to the VXLAN interfaces.

Understanding CoS on VXLAN Interfaces

This section describes how classification and rewrite rules are applied to packets in a VXLAN instance. Figure 63 on page 847 shows a simple VXLAN with two leaf nodes and one spine node.

Figure 63: Classifiers and Rewrite Rules on VXLANs



Refer to Figure 63 on page 847 to understand the packet flow with DSCP/ToS fields in a VXLAN:

1. CE 1 sends a packet with Layer3 DSCP/ToS bit programmed to the Leaf 1 node.
2. Leaf 1 receives the original packet and appends the VXLAN header on top of the original packet. The outer VXLAN Layer3 header uses the original packet DSCP/Tos bit. You can create classifiers based on the original packet DSCP/802.1p bit. The ingress interface on the ingress leaf supports DSCP and 802.1p classifiers.
3. If rewrite is configured on Leaf 1, the inner header will have the DSCP/802.1p bit set by CE 1 and the outer header will have the rewrite bit. Only DSCP rewrite rules are supported, except on QFX10000 switches where 802.1p rewrite is also supported if the underlay is tagged.
4. The Spine node receives the VXLAN packet and can use ingress classification using these DSCP bits and forward the packet to the egress interface with the appropriate forwarding class.
5. The Spine egress interface can rewrite these bits using rewrite rules. These Spine rewrite rules only affects the outer Layer3 DSCP field. The inner/original packet still holds the DSCP/802.1p bit that was set by CE 1.
6. Leaf 2 receives the packet, processes the tunnel termination, and remove the outer VXLAN header.
7. Leaf 2 classification and rewrite functionality works on the inner header.
8. The original packet arrives on CE 2.

NOTE: On the leaf nodes, if the packet is multicast, you can use `multi-destination` classification to create appropriate multicast classification and rewrite rules.

Configuring CoS on VXLAN Interfaces

This section shows sample configurations of classifiers and rewrite rules for the leaf and spine nodes in VXLAN using Figure 63 on page 847 as a reference. You can create schedulers as normal for the classifiers on each node.

Sample configuration of classifiers and rewrite rules on Leaf 1.

1. Create a classifier based on the *original* DSCP/ToS bits:

```
[edit class-of-service classifiers]
user@leaf1#set dscp dscp_cf forwarding-class best-effort loss-priority low code-points 100000
user@leaf1#set dscp dscp_cf forwarding-class network-control loss-priority high code-points
110000
user@leaf1#set dscp dscp_cf forwarding-class expedited-forwarding loss-priority low code-
points 011010
user@leaf1#set dscp dscp_cf forwarding-class assured-forwarding loss-priority high code-
points 001010
```

2. Apply the classier to the ingress interface:

```
[edit class-of-service interfaces]
user@leaf1#set ge-0/0/0 unit 0 classifiers dscp dscp_cf
```

3. Create a rewrite rule for the *outer* VXLAN DSCP/ToS bits:

```
[edit class-of-service rewrite-rules]
user@leaf1#set dscp dscp_rw forwarding-class best-effort loss-priority low code-points af22
user@leaf1#set dscp dscp_rw forwarding-class network-control loss-priority high code-points
af31
user@leaf1#set dscp dscp_rw forwarding-class expedited-forwarding loss-priority low code-
points af13
user@leaf1#set dscp dscp_rw forwarding-class assured-forwarding loss-priority high code-
points cs3
```

4. Apply the rewrite rule to the egress Leaf 1 interfaces:

```
[edit class-of-service interfaces]
user@leaf1#set ge-0/0/1 unit 0 rewrite-rules dscp dscp_rw
user@leaf1#set ge-0/0/2 unit 0 rewrite-rules dscp dscp_rw
```

Sample configuration of classifiers and rewrite rules on the Spine.

1. Create a classifier based on the outer VXLAN DSCP/ToS bits:

```
[edit class-of-service classifiers]
user@spine#set dscp dscp_cf forwarding-class best-effort loss-priority low code-points af22
user@spine#set dscp dscp_cf forwarding-class network-control loss-priority high code-points
af31
user@spine#set dscp dscp_cf forwarding-class expedited-forwarding loss-priority low code-
points af13
user@spine#set dscp dscp_cf forwarding-class assured-forwarding loss-priority high code-
points cs3
```

2. Apply the classier to the ingress Spine interfaces:

```
[edit class-of-service interfaces]
user@spine#set ge-0/0/3 unit 0 classifiers dscp dscp_cf
user@spine#set ge-0/0/5 unit 0 classifiers dscp dscp_cf
```

3. Create a rewrite rule for the outer VXLAN DSCP/ToS bits:

```
[edit class-of-service rewrite-rules]
user@spine#set dscp dscp_rw forwarding-class best-effort loss-priority low code-points af22
user@spine#set dscp dscp_rw forwarding-class network-control loss-priority high code-points
af31
user@spine#set dscp dscp_rw forwarding-class expedited-forwarding loss-priority low code-
points af13
user@spine#set dscp dscp_rw forwarding-class assured-forwarding loss-priority high code-
points cs3
```

4. Apply the rewrite rule to the egress Spine interfaces:

```
[edit class-of-service interfaces]
user@spine#set ge-0/0/4 unit 0 rewrite-rules dscp dscp_rw
user@spine#set ge-0/0/6 unit 0 rewrite-rules dscp dscp_rw
```

Sample configuration of classifiers and rewrite rules on Leaf 2.

1. Create a classifier based on the *original* DSCP/ToS bits, as the VXLAN header is removed at tunnel termination *before* forwarding classes are applied:

   ```
   [edit class-of-service classifiers]
   user@leaf2#set dscp dscp_cf forwarding-class best-effort loss-priority low code-points 100000
   user@leaf2#set dscp dscp_cf forwarding-class network-control loss-priority high code-points
   110000
   user@leaf2#set dscp dscp_cf forwarding-class expedited-forwarding loss-priority low code-
   points 011010
   user@leaf2#set dscp dscp_cf forwarding-class assured-forwarding loss-priority high code-
   points 001010
   ```

2. Apply the classier to the ingress Leaf 2 interfaces:

   ```
   [edit class-of-service interfaces]
   user@leaf2#set ge-0/0/7 unit 0 classifiers dscp dscp_cf
   user@leaf2#set ge-0/0/8 unit 0 classifiers dscp dscp_cf
   ```

3. Create a rewrite rule for the *original* DSCP/ToS bits:

   ```
   [edit class-of-service rewrite-rules]
   user@leaf2#set dscp dscp_rw forwarding-class best-effort loss-priority low code-points 100000
   user@leaf2#set dscp dscp_rw forwarding-class network-control loss-priority high code-points
   110000
   user@leaf2#set dscp dscp_rw forwarding-class expedited-forwarding loss-priority low code-
   points 011010
   user@leaf2#set dscp dscp_rw forwarding-class assured-forwarding loss-priority high code-
   points 001010
   ```

4. Apply the rewrite rule to the egress Leaf 2 interface:

   ```
   [edit class-of-service interfaces]
   user@leaf2#set ge-0/0/9 unit 0 rewrite-rules dscp dscp_rw
   ```

To check the CoS configuration on one of the interfaces:

```
user@node#show class-of-service interface interface-name
```

To check the queue statistics on one of the interfaces:

```
user@node#show interfaces queue interface-name
```

Implementing CoS on VXLAN Interfaces (Junos OS Evolved)

CoS for EVPN VXLAN traffic is supported using a combination of classifiers, schedulers, and rewrite rules. This section describes how these components are implemented across different nodes on devices running Junos OS Evolved to apply CoS on the EVPN VXLAN traffic.

- **Classification at User Network Interface (UNI)/Ingress PE** — Traffic classification based on IEEE 802.1p and Differentiated Services code point (DSCP) are supported on the ingress PE where the EVPN VXLAN tunnel is initiated. BA and MF classifiers can be applied to Enterprise style (EP) or Service Provider (SP) style access interfaces.

- **Classification at Network Node Interface (NNI)/Egress PE** — Traffic classification based on IEEE 802.1p and Differentiated Services code point (DSCP) are supported on the egress PE where the EVPN VXLAN tunnel is terminated. BA classifiers can be applied to the underlying logical interface or unit. MF classifiers are not supported in tunnel terminations.

- **Rewrite at NNI** — After the encapsulation of the VXLAN tunnel, the rewrites on the outer/tunnel header are configured using the rewrite rules on the underlying logical interface or unit. Based on the configured rewrite rules, the VXLAN traffic is classified in the Spine/Network.

  DSCP rewrite on the outer/tunnel header of VXLAN packets is supported on the NNI interface.

  Rewrite rules are supported in the following EVPN VXLAN scenarios:

  - Intra-VNI L2 gateway — Rewrite rules are applied to both unicast and broadcast, unknown unicast, and multicast (BUM) traffic.

  - Inter-VNI L3 gateway — Centrally-routed bridging (CRB) and edge-routed bridging (ERB).

  - EVPN Type 5 routes.

- **Rewrite at UNI** — After the termination of the VXLAN tunnel, the rewrites on the inner headers are configured using rewrite rules on the Enterprise style (EP) or Service Provider (SP) style access interfaces. Based on the configured rewrite rules, the de-encapsulated packets are classified in the CE side network. The following rewrite rules are supported on the UNI interface for the de-encapsulated packets:

  - DSCP rewrites on the inner IPv4/IPv6 header

  - IEEE 802.1p rewrites on the inner Ethernet header (if tagged)

  Rewrite rules are supported in the following EVPN VXLAN scenarios:

- Intra-VNI L2 gateway — Rewrite rules are applied to both unicast and broadcast, unknown unicast, and multicast (BUM) traffic.
- Inter-VNI L3 gateway — Centrally-routed bridging (CRB) and edge-routed bridging (ERB).
- EVPN Type 5 routes.

- **Scheduling** — Traffic prioritization and bandwidth reservation are achieved by using schedulers. The schedulers are associated with a forwarding class set via classifiers.

Platform-Specific CoS on EVPN-VxLANs Behavior

Use Feature Explorer to confirm platform and release support for specific features.

Use the following table to review platform-specific behaviors for CoS on EVPN-VxLANS:

Platform	Difference
EX4100, EX4300MP, and EX4400 switches	EX4100, EX4300MP, and EX4400 switches support CoS features such as classification and rewrite on IRB interfaces that carry EVPN-VxLAN traffic.

(Continued)

Platform	Difference
PTX Series routers	The following limitations apply to PTX routers: • DSCP rewrite rules are not supported on Integrated Routing and Bridging (IRB) (L3 gateway scenarios). • IEEE 802.1p rewrite rules are not supported on the NNI interface. • Explicit congestion notification (ECN) rewrites are not supported on either UNI or NNI interfaces. • Priority-based flow control (PFC) is not supported. • No support for CoS classification and rewrite mechanism for IRB underlay. **NOTE**: In EVPN-VXLAN networks with an IPv6 underlay, some PTX Series routers support CoS classification and explicit congestion notification (ECN) copy operations, but not PFC, DSCP copy, or IEEE 802.1p rewrite. See Feature Explorer for platform and release support.
PTX10002-36QDD routers	PTX10002-36QDD routers that originate or terminate EVPN-VXLAN tunnels and have ECN enabled automatically copy the ECN bits from the inner header to the outer header. The router copies the ECN bits from the outer header to the inner header if the inner header has the ECT bit set. If the router experiences congestion, it sets the CE bits if the ECT bit is enabled. These routers support ECN copy operations in EVPN-VXLAN networks with an IPv4 underlay or an IPv6 underlay. PTX10002-36QDD routers that originate EVPN-VXLAN tunnels copy the DSCP bits from the inner to the outer header during encapsulation. If DSCP rewrite is configured, DSCP rewrite takes precedence over DSCP copy.

(Continued)

Platform	Difference
QFX5130 and QFX5700 switches	The following limitations apply to the QFX5130 and QFX5700 platforms: • HQoS is not supported due to hardware limitations. • Classifier, rewrite and scheduler on IRB interface is not supported. • DOT1P rewrite and classifier on the NNI port is not supported. • DOT1P and DSCP rewrite on the UNI port is not supported. • DSCP rewrite on the NNI port is supported with the following conditions: • DSCP rewrite on the NNI port is supported by default and does not work if `vxlan-tos-copy-filter` at `[edit forwarding-options]` is enabled. • Inner ECN bits are copied to the outer VXLAN header regardless of whether `vxlan-tos-copy-filter` at `[edit forwarding-options]` is enabled. • Do not enable `vxlan-tos-copy-filter` and `vxlan-disable-copy-tos-encap` at `[edit forwarding-options]`at the same time as this causes in-deterministic behavior. • PFC configuration will cause momentary traffic drops of up to 10ms. • DSCP IPV6 classifiers and rewrites are not supported. Use DSCP classifier and rewrite instead. • TOS copy feature does not work for Type-5 EVPN VXLANs.

(Continued)

Platform	Difference
QFX10000 Series switches	The following limitation applies to QFX10000 platforms: • Because IRB interfaces do not support dscp rewrite rules, you can apply rewrite rules on underlying L2 interfaces. 802.1p/dscp values in a VXLAN tunneled packet are written using underlying L2 interface rules.

Understanding CoS CLI Configuration Statements on PTX Series Routers

PTX Series Packet Transport Routers have no new Junos OS CLI configuration statements. However, some statements or statement options supported on other platforms are not supported or may not have effect on PTX Series devices. These exceptions are summarized here.

[edit chassis] Hierarchy Level

The following statement is not applicable to PTX Series Packet Transport Routers. There are always eight queues available. However, if there is a requirement to use only four of eight queues, you can do this by configuring the forwarding class to queue mapping, as appropriate.

```
[edit chassis fpc slot-number pic pic-number],
    max-queues-per-interface (4 | 8);
```

The following CLI is not applicable to PICs supported on PTX Series Packet Transport Routers:

```
[edit chassis fpc slot-number pic pic-number],
    q-pic-large-buffer {
    [large-scale | small-scale]
}
```

On PTX Series Packet Transport Routers, buffer occupancy is computed as weighted average. However, configuration of weight at the PIC level is not supported. The default weights are applied.

```
[edit chassis fpc slot-number pic pic-number],
red-buffer-occupancy {
    weighted-averaged [ instant-usage-weight-exponent ] weight-value;
}
```

The following CLI is not applicable to PICs supported on PTX Series Packet Transport Routers:

```
[edit chassis fpc slot-number pic pic-number],
traffic-manager {
    egress-shaping-overhead number;
    ingress-shaping-overhead number;
    mode session-shaping;
}
```

[edit class-of-service] Hierarchy Level

The following CLI is not applicable to PTX Series Packet Transport Routers because there are no separate fabric queues and egress queues:

```
fabric {
    scheduler-map {
        priority (high | low) scheduler scheduler-name;
    }
}
```

The following CLI does not support the `priority` and `policing-priority` options.

```
forwarding-classes {
    class queue-num queue-number priority (high | low);
    queue queue-number class-name priority (high | low) [ policing-priority (premium | normal) ];
}
```

The following statements are not supported on PTX Series Packet Transport Routers:

- `inet-precedence` rewrite
- Rewrite of both exp and inet-precedence fields for VPN and non-VPN traffic that use the `mpls-inet-both` and `mpls-inet-both-non-vpn` protocol types.

- `exp-push-push-push` and `exp-swap-push-push` rules
- `input-scheduler-map` and `input-shaping-rate`
- The physical interface scheduler is applied on the Packet Forwarding Engine, hence the `scheduler-map-chassis` statement is not applicable.

```
interfaces {
    interface-name {
        input-scheduler-map map-name;
        input-shaping-rate rate;
        scheduler-map-chassis map-name;
        unit logical-unit-number {
            rewrite-rules{
                inet-precedence (rewrite-name | default) protocol
                protocol-types;
                exp (write-name | default) protocol protocol-types;
                exp-push-push-push default;
                exp-swap-push-push default;
            }
        }
    }
}
```

In the following CLI, only the `inet-precedence` statement is not supported.

```
rewrite-rules {
    (dscp | dscp-ipv6 | exp | ieee-802.1 |ieee-802.1ad |inet-precedence) rewrite-name {
        import (rewrite-name | default);
        forwarding-class class-name {
            loss-priority level code-point (alias | bits);
        }
    }
}
```

Classifiers on routing instances are not supported on PTX Series Packet Transport Routers because L3VPN is not supported. Hence, the following CLI is not applicable.

```
[edit class-of-service]
routing-instances routing-instance-name {
    classifiers {
        exp (classifier-name | default);
        dscp (classifier-name | default);
        dscp-ipv6 (classifier-name | default);
    }
}
```

The following limitations apply to statements under `schedulers` on PTX Series Packet Transport Routers:

- **protocol** (non-tcp | tcp) is not supported for **drop-profile-map**. The **any** option is supported.
- **excess-priority** is not supported.
- **rate-limit** is supported for **transmit-rate**. It is applied only when schedulers are configured as **strict-high**.

```
schedulers (CoS) {
    scheduler-name {
        buffer-size (percent percentage | remainder | temporal microseconds);
        drop-profile-map loss-priority (any | low | medium-low | medium-high
        high) protocol (any ) drop-profile profile-name;
        priority priority-level;
        transmit-rate (rate | percent percentage | remainder) <exact | rate-limit>;
    }
}
```

NOTE: Configurations that are supported only on Gigabit Ethernet IQ PICs, channelized IQ PICs, and so forth are not applicable to PTX Series Packet Transport Routers. These PICs are not supported on this platform. Those CLIs are not listed here.

[edit firewall] Hierarchy Level

In the following CLI, the **dscp** clause is not supported.

```
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;
            }
        }
    }
}
```

Classification Based on Outer Header of Decapsulation Tunnel

SUMMARY

By default, when a packet exits a tunnel, the terminating router removes the tunnel (outer) header and forwards the packet based on the classification in the inner header. On supported routers that act as a terminating router for a tunnel, you can forward the packet based on the classification you assign to the *outer* header at the underlay (ingress) interface.

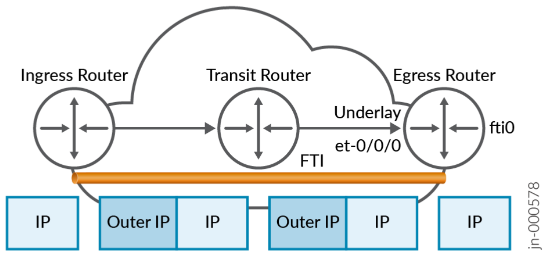
IN THIS SECTION

Overview of Classification Based on Outer Header of Decapsulation Tunnel

A flexible tunnel interface (FTI) is a type of logical tunnel interface that uses static routing and BGP protocols to exchange routes over a tunnel that connects endpoints to routers. You can configure GRE, IPIP, or UDP tunnels on the logical interfaces of an FTI. For information on configuring FTIs, see *Configuring Flexible Tunnel Interfaces*.

For an IP packet entering the tunnel, the ingress router (as shown in Figure 64 on page 860) encapsulates the packet with the tunnel header (the outer IP header). This tunnel header directs the packet through the tunnel to the egress router.

Figure 64: FTI Tunnel



By default, when the packet exits the tunnel, the egress router removes the tunnel (outer) header and forwards the packet based on the classification in the inner header. You *can* forward the packet based on the classification you assign to the *outer* header at the underlay (ingress) interface of the egress router. You do this by first setting the `no-default` option at the `[edit class-of-service interfaces fti`*`n`*` unit 0 classifiers]` hierarchy level. You then assign the desired classifier to the underlay interface of the egress router.

NOTE: Applying the `no-default` option disables the application of any default classifier to the routing instance. In this case, applying the `no-default` option specifically prevents classification based on the inner header, enabling classification based on the outer header.

Configuring Classification Based on Outer Header of Decapsulation Tunnel

This section describes how to configure the classification of a packet exiting a tunnel on a terminating router to be based on the outer (underlay) header of the packet.

1. Ensure the flexible tunnel interface that terminates the tunnel is indeed configured for tunnel termination.

   For example:

```
[edit interfaces]
user@host# show fti0
unit 0 {
    tunnel {
        encapsulation gre|ipip|udp {
            tunnel-termination;
            ...
        }
    }
}
```

2. Apply the no-default option for classifiers on the flexible tunnel interface.

   For example:

```
[edit class-of-service]
user@host# set interfaces fti0 unit 0 classifiers no-default
user@host# show interfaces
fti0 {
    unit 0 {
        classifiers {
            no-default;
        }
    }
}
```

3. Apply the desired classifier for packets exiting the tunnel to the underlay (ingress) interface.

   For example:

```
[edit class-of-service]
user@host# set interfaces et-0/0/0 unit 0 classifiers dscp dscp1
user@host# show interfaces
et-0/0/0 {
    unit 0 {
        classifiers {
            dscp dscp1;
        }
    }
}
fti0 {
    unit 0 {
        classifiers {
            no-default;
        }
    }
}
```

4. Commit your changes.

RELATED DOCUMENTATION

classifiers (Interfaces)

Configuring Flexible Tunnel Interfaces

4

PART

Configuring Line Card-Specific and Interface-Specific Functionality

CHAPTER 14

Feature Support of Line Cards and Interfaces

IN THIS CHAPTER

CoS Features of the Router Hardware, PIC, MIC, and MPC Interface Families

Table 76 on page 864 compares the PIC families with regard to major CoS features. Note that this table reflects the ability to perform the CoS function *at the PIC , MIC, or MPC interface level* and not on the system as a whole.

Table 76: CoS Features of the Router Hardware and Interface Families Compared

Feature:	MIC and MPC Interfaces	IQ PICs	IQ2 PICs	IQ2E PICs	Enhanced IQ PICs
BA classification	Yes	–	–	–	Yes
ToS bit rewrites	Yes	Yes, for IEEE bits only	Yes, for IEEE bits only	Yes, for IEEE bits only	–
Ingress ToS bit rewrites	Yes, with firewall filter	–	–	–	Yes
Hierarchical policers	Yes	–	–	–	Yes

Scheduling on the Router Hardware, PIC, MIC, and MPC Interface Families

Table 77 on page 865 compares the PIC, MIC, and MPC interface families with regard to scheduling abilities or features. Note that this table reflects the ability to perform the function *at the PIC, MIC, or MPC interface level* and not necessarily on the system as a whole.

Table 77: Scheduling on the Router Hardware and Interface Families Compared

Scheduling Feature:	MIC and MPC Interfaces	IQ PICs	IQ2 PICs	IQ2E PICs	Enhanced IQ PICs
Per–unit scheduling	Yes, for EQ MPC	Yes	Yes	Yes	Yes
Physical port and logical unit shaping	Yes	–	Yes	Yes	Yes
Guaranteed rate or peak rate support	Yes	–	Yes, supports both CIR and PIR on the same logical unit.	Yes	Yes, at the logical unit
Excess rate support	Yes	–	–	–	Yes, at the logical unit
Shared scheduler support	–	–	Yes	Yes	–

Schedulers on the Router Hardware, PIC, MIC, and MPC Families

Table 78 on page 866 compares the PIC, MIC, and MPC interface families with regard to scheduler statements or features. Note that this table reflects the ability to perform the scheduler function *at the PIC, MIC, or MPC interface level* and not necessarily on the system as a whole.

Table 78: Schedulers on the Router Hardware and Interface Families Compared

Scheduler Statement or Feature:	MIC and MPC Interfaces	IQ PICs	IQ2 PICs	IQ2E PICs	Enhanced IQ PICs
Exact transmit rate	Yes	Yes	–	–	Yes
Rate-limit transmit rate	Yes	–	Yes	Yes	Yes
More than one high-priority queue	Yes	Yes	–	Yes	Yes
Excess priority or sharing	Yes	–	–	–	Yes
Hierarchical Scheduling	Yes, for EQ MPC	–	–	Yes	–

Queuing Parameters for the Router Hardware, PIC, MIC, and MPC Interface Families

Table 79 on page 867 compares the PIC, MIC, and MPC interface families with regard to queuing parameters and features.

Table 79: Queue Parameters on the Router Hardware and Interface Families Compared

Queuing Statement or Feature:	MIC and MPC Interfaces	IQ PICs	IQ2 PICs	IQ2E PICs	Enhanced IQ PICs
Maximum number of queues	8	8	8	8	8
Maximum delay buffer bandwidth	100 ms for 1 Gbps and up; 500 ms for others	100 ms	200 ms	200 ms	up to 4000 ms
Packet transmit priority level	3 and 2	2 and 2	2	3	3 and 2
Maximum number of drop profiles	64	32 (32 samples)	32	32	64
Packet loss priority level	4	4	4	4	4

CHAPTER 15

Configuring Class of Service for Tunnels

IN THIS CHAPTER

CoS for Tunnels Overview

IN THIS SECTION

Class-of-service (CoS) information is preserved inside generic routing encapsulation (GRE) and IP-IP tunnels.

Class-of-service information is preserved inside IP Security (IPsec) tunnels. For IPsec tunnels, you do not need to configure CoS, because the ES PIC copies the type-of-service (ToS) byte from the inner IP header to the GRE or IP-IP header.

For IPsec tunnels, the IP header type-of-service (ToS) bits are copied to the outer IPsec header at encryption side of the tunnel. You can rewrite the outer ToS bits in the IPsec header using a rewrite rule. On the decryption side of the IPsec tunnel, the ToS bits in the IPsec header are not written back to the original IP header field. You can still apply a *firewall filter* to the ToS bits to apply a packet action on egress. For more information about IPsec and Multiservices PICs, see the Junos OS Services Interfaces Library for Routing Devices.

To configure CoS for tunnels, include the following statements at the `[edit class-of-service]` and `[edit interfaces]` hierarchy level:

```
[edit class-of-service]
interfaces {
    interface-name {
        unit logical-unit-number {
            rewrite-rules {
                dscp (rewrite-name | default);
                dscp-ipv6 (rewrite-name | default);
                exp (rewrite-name | default)protocol protocol-types;
                exp-push-push-push default;
                exp-swap-push-push default;
                ieee-802.1 (rewrite-name | default);
                inet-precedence (rewrite-name | default);
            }
        }
    }
}
rewrite-rules {
    (dscp | dscp-ipv6 | exp | ieee-802.1 | inet-precedence) rewrite-name {
        import (rewrite-name | default);
        forwarding-class class-name {
            loss-priority level code-point (alias | bits);
        }
    }
}
[edit interfaces]
gre-interface-name {
    unit logical-unit-number {
        copy-tos-to-outer-ip-header;
        copy-tos-to-outer-ip-header-transit;
        force-control-packets-on-transit-path
        tunnel {
            traffic-class traffic-class;
        }
    }
}
```

Tunneling and BA Classifiers

On supported platforms, BA classifiers can be used with GRE and IP-IP tunnels.

NOTE: MPCs do not support BA classifiers on gr- interfaces. Use multifield classifiers instead.

When a GRE or IP-IP tunnel is configured on an incoming (core-facing) interface, the queue number and PLP information are carried through the tunnel. At the egress (customer-facing) interface, the packet is queued and the CoS bits rewritten based on the information carried through the tunnel.

If no BA classifier is configured in the incoming interface, the default classifier is applied. If no rewrite rule is configured, the default rewrite rule is applied.

NOTE: For GRE and IP-IP tunnels, IP precedence and DSCP rewrite marking of the inner header do not work with more than eight forwarding classes.

RELATED DOCUMENTATION

Configuring CoS for GRE and IP-IP Tunnels | 870

Example: Configuring CoS for GRE or IP-IP Tunnels | 871

Configuring CoS for GRE and IP-IP Tunnels

To configure CoS for GRE and IP-IP tunnels, perform the following configuration tasks:

1. To configure CoS for an IP-IP tunnel, include the `tunnel` statement at the `[edit interfaces ip-`*`fpc/pic/port`* `unit` *`logical-unit-number`*`]` hierarchy level. To configure CoS for a GRE tunnel, include the `tunnel` statement at the `[edit interfaces gr-`*`fpc/pic/port`* `unit` *`logical-unit-number`*`]` hierarchy level.
2. To rewrite traffic on the outbound interface, include the `rewrite-rules` statement at the `[edit class-of-service]` and `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`*`]` hierarchy levels. For GRE and IP-IP tunnels, you can configure IP precedence and DSCP rewrite rules.
3. To classify traffic on the inbound interface, you can configure a behavior aggregate (BA) classifier or a firewall filter. Include the `loss-priority` and `forwarding-class` statements at the `[edit firewall filter` *`filter-name`* `term` *`term-name`* `then]` hierarchy level, or the `classifiers` statement at the `[edit class-of-service]` hierarchy level.

NOTE: You cannot configure BA classifiers on `gr-` interfaces. You must classify traffic on `gr-` interfaces using firewall filters (multifield classifiers).

4. For a GRE tunnel, the default is to set the ToS bits in the outer IP header to all 0s. To copy the ToS bits from the inner IP header to the outer, include the `copy-tos-to-outer-ip-header-transit` statement at the `[edit interfaces gr-`*`fpc/pic/port`* `unit` *`logical-unit-number`*`]` hierarchy level. (This inner-to-outer ToS bits copying is already the default behavior for IP-IP tunnels.)

   NOTE: This option works only for IPv4 GRE tunnels. This option does not support IPv6 traffic.

   To verify that this option is enabled at the interface level, use the `show interfaces` *`interface-name`* `detail` command.

   To set a static ToS/Traffic Class value in the outer IP header, include the `traffic-class` *`traffic-class-value`* statement at the `[edit interfaces gr-`*`fpc/pic/port`* `unit` *`logical-unit-number`* `tunnel]` hierarchy level. Setting this value overrides the `copy-tos-to-outer-ip-header-transit` statement. If rewrite rules are configured on the egress WAN interface, those rewrite rules will overwrite this setting. Therefore the `traffic-class` setting only makes sense when no rewrite rules are configured.

RELATED DOCUMENTATION

CoS for Tunnels Overview | **868**

Example: Configuring CoS for GRE or IP-IP Tunnels

IN THIS SECTION

This topic provides an example of how to configure class of service (CoS) for GRE or IP-IP tunnels.

Requirements

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

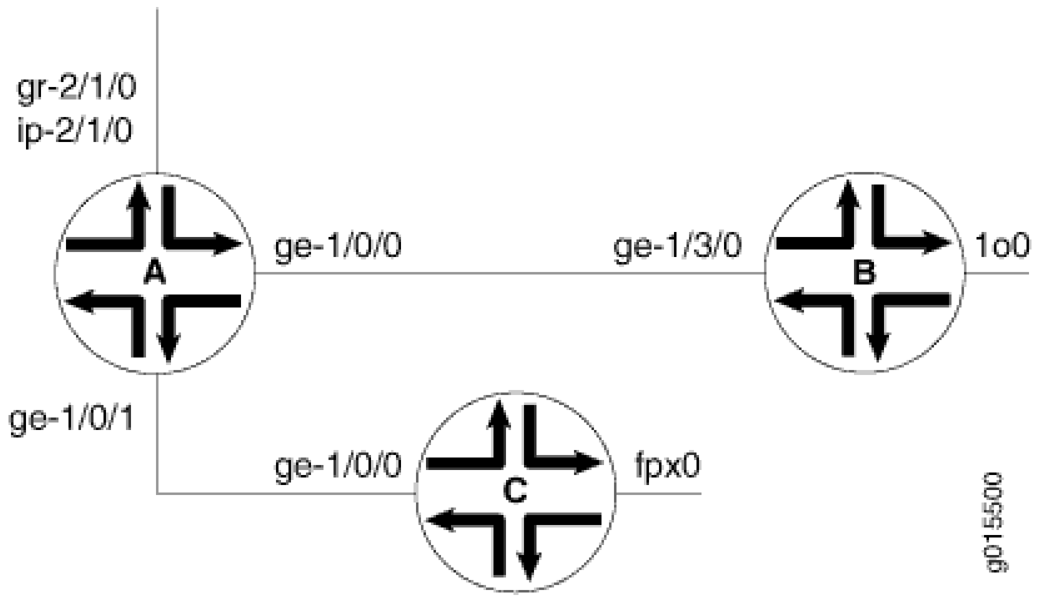
Overview

IN THIS SECTION

Topology

In Figure 65 on page 873, Router A acts as a tunnel ingress device. The link between interfaces `ge-1/0/0` in Router A and `ge-1/3/0` in Router B is the GRE or IP-IP tunnel. Router A monitors the traffic received from interface `ge-1/3/0`. By way of interface `ge-1/0/0`, Router C generates traffic to Router B.

Figure 65: CoS with a Tunnel Configuration



Configuration

IN THIS SECTION

To configure this example, perform these tasks:

CLI Quick Configuration

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

Router A

```
set interfaces ge-1/0/0 unit 0 family inet address 10.80.0.2/24
set interfaces ge-1/0/1 unit 0 family inet filter input zf-catch-all
set interfaces ge-1/0/1 unit 0 family inet address 10.90.0.2/24
set interfaces gr-2/1/0 unit 0 tunnel source 10.11.11.11 destination 10.255.245.46
set interfaces gr-2/1/0 unit 0 family inet address 10.21.21.21/24
set interfaces ip-2/1/0 unit 0 tunnel source 10.12.12.12 destination 10.255.245.46
set interfaces ip-2/1/0 unit 0 family inet address 10.22.22.22/24
set routing-options static route 10.1.1.1/32 next-hop gr-2/1/0.0
set routing-options static route 10.2.2.2/32 next-hop ip-2/1/0.0
set class-of-service interfaces ge-1/0/0 unit 0 rewrite-rules inet-precedence zf-tun-rw-ipprec-00
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class best-
effort loss-priority low code-point 000
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class best-
effort loss-priority high code-point 001
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class
expedited-forwarding loss-priority low code-point 010
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class
expedited-forwarding loss-priority high code-point 011
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class assured-
forwarding loss-priority low code-point 100
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class assured-
forwarding loss-priority high code-point 101
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class network-
control loss-priority low code-point 110
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class network-
control loss-priority high code-point 111
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class best-effort loss-
priority low code-point 000000
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class best-effort loss-
priority high code-point 001001
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class expedited-forwarding
loss-priority low code-point 010010
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class expedited-forwarding
loss-priority high code-point 011011
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class assured-forwarding
loss-priority low code-point 100100
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class assured-forwarding
loss-priority high code-point 101101
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class network-control loss-
priority low code-point 110110
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class network-control loss-
priority high code-point 111111
set firewall filter zf-catch-all term term1 then loss-priority high
set firewall filter zf-catch-all term term1 then forwarding-class network-control
```

Router B

```
user@router-B# set interfaces ge-1/3/0 unit 0 family inet address 10.80.0.1/24
user@router-B# set interfaces lo0 unit 0 family inet address 10.255.245.46/32
```

Router C

```
set interfaces ge-1/0/0 unit 0 family inet address 10.90.0.1/24
set routing-options static route 10.1.1.1/32 next-hop 10.90.0.2
set  routing-options static route 10.2.2.2/32 next-hop 10.90.0.2
```

Configuring Router A

IN THIS SECTION

Procedure

Step-by-Step Procedure

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

To configure router A:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@router-A# set ge-1/0/0 unit 0 family inet address 10.80.0.2/24
   user@router-A# set ge-1/0/1 unit 0 family inet filter input zf-catch-all
   user@router-A# set ge-1/0/1 unit 0 family inet address address 10.90.0.2/24
   user@router-A# set gr-2/1/0 unit 0 tunnel source 10.11.11.11 destination 10.255.245.46
   user@router-A# set gr-2/1/0 unit 0 family inet address 10.21.21.21/24
   user@router-A# set ip-2/1/0 unit 0 tunnel source 10.12.12.12 destination 10.255.245.46
   user@router-A# set ip-2/1/0 unit 0 family inet address 10.22.22.22/24
   ```

2. Configure the static routes.

```
[edit routing-options static]
user@router-A# set static route 10.1.1.1/32 next-hop gr-2/1/0.0
user@router-A# set static route 10.2.2.2/32 next-hop ip-2/1/0.0
```

3. Apply the rewrite rule to the interface.

```
[edit class-of-service]
user@router-A# set interfaces ge-1/0/0 unit 0 rewrite-rules inet-precedence zf-tun-rw-
ipprec-00
```

4. Define the rewrite rules.

```
[edit class-of-service]
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class best-
effort loss-priority low code-point 000
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class best-
effort loss-priority high code-point 001
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class
expedited-forwarding loss-priority low code-point 010
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class
expedited-forwarding loss-priority high code-point 011
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class assured-
forwarding loss-priority low code-point 100
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class assured-
forwarding loss-priority high code-point 101
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class network-
control loss-priority low code-point 110
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class network-
control loss-priority high code-point 111
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class best-effort loss-
priority low code-point 000000
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class best-effort loss-
priority high code-point 001001
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class expedited-forwarding
loss-priority low code-point 010010
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class expedited-forwarding
loss-priority high code-point 011011
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class assured-forwarding
```

```
loss-priority low code-point 100100
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class assured-forwarding
loss-priority high code-point 101101
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class network-control loss-
priority low code-point 110110
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class network-control loss-
priority high code-point 111111
```

5. Configure the firewall filter.

```
[edit firewall]
user@router-A# set filter zf-catch-all term term1 then loss-priority high
user@router-A# set filter zf-catch-all term term1 then forwarding-class network-control
```

Results

From configuration mode, confirm your configuration by entering the `show interfaces`, `show routing-options`, `show class-of-service`, and `show firewall` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@router-A# show interfaces
ge-1/0/0 {
    unit 0 {
        family inet {
            address 10.80.0.2/24;
        }
    }
}
ge-1/0/1 {
    unit 0 {
        family inet {
            filter {
                input zf-catch-all;
            }
            address 10.90.0.2/24;
        }
    }
}
gr-2/1/0 {
    unit 0 {
        tunnel {
            source 10.11.11.11;
            destination 10.255.245.46;
        }
        family inet {
            address 10.21.21.21/24;
        }
    }
}
ip-2/1/0 {
    unit 0 {
        tunnel {
            source 10.12.12.12;
            destination 10.255.245.46;
        }
        family inet {
            address 10.22.22.22/24;
        }
    }
}
```

```
user@router-A# show routing-options
static {
    route 10.1.1.1/32 next-hop gr-2/1/0.0;
    route 10.2.2.2/32 next-hop ip-2/1/0.0;
}
```

```
user@router-A# show class-of-service
interfaces {
    ge-1/0/0 {
        unit 0 {
            rewrite-rules {
                inet-precedence zf-tun-rw-ipprec-00;
            }
        }
    }
}
rewrite-rules {
    inet-precedence zf-tun-rw-ipprec-00 {
        forwarding-class best-effort {
            loss-priority low code-point 000;
            loss-priority high code-point 001;
        }
        forwarding-class expedited-forwarding {
            loss-priority low code-point 010;
            loss-priority high code-point 011;
        }
        forwarding-class assured-forwarding {
            loss-priority low code-point 100;
            loss-priority high code-point 101;
        }
        forwarding-class network-control {
            loss-priority low code-point 110;
            loss-priority high code-point 111;
        }
    }
}
dscp zf-tun-rw-dscp-00 {
    forwarding-class best-effort {
        loss-priority low code-point 000000;
        loss-priority high code-point 001001;
    }
    forwarding-class expedited-forwarding {
        loss-priority low code-point 010010;
        loss-priority high code-point 011011;
    }
    forwarding-class assured-forwarding {
        loss-priority low code-point 100100;
        loss-priority high code-point 101101;
    }
    forwarding-class network-control {
        loss-priority low code-point 110110;
        loss-priority high code-point 111111;
    }
}
```

```
user@router-A# **show firewall**
filter zf-catch-all {
    term term1 {
        then {
            loss-priority high;
            forwarding-class network-control;
        }
    }
}
```

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

Configuring Router B

IN THIS SECTION

Procedure

Step-by-Step Procedure

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

To configure router B:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@router-B# set ge-1/3/0 unit 0 family inet address 10.80.0.1/24
   user@router-B# set lo0 unit 0 family inet address 10.255.245.46/32
   ```

Results

From configuration mode, confirm your configuration by entering the `show interfaces` command. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Router B

```
user@router-B# show interfaces
ge-1/3/0 {
```

```
    unit 0 {
        family inet {
            address 10.80.0.1/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.245.46/32;
        }
    }
}
```

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

Configuring Router C

IN THIS SECTION

Procedure

Step-by-Step Procedure

To configure router C:

1. Configure the device interfaces.

   ```
   [edit interfaces]
   user@router-B# set ge-1/0/0 unit 0 family inet address 10.90.0.1/24
   ```

2. Configure the static routes.

   ```
   [edit routing-options static]
   user@router-A# set static route 10.1.1.1/32 next-hop 10.90.0.2
   user@router-A# set static route 10.2.2.2/32 next-hop 10.90.0.2
   ```

Results

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

Router C

```
user@router-C show interfaces
ge-1/0/0 {
    unit 0 {
        family inet {
            address 10.90.0.1/24;
        }
    }
}

user@router-C show routing-options
static {
    route 10.1.1.1/32 next-hop 10.90.0.2;
    route 10.2.2.2/32 next-hop 10.90.0.2;
}
```

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

Verification

To verify the configuration, run the following commands:

- *show class-of-service rewrite-rule*
- *show firewall*

RELATED DOCUMENTATION

CoS for Tunnels Overview | **868**

Configuring CoS for GRE and IP-IP Tunnels | **870**

Configuring a GRE Tunnel to Copy ToS Bits to the Outer IP Header

Unlike IP-IP tunnels, GRE tunnels do not copy the ToS bits to the outer IP header by default. To copy the inner ToS bits to the outer IP header (which is required for some tunneled routing protocols) on packets sent by the Routing Engine, include the `copy-tos-to-outer-ip-header` statement at the logical unit hierarchy level of a GRE interface. To copy the inner ToS bits to the outer IP header on packets transiting the device (MX Series routers with MPCs only), include the `copy-tos-to-outer-ip-header-transit` statement at the logical unit hierarchy level of a GRE interface.

To copy the inner ToS bits to the outer IP header on a GRE tunnel:

- Specify the interface on which to enable the inner IP header's ToS bits to be copied to the outer IP packet header

```
[edit]
user@host# edit interfaces
user@host# set gr-0/0/0 unit 0 copy-tos-to-outer-ip-header
user@host# set gr-0/0/0 unit 0 copy-tos-to-outer-ip-header-transit
user@host# set gr-0/0/0 unit 0 family inet
```

You can also copy the inner ToS bits to the outer IP header on packets transiting the device on a global basis for all GRE interfaces by including the `copy-tos-to-outer service-type gre` statement at the `[edit chassis]` hierarchy level. This statement affects all GRE interfaces on MPCs and takes precedence over the `copy-tos-to-outer-ip-header-transit` statement. Once committed, this configuration statement only affects new `gr-` interfaces. To affect an existing `gr-` interface, you must delete and re-add the interface.

NOTE: Both the `copy-tos-to-outer service-type gre` and `copy-tos-to-outer-ip-header-transit` options work only for IPv4 GRE tunnels. These options do not support IPv6 traffic.

To verify that this option is enabled at the interface level, use the `show interfaces` *`interface-name`* `detail` command.

RELATED DOCUMENTATION

CoS for Tunnels Overview | 868

Configuring CoS for GRE and IP-IP Tunnels | 870

force-control-packets-on-transit-path

CHAPTER 16

Configuring Class of Service on Services PICs

IN THIS CHAPTER

CoS on Services PICs Overview

On Multiservices PICs with `lsq-` interfaces, there are additional features you can configure. One such feature is an additional method of classifying traffic flows based on applications, for example stateful firewalls and network address translation (NAT).

Application-based traffic flow classification enables you to configure a rule-based service that provides DiffServ code point (DSCP) marking and forwarding-class assignments for traffic transiting the Multiservices PIC. The service enables you to specify matching by application, application set, source, destination address, and match direction, and uses a similar structure to other rule-based services such as stateful firewall. The service actions allow you to associate the DSCP alias or value, forwarding-class name, system log activity, or a preconfigured application profile with the matched packet flows.

NOTE: If you configure a forwarding class map associating a forwarding class with a queue number, these maps are not supported on Multiservices link services intelligent queuing (`lsq-`) interfaces.

To configure class-of-service (CoS) features on the Multiservices PIC, include the `cos` statement at the `[edit services]` hierarchy level:

```
[edit services]
cos {
    application-profile profile-name {
        ftp {
            data {
                dscp (alias | bits);
                forwarding-class class-name;
            }
        }
        sip {
            video {
                dscp (alias | bits);
                forwarding-class class-name;
            }
            voice {
                dscp (alias | bits);
                forwarding-class class-name;
            }
        }
    }
    rule rule-name {
        match-direction (input | output | input-output);
        term term-name {
            from {
                application-sets [ set-names ];
                applications [ application-names ];
                destination-address address;
                destination-prefix-list list-name <except>;
                source-address address;
                source-address-range source-address-range low minimum-value high maximum-value
<except>;
                source-prefix-list list-name <except>;
            }
            then {
                application-profile profile-name;
                dscp (alias | bits);
                forwarding-class class-name;
                syslog;
                reflexive; | revert; | reverse { {
                    application-profile profile-name;
                    dscp (alias | bits);
                    forwarding-class class-name;
                    syslog;
                }
            }
        }
    }
    rule-set rule-set-name {
        [ rule rule-names ];
    }
}
```

RELATED DOCUMENTATION

Configuring CoS Rules on Services PICs | 886

Configuring CoS Rule Sets on Services PICs | 895

Configuring CoS Rules on Services PICs

IN THIS SECTION

This topic describes how to configure CoS rules on Services PICs.

Each CoS rule consists of a set of terms, similar to those in a firewall filter configuration. A term consists of the following:

- `from` statement—Specifies the match conditions and applications that are included and excluded.
- `then` statement—Specifies the actions and action modifiers to be performed by the router software.

If you omit the `from` term, the router accepts all traffic and the default protocol handlers take effect:

- User Datagram Protocol (UDP), Transmission Control Protocol (TCP), and Internet Control Message Protocol (ICMP) create a bidirectional flow with a predicted reverse flow.
- IP creates a unidirectional flow.

In addition, each rule must include a `match-direction` statement that specifies the direction in which the rule match is applied. To configure where the match is applied, include the `match-direction` statement at the `[edit services cos rule` *`rule-name`*`]` hierarchy level:

```
match-direction (input | output | input-output);
```

If you configure `match-direction input-output`, bidirectional rule creation is allowed.

The match direction is used with respect to the traffic flow through the Services PIC. When a packet is sent to the Services PIC, direction information is carried along with it.

On interface service sets, packet direction is determined by whether a packet is entering or leaving the interface on which the service set is applied.

With a next-hop service set, packet direction is determined by the interface used to route the packet to the Services PIC. If the inside interface is used to route the packet, the packet direction is `input`. If the outside interface is used to direct the packet to the Services PIC, the packet direction is `output`. For more information on inside and outside interfaces, see *Configuring Service Sets to be Applied to Services Interfaces*.

On the Services PIC, a flow lookup is performed. If no flow is found, rule processing is performed. All rules in the service set are considered. During rule processing, the packet direction is compared against rule directions. Only rules with direction information that matches the packet direction are considered.

You can use either the source address or the destination address as a match condition, in the same way that you would configure a firewall filter; for more information, see the Routing Policies, Firewall Filters, and Traffic Policers User Guide.

You can also include application protocol definitions that you have configured at the `[edit applications]` hierarchy level; for more information, see the Junos OS Services Interfaces Library for Routing Devices.

- To apply one or more specific application protocol definitions, include the `applications` statement at the `[edit services cos rule` *`rule-name`* `term` *`term-name`* `from]` hierarchy level.
- To apply one or more sets of application protocol definitions you have defined, include the `application-sets` statement at the `[edit services cos rule` *`rule-name`* `term` *`term-name`* `from]` hierarchy level.

NOTE: If you include a statement that specifies application protocols, the router derives port and protocol information from the corresponding configuration at the `[edit applications]` hierarchy level; you cannot specify these properties as match conditions.

The following sections describe how to configure CoS rules in more detail:

Configuring Match Conditions in a CoS Rule

This topic describes how to configure the match conditions for CoS rules.

Before you begin, make sure you have completed the following tasks:

- Configure the application protocol definitions at the `[edit applications]` hierarchy level; for more information, see the `application` and Junos OS Services Interfaces Library for Routing Devices.
- Configure a destination prefix list by including the `prefix-list` statement at the `[edit policy-options]` hierarchy level.
- Configure a source prefix list by including the `prefix-list` statement at the `[edit policy-options]` hierarchy level.

To configure the match conditions for a CoS rule:

1. Create the CoS rule by specifying a name for it.

```
[edit]
user@host# edit services cos rule rule-name
```

2. Specify the direction in which the rule match is applied.

```
[edit services cos rule rule-name]
user@host# set match-direction (input | output | input-output)
```

3. Specify the input conditions for the CoS term:

   a. Define one or more target application sets.

```
[edit services cos rule rule-name]
user@host# set term term-name from application-sets [ set-names ]
```

NOTE: You must configure the application protocol definitions at the `[edit applications]` hierarchy level; for more information, see the Junos OS Services Interfaces Library for Routing Devices.

b. Define one or more applications to which the CoS services apply.

```
[edit services cos rule rule-name]
user@host# set term term-name from applications [ application-names ]
```

c. Specify the destination address for rule matching.

```
[edit services cos rule rule-name]
user@host# set  term term-name from destination-address address
```

d. Specify the name of the destination prefix list for rule matching.

```
[edit services cos rule rule-name]
user@host# set term term-name from destination-prefix-list list-name <except>
```

NOTE: You must configure the destination prefix list by including the `prefix-list` statement at the `[edit policy-options]` hierarchy level.

e. Specify the source address for rule matching.

```
[edit services cos rule rule-name]
user@host# set term term-name from source-address address
```

f. Specify the source address range for rule matching.

```
[edit services cos rule rule-name]
user@host# set term term-name from source-address-range source-address-range low minimum-
value high maximum-value <except>
```

g. Specify the source prefix list for rule matching.

```
[edit services cos rule rule-name]
user@host# set term term-name from source-prefix-list list-name <except>
```

NOTE: You must configure the source prefix list by including the `prefix-list` statement at the `[edit policy-options]` hierarchy level.

Configuring Actions in a CoS Rule

IN THIS SECTION

The principal CoS actions are:

- `dscp`—Marks the packet with the specified DiffServ code point (DSCP) value or alias.
- `forwarding-class`—Assigns the packet to the specified forwarding class.

This section describes how to configure these CoS actions and includes the following topics:

Configuring Application Profiles

You can optionally define one or more application profiles for inclusion in CoS actions.

The `application-profile` statement includes two main components and three traffic types: `ftp` with the `data` traffic type and `sip` with the `video` and `voice` traffic types. You can set the appropriate `dscp` and `forwarding-class` values for each component within the application profile.

NOTE: The `ftp` and `sip` statements are not supported on Juniper Network MX Series 5G Universal Routing Platforms.

You can apply the application profile to a CoS configuration by including it at the `[edit services cos rule` *`rule-name`* `term` *`term-name`* `then]` hierarchy level.

To configure an application profile for inclusion in CoS actions:

1. Specify the `application-profile` statement at the `[edit services cos]` hierarchy level.

   ```
   [edit]
   user@host# edit services cos application-profile profile-name
   ```

2. Specify the appropriate `dscp` and `forwarding-class` value for FTP traffic.

   ```
   [edit services cos application-profile profile-name]
   user@host# set ftp data dscp (alias | bits)
   user@host# set ftp data forwarding-class class-name
   ```

3. Specify the appropriate `dscp` and `forwarding-class` value for SIP video traffic.

   ```
   [edit services cos application-profile profile-name]
   user@host# set sip video dscp (alias | bits)
   user@host# set sip video forwarding-class class-name
   ```

4. Specify the appropriate `dscp` and `forwarding-class` value for SIP voice traffic.

   ```
   [edit services cos application-profile profile-name]
   user@host# set sip voice dscp (alias | bits)
   user@host# set sip voice forwarding-class class-name
   ```

Configuring Reflexive and Reverse CoS Actions

It is important to understand that CoS services are unidirectional. It might be necessary to specify different treatments for flows in opposite directions.

Regardless of whether a packet matches the input, output, or input-output direction, flows in both directions are created. The difference is that a forward, reverse, or forward-and-reverse CoS action is associated with each flow. You should bear in mind that the flow in the opposite direction might end up having a CoS action associated with it, which you have not specifically configured.

To control the direction in which service is applied, separate from the direction in which the rule match is applied, you can configure the `reflexive` or `reverse` statement at the `[edit services cos rule` *`rule-name`* `term` *`term-name`* `then]` hierarchy level.

These two actions are mutually exclusive. If nothing is specified, data flows inherit the CoS behavior of the forward control flow.

- `reflexive` causes the equivalent reverse CoS action to be applied to flows in the opposite direction.

- `reverse` allows you to define the CoS behavior for flows in the reverse direction.

To control the direction in which a service is applied:

1. Define the CoS term actions.

   ```
   [edit]
   user@host# edit services cos rule rule-name term term-name then
   ```

2. Specify the action.

   ```
   [edit services cos rule rule-name term term-name then]
   user@host# set reflexive; | revert; | reverse {
   ```

3. Specify the application profile name.

   ```
   [edit services cos rule rule-name term term-name then]
   user@host# set application-profile profile-name
   ```

4. Define the Differentiated Services code point (DSCP) mapping that is applied to the packets.

   ```
   [edit services cos rule rule-name term term-name then]
   user@host# set dscp (alias | bits)
   ```

5. Define the forwarding class to which packets are assigned.

   ```
   [edit services cos rule rule-name term term-name then]
   user@host# set forwarding-class class-name
   ```

6. (Optional) Set the configuration to record information in the system logging facility.

   Define the forwarding class to which packets are assigned.

   ```
   [edit services cos rule rule-name term term-name then]
   user@host# set syslog
   ```

Example: Configuring CoS Rules on Services PICs

The following example show a CoS configuration containing two rules, one for input matching on a specified application set and the other for output matching on a specified source address:

```
[edit services]
cos {
    application-profile cosprofile {
        ftp {
            data {
                dscp af11;
                forwarding-class 1;
            }
        }
    }
    application-profile cosrevprofile {
        ftp {
            data {
                dscp af22;
            }
        }
    }
    rule cosrule {
        match-direction input;
        term costerm {
            from {
                source-address {
                    any-unicast;
                }
                applications junos-ftp;
            }
            then {
                dscp af33;
                forwarding-class 3;
                application-profile cosprofile;
                reverse {
                    dscp af43;
                    application-profile cosrevprofile;
                }
            }
        }
    }
}
stateful-firewall {
    rule r1 {
        match-direction input;
        term t1 {
            from {
                application-sets junos-algs-outbound;
            }
            then {
                accept;
            }
        }
        term t2 {
            then {
                accept;
            }
        }
    }
    service-set test {
        stateful-firewall-rules r1;
        cos-rules cosrule;
        interface-service {
            service-interface sp-1/3/0;
        }
    }
}
```

RELATED DOCUMENTATION

Class of Service Overview

Restrictions and Cautions for CoS Configuration on Services Interfaces

Configuring CoS Rule Sets

Examples: Configuring CoS on Services Interfaces

CoS on Services PICs Overview | **884**

Configuring CoS Rule Sets on Services PICs | **895**

Configuring CoS Rule Sets on Services PICs

You can define a collection of CoS rules that determine what actions the router software performs on packets in the data stream. Junos OS processes the rules in the order in which you specify them in the configuration. If a term in a rule matches the packet, the router performs the corresponding action and the rule processing stops. If no term in a rule matches the packet, processing continues to the next rule in the rule set. If none of the rules match the packet, the packet is dropped by default.The `rule-set` statement defines a collection of CoS rules that determine what actions the router software performs on packets in the data stream. You define each rule by specifying a rule name and configuring terms. You then specify the order of the rules by including the `rule-set` statement at the `[edit services cos]` hierarchy level:

This topic explains how to configure a set of CoS rules.

Before starting this procedure, make sure you define the terms and actions for the CoS rules in this rule set under the `[edit services cos rule rule-name]` hierarchy level.

To configure a CoS rule set:

1. Specify a name for the CoS rule set.

```
[edit]
user@host# edit services cos rule-set rule-set-name
```

2. Specify the name of each rule you want included in the rule set.

> **NOTE**: Junos OS processes the rules in the order in which you specify them in the configuration.

```
[edit services cos rule-set rule-set-name]
user@host# set rule rule-name1
user@host# set rule rule-name2
```

RELATED DOCUMENTATION

CoS on Services PICs Overview | 884

Configuring CoS Rules on Services PICs | 886

Packet Rewriting on Services Interfaces

You can configure *rewrite rules* to change packet header information and attach it to an output interface. Because these rules can possibly overwrite the DSCP marking configured on Multiservices and Services PICs, it is important to create system-wide configurations carefully.

For example, knowing that the Services or Multiservices PICs can mark packets with any ToS or DSCP value and the output interface is restricted to only eight DSCP values, rewrite rules on the output interface condense the mapping from 64 to 8 values with overall loss of granularity. In this case, you have the following options:

- Remove rewrite rules in the output interface.
- Configure the output interface to include the most important mappings.

RELATED DOCUMENTATION

Rewriting Packet Headers to Ensure Forwarding Behavior | 558

Configuring Rewrite Rules | 564

Fragmentation by Forwarding Class Overview

For Multiservices and Services *Physical Interface Card* (PIC) link services IQ (LSQ) and virtual LSQ redundancy (`rlsq-`) interfaces, you can specify fragmentation properties for specific forwarding classes. Traffic on each forwarding class can be either multilink fragmented or interleaved. By default, traffic in all forwarding classes is fragmented.

If you do not configure fragmentation properties for particular forwarding classes in multilink Point-to-Point Protocol (MLPPP) interfaces, the fragmentation threshold you set at the `[edit interfaces` *`interface-name`* `unit` *`logical-unit-number`* `fragment-threshold]` hierarchy level is used for all forwarding classes within the MLPPP interface. If you do not set a maximum fragment size anywhere in the configuration, packets are still fragmented if they exceed the smallest maximum transmission unit (MTU) of all the links in the bundle.

To configure fragmentation by forwarding class, include the following statements at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
fragmentation-maps {
    map-name {
        forwarding-class class-name {
            drop-timeout milliseconds;
            fragment-threshold bytes;
            multilink-class number;
            no-fragmentation;
        }
    }
}
interfaces {
    interface-name {
        unit logical-unit-number {
            fragmentation-map map-name;
        }
    }
}
```

RELATED DOCUMENTATION

Configuring Fragmentation by Forwarding Class | 897

Example: Configuring Fragmentation by Forwarding Class | 901

Configuring Drop Timeout Interval for Fragmentation by Forwarding Class | 899

fragmentation-map

fragmentation-maps

Configuring Fragmentation by Forwarding Class

For Multiservices and Services PIC link services IQ (LSQ) and virtual LSQ redundancy (rlsq-) interfaces only, you can configure fragmentation properties on a particular forwarding class.

To configure fragmentation properties on a specific forwarding class:

1. Specify the name of the fragmentation map and forwarding class.

```
[edit]
user@host# edit class-of-service fragmentation-maps map-name forwarding-class class-name
```

2. Specify how many milliseconds to wait for fragments.

```
[edit class-of-service fragmentation-maps map-name forwarding-class class-name]
user@host# set drop-timeout milliseconds
```

NOTE: If you set this value, you must also include a `multilink-class` value for resequencing fragments.

3. (Optional) Specify the maximum size, in bytes, for multilink packet fragments.

```
[edit class-of-service fragmentation-maps map-name forwarding-class class-name]
user@host# set fragment-threshold bytes
```

NOTE: If you set the option, you cannot configure `no-fragmentation` for the forwarding class.

4. Specify the multilink class assigned to this forwarding class.

```
[edit class-of-service fragmentation-maps map-name forwarding-class class-name]
user@host# set multilink-class number
```

5. (Optional) Specify that the traffic on this particular forwarding class is interleaved, rather than fragmented.

```
[edit class-of-service fragmentation-maps map-name forwarding-class class-name]
user@host# set no-fragmentation
```

6. Apply the fragmentation map to the logical interface.

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
user@host# set fragmentation-map map-name
```

RELATED DOCUMENTATION

Fragmentation by Forwarding Class Overview | 896

Configuring Drop Timeout Interval for Fragmentation by Forwarding Class

For `LSQ` interfaces configured for multiclass MLPPP, you can change the drop timeout interval that the interface waits for fragment resequencing by forwarding class. This feature is mutually exclusive with the `no-fragmentation` statement configured for a forwarding class.

You can also disable the fragment resequencing function altogether by forwarding class. You do this by setting the `drop-timeout` interval to 0.

The `drop-timeout` interval can also be set at the bundle level. When the `drop-timeout` interval is set to 0 at the bundle level, *none* of the individual classes forward fragmented packets. Sequencing is ignored also, and packets are forwarded in the order in which they were received. The `drop-timeout` interval value configured at the bundle level overrides the values configured at the class level.

This example configures a logical unit on an LSQ interface with a fragmentation map setting different drop timeout values for each forwarding class:

- Best effort (BE)—The value of 0 means that no resequencing of fragments takes place for BE traffic.
- Expedited Forwarding (EF)—The value of 800 ms means that the multiclass MLPPP waits 800 ms for fragment to arrive on the link for EF traffic.
- Assured Forwarding (AF)—The absence of the timeout statements means that the default timeouts of 500 ms are in effect for AF traffic.
- Network Control (NC)—The value of 100 ms means that the multiclass MLPPP waits 100 ms for fragment to arrive on the link for NC traffic.

To configure the drop timeout interval:

1. Define the fragmentation properties for each forwarding class.

```
[edit]
user@host# edit class-of-service fragmentation-maps Timeout_Frag_Map
user@host# set forwarding-class BE drop-timeout 0 multilink-class 3 fragment-threshold 128
```

```
user@host# set forwarding-class EF drop-timeout 800 multilink-class 2
user@host# set forwarding-class NC drop--timeout 100 multilink-class 0 fragment-threshold 512
user@host# set forwarding-class AF multilink-class 1 fragment-threshold 256
```

2. Apply the fragmentation map to the logical interface.

```
[edit class-of-service]
user@host# set interfaces lsq-1/0/0 unit 1 fragmentation-map Timeout_Frag_Map
```

3. Verify the configuration.

   Verify the Configuration of the Fragmentation Map Properties

```
[edit class-of-service fragmentation-maps Timeout_Frag_Map]
user@host# show
forwarding-class {
    BE {
        fragment-threshold 128;
        multilink-class 3;
        drop-timeout 0;  # no resequencing for this forwarding class
    }
    EF {
        multilink-class 2;
        drop-timeout 800;
    }
    NC {
        fragment-threshold 512;
        multilink-class 0;
        drop-timeout 100;
    }
    AF {
        fragment-threshold 256;  # Default timeout in effect for this class
        multilink-class 1;
    }
}
```

   Verify the Fragmentation Map is Applied to the Logical Interface

```
[edit class-of-service]
user@host# show
interfaces {
 lsq-1/0/0 {
        unit 1 {
            fragmentation-map Tineout_frag_Map;
        }
    }
```

4. Save the configuration.

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

RELATED DOCUMENTATION

Example: Configuring Fragmentation by Forwarding Class | **901**

Configuring Fragmentation by Forwarding Class | **897**

Example: Configuring Fragmentation by Forwarding Class

IN THIS SECTION

This example shows you how to configure fragmentation maps for specific forwarding classes on Multiservices PICs or Services PICs.

Requirements

This example uses the following hardware and software components:

- Multiservices PIC or Services PIC.

Overview

Configure two logical units on an LSQ interface. The logical units use two different fragmentation maps.

Configuration

IN THIS SECTION

To configure fragmentation maps for specific forwarding classes, perform these tasks:

CLI Quick Configuration

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

Define the Fragmentation Maps

```
set class-of-service fragmentation-maps frag-map-A forwarding-class AF no-fragmentation
set class-of-service fragmentation-maps frag-map-A forwarding-class EF no-fragmentation
set class-of-service fragmentation-maps frag-map-A forwarding-class BE fragment-threshold 100
set class-of-service fragmentation-maps frag-map-B forwarding-class EF fragment-threshold 200
set class-of-service fragmentation-maps frag-map-B forwarding-class BE fragment-threshold 200
set class-of-service fragmentation-maps frag-map-B forwarding-class AF fragment-threshold 200
```

Associate the Fragmentation Map with an Interface

```
set class-of-service interfaces lsq-1/0/0 unit 1 fragmentation-map frag-map-A
set class-of-service interfaces lsq-1/0/0 unit 2 fragmentation-map frag-map-B
```

Define the Fragmentation Maps

Step-by-Step Procedure

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

To define the fragmentation maps:

1. Specify a name for the first fragmentation map.

   ```
   [edit]
   user@host# edit class-of-service fragmentation-maps frag-map-A
   ```

2. Define the first fragmentation map.

   a. Define the fragmentation properties for the AF forwarding class to be interleaved, rather than fragmented.

      ```
      [edit class-of-service fragmentation-maps frag-map-A]
      user@ost# set forwarding-class AF no-fragmentation
      ```

   b. Define the fragmentation properties for the EF forwarding class to be interleaved, rather than fragmented.

      ```
      [edit class-of-service fragmentation-maps frag-map-A]
      user@ost# set forwarding-class EF no-fragmentation
      ```

   c. Define the fragmentation properties for the BE forwarding class to be fragmented.

      ```
      [edit class-of-service fragmentation-maps frag-map-A]
      user@ost# set forwarding-class BE fragment-threshold 100
      ```

3. Define the second fragmentation map.

a. Specify a name for the second fragmentation map.

```
[edit class-of-service fragmentation-maps]
user@host# edit frag-map-B
```

b. Define the fragmentation properties for the EF forwarding class to be fragmented.

```
[edit class-of-service fragmentation-maps frag-map-B]
user@ost# set forwarding-class EF fragment-threshold 200
```

c. Define the fragmentation properties for the BE forwarding class to be fragmented.

```
[edit class-of-service fragmentation-maps frag-map-B]
user@ost# set forwarding-class BEfragment-threshold 200
```

d. Define the fragmentation properties for the AF forwarding class to be fragmented.

```
[edit class-of-service fragmentation-maps frag-map-B]
user@ost# set forwarding-class AF fragment-threshold 200
```

Results

Verify the configuration of the fragmentation maps and forwarding classes.

```
[edit class-of-service fragmentation-maps]
user@host# show
frag-map-A {
    forwarding-class {
        AF {
            no-fragmentation;
        }
        EF {
            no-fragmentation;
        }
        BE {
            fragment-threshold 100;
        }
    }
}
frag-map-B {
    forwarding-class {
        EF {
            fragment-threshold 200;
        }
        BE {
            fragment-threshold 200;
        }
        AF {
            fragment-threshold 200;
        }
    }
}
```

Associate the Fragmentation Maps with the MLPPP Interface

Step-by-Step Procedure

To associate a fragmentation map with an interface:

1. Associate each fragmentation map with a logical interface.

   ```
   [edit]
   user@host# edit class-of-service interfaces lsq-1/0/0
   user@host# set unit 1 fragmentation-map frag-map-A
   user@host# set unit 2 fragmentation-map frag-map-B
   ```

Results

Verify that the fragmentation maps are associated with the interfaces.

```
[edit class-of-service]
user@host# show
interfaces {
 lsq-1/0/0 {
        unit 1 {
            fragmentation-map frag-map-A;
        }
        unit 2 {
```

```
            fragmentation-map frag-map-B;
        }
    }
```

Verification

IN THIS SECTION

Verifying the Fragmentation Properties

Purpose

Verify the fragmentation properties for specific forwarding classes.

Action

The following output displays the fragmentation properties and forwarding class association.

```
user@host> show class-of-service fragmentation-map
  Fragmentation map: frag-map-A, Index: 19801
    Forwarding class: AF
    No Fragmentation

     Forwarding class: EF
    No Fragmentation

    Forwarding class: BE
    Fragmentation threshold: 100

  Fragmentation map: frag-map-B, Index: 19855
    Forwarding class: EF
    Fragmentation threshold: 200

     Forwarding class: BE
    Fragmentation threshold: 200
    Forwarding class: AF
    Fragmentation threshold: 200
```

Meaning

The output shows the forwarding class associated with each fragmentation map, as well as the fragmentation properties associated with the forwarding class.

RELATED DOCUMENTATION

Fragmentation by Forwarding Class Overview | 896

Configuring Fragmentation by Forwarding Class | 897

Configuring Rate Limiting and Sharing of Excess Bandwidth on Multiservices PICs

On Multiservices PICs, you can limit the transmit rate of a logical interface (`lsq-`) in the same way as other types of queuing PICs. You can also assign a percentage of the excess bandwidth to the logical interfaces. As with other types of PICs, the strict-high queue (voice) can "starve" low and medium priority queues. To prevent the strict-high queue from starving other queues, rate-limit the queue.

To rate-limit logical interfaces on a Multiservices PIC, include the `transmit-rate` statement with the `rate-limit` option at the `[edit class-of-service schedulers scheduler-name]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
transmit-rate (rate | percent percentage | remainder) rate-limit;
```

You can also make the excess strict-high bandwidth available for other queues. You can split the excess bandwidth among multiple queues, but the total excess bandwidth assigned to these queues can only add up to 100 percent. The excess-bandwidth `priority` statement option is not supported on the Multiservices PIC.

To share excess bandwidth among Multiservices PICs, include the `excess-rate` statement at the `[edit class-of-service schedulers scheduler-name]` hierarchy level.

```
[edit class-of-service schedulers scheduler-name]
excess-rate percent percentage;
```

Both of these rate-limiting and excess bandwidth sharing features apply to egress traffic only, and only for per-unit schedulers. Hierarchical schedulers and shared schedulers are not supported.

You must still complete the configuration by configuring the scheduler map and applying it to the Multiservices PIC interface.

This example configures a rate limit and excess bandwidth sharing for a Multiservices PIC interface.

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

1. Specify the scheduler name and parameter values.

```
[edit]
user@host# edit class-of-service
user@host# set schedulers scheduler0 transmit-rate percent 10 rate-limit
user@host# set schedulers scheduler0 priority strict-high excess-rate percent 30
user@host# set schedulers scheduler1 transmit-rate percent 1 rate-limit
user@host# set schedulers scheduler1 priority high excess-rate percent 70
```

2. Specify a scheduler map name and associate it with the scheduler configuration and forwarding class.

```
[edit class-of-service]
user@host# set scheduler-maps scheduler0 forwarding-class ef scheduler scheduler0
user@host# set scheduler-maps scheduler0 forwarding-class af scheduler scheduler1
```

3. Associate the scheduler map name with the interface.

```
[edit class-of-service]
user@host# set interfaces lsq-1/3/0 unit 0 scheduler-map scheduler0
user@host# set interfaces lsq-1/3/0 unit 1 scheduler-map scheduler1
```

4. Verify the configuration.

```
scheduler0 {
    transmit-rate {
        percent 10;
        rate-limit;
    }
    excess-rate percent 30;
    priority strict-high;
}
scheduler1 {
    transmit-rate {
        percent 1;
        rate-limit;
    }
    excess-rate percent 70;
    priority high;
}
```

```
[edit class-of-service]
user@host# show schedulers
```

```
interfaces {
 lsq-1/3/0 {
        unit 0 {
            scheduler-map scheduler0;
        }
    }
```

```
scheduler-maps {
 scheduler0 {
        forwarding-class ef scheduler scheduler0;
        forwarding-class af scheduler scheduler1;
    }
}
```

RELATED DOCUMENTATION

CoS on Services PICs Overview | **884**

Configuring Schedulers | **348**

Configuring Scheduler Maps | **348**

Excess Rate and Excess Priority Configuration Examples | **381**

CHAPTER 17

Configuring Class of Service on 10-Gigabit Ethernet LAN/WAN PICs with SFP+

IN THIS CHAPTER

CoS on 10-Gigabit Ethernet LAN/WAN PIC with SFP+ Overview

The 10-Gigabit Ethernet LAN/WAN PIC with SFP+ supports intelligent handling of oversubscribed traffic in applications, such as data centers and dense-core uplinks. The 10-Gigabit Ethernet LAN/WAN PIC with SFP+ supports line-rate operation for five 10-Gigabit Ethernet ports from each port group or a total WAN bandwidth of 100 Gbps with Packet Forwarding Engine bandwidth of 50 Gbps.

NOTE: This PIC has a front panel label with the designation "ETHERNET 10GBASE-SFP+ LAN-WAN" and can also be identified by its model number, PD-5-10XGE-SFPP. It is referred to hereafter as the 10-Gigabit Ethernet LAN/WAN PIC.

The 10-Gigabit Ethernet LAN/WAN PICs support behavior aggregate (BA) and fixed classification, weighted round-robin scheduling with two queue priorities (low and strict-high), committed and peak information rate shaping on a per-queue basis, and excess information rate configuration for allocation of excess bandwidth.

The 10-Gigabit Ethernet LAN/WAN PICs have the following features to support queuing:

- Committed and peak information rate shaping on a per-queue basis
- Excess information rate configuration for allocation of excess bandwidth
- Ingress queuing based on behavior aggregate (BA) classification

- Egress queuing at the Packet Forwarding Engine and at the PIC level

  The Packet Forwarding Engine egress queues are shared by two physical interfaces in a port group.

- Weighted round-robin (WRR) scheduling with two queue priorities (low and strict-high)

- Two special queues available in ingress, one per physical interface, called *control queues*

  Layer 2 and Layer 3 control protocol packets (OSPF, OSPF3, VRRP, IGMP, RSVP, PIM, BGP, BFD, LDP, ISIS, RIP, RIPV6, LACP, ARP, IPv6 NDP, CFM, and LFM) are mapped to the control queue. In the control queue, these packets are not dropped even if there is oversubscription or congestion on a port group.

  NOTE: The control queue is rate-limited to 2 Mbps per physical interface. The packets in excess of 2 Mbps are dropped and accounted for.

To configure these features, include the corresponding class-of-service (CoS) statements at the `[edit class-of-service]` hierarchy level. The CoS statements supported on the 10-Gigabit Ethernet LAN/WAN PICs are shown in Table 80 on page 911.

Table 80: CoS Statements Supported on the 10-Gigabit Ethernet LAN/WAN PICs

CoS Statements	Supported
`buffer-size`	No
`drop-profile-map`	No
`excess-priority`	No
`excess-rate`	Yes
`priority`	Yes
`shaping-rate`	Yes
`transmit-rate`	Yes

RELATED DOCUMENTATION

Junos OS Network Interfaces Library for Routing Devices

BA and Fixed Classification on 10-Gigabit Ethernet LAN/WAN PIC with SFP+ Overview

The 10-Gigabit Ethernet LAN/WAN PICs support the following behavior aggregate (BA) classifiers:

- DSCP, DSCP IPv6, or IP precedence—IP packet classification (Layer 3 headers)
- MPLS EXP—MPLS packet classification (Layer 2 headers)
- IEEE 802.1p—Packet classification (Layer 2 headers)
- IEEE 802.1ad—Packet classification for IEEE 802.1ad formats (including DEI bit)

Multiple classifiers can be configured to a single *logical interface*. However, there are some restrictions on which the classifiers can coexist. For example, the DSCP and IP precedence classifiers cannot be configured on the same logical interface. The DSCP and IP precedence classifiers can coexist with the DSCP IPv6 classifier on the same logical interface. An IEEE 802.1 classifier can coexist with other classifiers and is applicable only if a packet does not match any of the configured classifiers. For information about the supported combinations, see "Applying Behavior Aggregate Classifiers to Logical Interfaces" on page 83.

If the classifiers are not defined explicitly, then the default classifiers are applied as follows:

- All MPLS packets are classified using the MPLS (EXP) classifier. If there is no explicit MPLS (EXP) classifier, then the default MPLS (EXP) classifier is applied.
- All IPv4 packets are classified using the IP precedence and DSCP classifiers. If there is no explicit IP precedence or DSCP classifier, then the default IP precedence classifier is applied.
- All IPv6 packets are classified using a DSCP IPv6 classifier. If there is no explicit DSCP IPv6 classifier, then the default DSCP IPv6 classifier is applied.
- If the IEEE 802.1p classifier is configured and a packet does not match any explicitly configured classifier, then the IEEE 802.1p classifier is applied.

The fixed classification matches the traffic on a logical interface level. The following example classifies all traffic on logical unit zero to the queue corresponding to assured forwarding.

```
[edit class-of-service interfaces xe-0/1/2 unit 0]
forwarding-class fc-af11;
```

NOTE: The 10-Gigabit Ethernet LAN/WAN PICs do not support multifield classification. However, the multifield classification can be done at the Packet Forwarding Engine using the firewall filters, which overrides the classification done at the PIC level. The multifield classification at the Packet Forwarding Engine occurs after the PIC handles the oversubscribed traffic.

Scheduling and Shaping on 10-Gigabit Ethernet LAN/WAN PICs Overview

The 10-Gigabit Ethernet LAN/WAN PIC has ten 10-Gigabit Ethernet ports providing 100 Gbps of WAN bandwidth and 50 Gbps of Packet Forwarding Engine bandwidth. On the ingress side of the 10-Gigabit Ethernet LAN/WAN PIC, two consecutive physical interfaces on the PICs are grouped together into a port group and are serviced by a single scheduler. The port groups are as shown in Table 81 on page 913:

Table 81: Port Groups on 10-Gigabit Ethernet LAN/WAN PICs

Port Group	Mapped Ports
Group 1	xe-*fpc*/*pic*/0 xe-*fpc*/*pic*/1
Group 2	xe-*fpc*/*pic*/2 xe-*fpc*/*pic*/3
Group 3	xe-*fpc*/*pic*/4 xe-*fpc*/*pic*/5

Table 81: Port Groups on 10-Gigabit Ethernet LAN/WAN PICs *(Continued)*

Port Group	Mapped Ports

Group 4	`xe-`*fpc*`/`*pic*`/6`
	
	`xe-`*fpc*`/`*pic*`/7`

Group 5	`xe-`*fpc*`/`*pic*`/8`
	
	`xe-`*fpc*`/`*pic*`/9`

The two physical interfaces in a port group share 10 Gbps bandwidth towards the Packet Forwarding Engine. A scheduler has eight class-of-service (CoS) queues and two control queues. On the ingress side of the 10-Gigabit Ethernet LAN/WAN PIC, the eight CoS queues are split four plus four for the two physical interfaces. Thus, the 10-Gigabit Ethernet LAN/WAN PIC supports four ingress queues and eight egress queues per physical interface.

At the ingress side of the 10-Gigabit Ethernet LAN/WAN PIC, multiple forwarding classes can be mapped to one queue using the restricted-queue configuration. When creating a scheduler-map for the ingress queues, only one forwarding class should be chosen from the multiple forwarding classes that map to the same queue. Then, the scheduler-map can be specified using the `set class-of-service scheduler-maps` *`map-name`* `forwarding-class` *`class-name`* `scheduler` *`scheduler`* command.

The 10-Gigabit Ethernet LAN/WAN PICs manage packet buffering internally and no configuration is required.

NOTE: The delay-bandwidth buffering configuration is not supported on the 10-Gigabit Ethernet LAN/WAN PICs.

Example: Configuring Shaping Overhead on 10-Gigabit Ethernet LAN/WAN PICs

By default, the 10-Gigabit Ethernet LAN/WAN PIC uses 20 bytes as the shaping overhead. This includes 8 bytes preamble and 12 bytes interpacket gap (IPG) in shaper operations. To exclude this overhead, it should be configured as –20 bytes. The shaping overhead value can be set between 0 and 31 bytes, as shown in the following example. This range translates to a CLI range of –20 to 11 bytes for the shaping overhead configuration.

```
show chassis
    fpc 6 {
    pic 0 {
        traffic-manager {
        ingress-shaping-overhead -20;
        egress-shaping-overhead -20;
        }
    }
}
```

NOTE: When the configuration for the overhead bytes on a PIC are changed, the PIC is taken offline and then brought back online. In addition, the configuration in the CLI is on a per-PIC basis, and thus, applies to all the ports on the PIC.

RELATED DOCUMENTATION

Scheduling and Shaping on 10-Gigabit Ethernet LAN/WAN PICs Overview | **913**

CHAPTER 18

Configuring Class of Service on Enhanced Queuing DPCs

IN THIS CHAPTER

Enhanced Queuing DPC CoS Properties

On a Juniper Networks MX Series 5G Universal Routing Platform with Enhanced Queuing Dense Port Concentrators (DPCs), you can configure schedulers and queues. You can configure 15 VLAN sets per Gigabit Ethernet (1G) port and 255 VLAN sets per 10-Gigabit Ethernet (10G) port. The Enhanced Queuing DPC performs priority propagation from one hierarchy level to another and drop statistics are available on the Enhanced Queuing DPC per color per queue instead or just per queue.

NOTE: The Enhanced Queuing DPC (EQ DPC) does not support BA classification for packets received from a Layer 3 routing interface or a virtual routing and forwarding (VRF) interface and routed to an integrated routing and bridging interface (IRB) to reach the remote end of a pseudowire connection. The EQ DPC also does not support BA classification for Layer 2 frames received from a Virtual Private LAN Service (VPLS) pseudowire connection from a remote site and routed to a Layer 3 routing interface through an IRB interface.

Juniper Networks MX Series 5G Universal Routing Platforms with Enhanced Queuing DPCs have Packet Forwarding Engines that can support up to 515 MB of frame memory, and packets are stored in 512-

byte frames. Table 82 on page 917 compares the major properties of the Intelligent Queuing 2 (IQ2) PIC and the Packet Forwarding Engine within the Enhanced Queuing DPC.

Table 82: IQ2 PIC and Enhanced Queuing DPC Compared

Feature	IQ2 PIC	Packet Forwarding Engine Within Enhanced Queuing DPC
Number of usable queues	8,000	16,000
Number of shaped logical interfaces	1,000 with 8 queues each.	2,000 with 8 queues each, or 4,000 with 4 queues each.
Number of hardware priorities	2	4
Priority propagation	No	Yes
Dynamic mapping	No: schedulers/port are fixed.	Yes: schedulers/port are not fixed.
Drop statistics	Per queues	Per queue per color (PLP high, low)

In addition, the Enhanced Queuing DPC features support for hierarchical weighted random early detection (WRED) and enhanced queuing on aggregated Ethernet interfaces with link protection as well.

The Enhanced Queuing DPC supports the following hierarchical scheduler characteristics:

- Shaping at the physical interface level
- Shaping and scheduling at the service VLAN interface set level
- Shaping and scheduling at the customer VLAN *logical interface* level
- Scheduling at the queue level

VLAN (Level 3) shaping on a 10-Gigabit Ethernet MX Series Enhanced Queuing DPC differs from the VLAN (Level3) shaping on a 1-Gigabit Ethernet Enhanced Queuing DPC. To use the VLAN (Level 3) shaping on a 10-Gigabit Ethernet MX Series Enhanced Queuing DPC, configure an interface set at the `[edit interfaces interface-set]` hierarchy level. The interface set configuration is not required for configuring a 1-Gigabit Ethernet VLANs on the same Enhanced Queuing DPC.

The Enhanced Queuing DPC supports the following features for scalability:

- 16,000 queues per Packet Forwarding Engine

- 4 Packet Forwarding Engines per DPC
    - 4000 schedulers at logical interface level (Level 3) with 4 queues each
    - 2000 schedulers at logical interface level (Level 3) with 8 queues each
- 255 schedulers at the interface set level (Level 2) per 1-port Packet Forwarding Engine on a 10-Gigabit Ethernet DPC
- 15 schedulers at the interface set level (Level 2) per 10-port Packet Forwarding Engine on a 1-Gigabit Ethernet DPC
- About 400 milliseconds of buffer delay (this varies by packet size and if large buffers are enabled)
- 4 levels of priority (strict-high, high, medium, and low)

NOTE: Including the `transmit-rate` *`rate`* `exact` statement at the `[edit class-of-service schedulers` *`scheduler-name`*`]` hierarchy level is not supported on Enhanced Queuing DPCs on MX Series routers.

The way that the Enhanced Queuing DPC maps a queue to a scheduler depends on whether 8 queues or 4 queues are configured. By default, a scheduler at level 3 has 4 queues. Level 3 scheduler X controls queue X*4 to X*4+3, so that scheduler 100 (for example) controls queues 400 to 403. However, when 8 queues per scheduler are enabled, the odd numbered schedulers are disabled, allowing twice the number of queues per subscriber as before. With 8 queues, level 3 scheduler X controls queue X*4 to X*4+7, so that scheduler 100 (for example) now controls queues 400 to 407.

You configure the `max-queues-per-interface` statement to set the number of queues at `4` or `8` at the FPC level of the hierarchy. Changing this statement results in a restart of the DPC. For more information about the `max-queues-per-interface` statement, see the Junos OS Network Interfaces Library for Routing Devices.

The Enhanced Queuing DPC maps level 3 (customer VLAN) schedulers in groups to level 2 (service VLAN) schedulers. Sixteen contiguous level 3 schedulers are mapped to level 2 when 4 queues are enabled, and 8 contiguous level 3 schedulers are mapped to level 2 when 8 queues are enabled. All of the schedulers in the group should use the same queue priority mapping. For example, if the queue priorities of one scheduler are high, medium, low, and low, then all members of this group should have the same queue priority.

Mapping of a group at level 3 to level 2 can be done at any time. However, a group at level 3 can only be unmapped from a level 2 scheduler only if all the schedulers in the group are free. Once unmapped, a level 3 group can be remapped to any level 2 scheduler. There is no restriction on the number of level 3 groups that can be mapped to a particular level 2 scheduler. There can be 256 level 3 groups, but fragmentation of the scheduler space can reduce the number of schedulers available. In other words, there are scheduler allocation patterns that might fail even though there are free schedulers.

In contrast to level-3-to-level-2 mapping, the Enhanced Queuing DPC maps level 2 (service VLAN) schedulers in a fixed mode to level 1 (physical interface) schedulers. On 40-port Gigabit Ethernet DPCs, there are 16 level 1 schedulers, and 10 of these are used for the physical interfaces. There are 256 level 2 schedulers, or 16 per level 1 scheduler. A level 1 scheduler uses level schedulers X*16 through X*16+15. So level 1 scheduler 0 uses level 2 schedulers 0 through 15, level 1 scheduler 1 uses level 2 schedulers 16 through 31, and so on. On 4-port 10-Gigabit Ethernet PICs, there is one level 1 scheduler for the physical interface, and 256 level 2 schedulers are mapped to the single level 1 scheduler.

The maximum number of level 3 (customer VLAN) schedulers that can be used is 4076 (4 queues) or 2028 (8 queues) for the 10-port Gigabit Ethernet Packet Forwarding Engine and 4094 (4 queues) or 2046 (8 queues) for the 10-Gigabit Ethernet Packet Forwarding Engine.

Enhanced Queuing is supported on aggregated Ethernet (AE) interfaces with two links in link protection mode. However, only one link in the AE bundle can be active at a time. Traffic is shaped independently on the two links, but the member's links do not need to reside in the same Packet Forwarding Engine or the same DPC. Finally, shared schedulers are not supported on the Enhanced Queuing DPC (use hierarchical schedulers to group logical interfaces).

RELATED DOCUMENTATION

Configuring Customer VLAN (Level 3) Shaping on Enhanced Queuing DPCs | **931**

Configuring Rate Limits on Enhanced Queuing DPCs

You can rate-limit the strict-high and high queues on the Enhanced Queuing DPC. Without rate limits, traffic in higher-priority queues can block the transmission of lower-priority packets. Unless limited, higher-priority traffic is always sent before lower-priority traffic, causing the lower-priority queues to "starve" and cause timeouts and unnecessarily resent packets.

On the Enhanced Queuing DPC, you can rate-limit queues before the packets are queued for output. All packets exceeding the configured rate limit are dropped, so care is required when establishing this limit.

NOTE: Rate limiting is implemented differently on Enhanced Queuing DPCs and non-queuing Packet Forwarding Engines. On Enhanced Queuing DPCs, rate limiting is implemented using a single-rate two-color policer. On nonqueuing Packet Forwarding Engines, rate limiting is achieved by shaping the queue to the transmit rate and keeping the queue delay buffers small to prevent too many packets from being queued after the shaping rate is reached.

To rate-limit queues, include the `transmit-rate` statement with the `rate-limit` option at the `[edit class-of-service schedulers scheduler-name]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
transmit-rate rate rate-limit;
```

The following example limits the transmit rate of a strict-high expedited-forwarding queue to 1 Mbps. The scheduler and scheduler map are defined, and then applied to the traffic at the `[edit interfaces]` and `[edit class-of-service]` hierarchy levels:

```
[edit class-of-service]
schedulers {
    scheduler-1 {
        transmit-rate 1m rate-limit; # This establishes the limit
        priority strict-high;
    }
}
scheduler-maps {
    scheduler-map-1 {
        forwarding-class expedited-forwarding scheduler scheduler-1;
    }
}

[edit interfaces]
s0-2/2/0 {
    per-unit-scheduler;
    encapsulation frame-relay;
    unit 0 {
        dlci 1;
    }
}

[edit class-of-service]
interfaces {
    so-2/2/0 {
        unit 0 {
            scheduler-map scheduler-map-1;
            shaping-rate 2m;
        }
    }
}
```

You can issue the following operational mode commands to verify your configuration (the first shows the rate limit in effect):

- `show class-of-service scheduler-map` *`scheduler-map-name`*
- `show class-of-service interface` *`interface-name`*

You can issue the `show interfaces queue` *`interface-name`* command to view the number of packets dropped at an interface. The output of the `show interfaces queue` *`interface-name`* command always displays the rate-limit counter fields whether or not rate limiting is configured on the queue. Rate-limit counters are displayed in two columns. The first column is the consolidated count of the packets dropped and the second column is the real-time count of the packets dropped.

Rate-limit packet drop counters display the value 0 when rate limiting is not configured on the queue or when the queue does not have rate-limit packet drops even with rate limiting configured.

Rate-limit packet drop counters display meaningful values in both columns when the queue has rate-limit packet drops. However, when rate limiting is not happening in real time but has occurred earlier, the first column displays the consolidated count and the second column displays the value 0.

You can clear the packet drop statistics by using the `clear interface statistics` *`interface-name`* command.

Configuring WRED on Enhanced Queuing DPCs

Shaping to drop out-of-profile traffic is done on the Enhanced Queuing DPC at all levels but the queue level. However, weighed random early detection (WRED) is done at the queue level with much the same result. With WRED, the decision to drop or send the packet is made before the packet is placed in the queue.

WRED shaping on the Enhanced Queuing DPC involves only two levels. The probabilistic drop region establishes a minimum and a maximum queue depth. Below the minimum queue depth, the drop probability is 0 (send). Above the maximum level, the drop probability is 100 (certainty).

There are four drop profiles associated with each queue. These correspond to each of four loss priorities (low, medium-low, medium-high, and high). Sixty-four sets of four drop profiles are available (32 for ingress and 32 for egress). In addition, there are eight WRED scaling profiles in each direction.

To configure WRED, include the `drop-profiles` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
drop-profiles {
    profile-name {
        fill-level percentage drop-probability percentage;
    }
}
```

The following example is an Enhanced Queuing DPC drop profile for expedited forwarding traffic:

```
[edit class-of-service drop-profiles]
drop-ef {
    fill-level 20 drop-probability 0; # Minimum Q depth
    fill-level 100 drop-probability 100; # Maximum Q depth
}
```

Note that only two fill levels can be specified for the Enhanced Queuing DPC. You can configure the `interpolate` statement, but only two fill levels are used. The `delay-buffer-rate` statement in the traffic control profile determines the maximum queue size. This delay buffer rate is converted to a packet delay buffers, where one buffer is equal to 512 bytes. For example, at 10 Mbps, the Enhanced Queuing DPC allocates 610 delay buffers when the delay-buffer rate is set to 250 milliseconds. The WRED threshold values are specified in terms of absolute buffer values.

The WRED scaling factor multiples all WRED thresholds (both minimum and maximum) by the value specified. There are eight values in all: 1, 2, 4, 8, 16, 32, 64, and 128. The WRED scaling factor is chosen to best match the user-configured drop profiles. This is done because the hardware supports only certain values of thresholds (all values must be a multiple of 16). So if the configured value of a threshold is 500 (for example), the multiple of 16 is 256 and the scaling factor applied is 2, making the value 512, which allows the value of 500 to be used. If the configured value of a threshold is 1500, the multiple of 16 is 752 and the scaling factor applied is 2, making the value 1504, which allows the value of 1500 to be used

Hierarchical RED is used to support the oversubscription of the delay buffers (WRED is only configured at the queue, physical interface, and PIC level). Hierarchical RED works with WRED as follows:

- If any level accepts the packet (the queue depth is less than the minimum buffer level), then this level accepts the packet.
- If any level probabilistically drops the packet, then this level drops the packet.

However, these rules might lead to the accepting of packets under loaded conditions which might otherwise have been dropped. In other words, the logical interface accepts packets if the physical interface is not congested.

RELATED DOCUMENTATION

RED Drop Profiles for Congestion Management | **462**

Configuring MDRR on Enhanced Queuing DPCs

The guaranteed rate (CIR) at the interface set level is implemented using modified deficit round-robin (MDRR). The Enhanced Queuing DPC hardware provides four levels of strict priority. There is no restriction on the number of queues for each priority. MDRR is used among queues of the same priority. Each queue has one priority when it is under the guaranteed rate and another priority when it is over the guaranteed rate but under the shaping rate (PIR). The Enhanced Queuing DPC hardware implements the priorities with 256 service profiles. Each service profile assigns eight priorities for eight queues. One set is for logical interfaces under the guaranteed rate and another set is for logical interfaces over the guaranteed rate but under the shaping rate. Each service profile is associated with a group of 16 level 3 schedulers, so there is a unique service profile available for all 256 groups at level 3, giving 4096 logical interfaces.

The Junos OS provides three priorities for traffic under the guaranteed rate and one reserved priority for traffic over the guaranteed rate that is not configurable. The Junos OS provides three priorities when there is no guaranteed rate configured on any logical interface.

The relationship between Junos OS priorities and the Enhanced Queuing DPC hardware priorities below and above the guaranteed rate (CIR) is shown in Table 83 on page 923.

Table 83: Junos OS Priorities Mapped to Enhanced Queuing DPC Hardware Priorities

Junos OS Priority	Enhanced Queuing DPC Hardware Priority Below Guaranteed Rate	Enhanced Queuing DPC Hardware Priority Above Guaranteed Rate
Strict-high	High	High
High	High	Low
Medium-high	Medium-high	Low
Medium-low	Medium-high	Low
Low	Medium-low	Low

To configure MDRR, configure a scheduler at the `[edit class-of-service schedulers]` hierarchy level:

```
[edit class-of-service schedulers]
scheduler-name {
    buffer-size (seconds | percent percentage | remainder | temporal microseconds);
    priority priority-level;
    transmit-rate (percent percentage | rate | remainder) <exact | rate-limit>;
}
```

The following example creates two schedulers for MDRR:

```
[edit class-of-service schedulers]
best-effort-scheduler {
    transmit-rate percent 30; # if no shaping rate
    buffer-size percent 30;
    priority high;
}
expedited-forwarding-scheduler {
    transmit-rate percent 40; # if no shaping rate
    buffer-size percent 40;
    priority strict-high;
}
```

NOTE: The use of both shaping rate and a guaranteed rate at the interface set level (level 2) is not supported.

MDRR is provided at three levels of the scheduler hierarchy of the Enhanced Queuing DPC with a granularity of 1 through 255. There are 64 MDRR profiles at the queue level, 16 at the interface set level, and 32 at the physical interface level.

Queue transmit rates are used for queue level MDRR profile weight calculation. The queue MDRR weight is calculated differently based on the mode set for sharing excess bandwidth. If you configure the `equal` option for excess bandwidth, then the queue MDRR weight is calculated as:

Queue weight = (255 * Transmit-rate-percentage) / 100

If you configure the `proportional` option for excess bandwidth, which is the default, then the queue MDRR weight is calculated as:

Queue weight = Queue-transmit-rate / Queue-base-rate, where

Queue-transmit-rate = (Logical-interface-rate * Transmit-rate-percentage) / 100, and

Queue-base-rate = Excess-bandwidth-proportional-rate / 255

To configure the way that the Enhanced Queuing DPC should handle excess bandwidth, configure the `excess-bandwidth-share` statement at the `[edit interface-set *interface-set-name*]` hierarchy level. By default, the excess bandwidth is set to `proportional` with a default value of 32.64 Mbps. In this mode, the excess bandwidth is shared in the ratio of the logical interface shaping rates. If set to `equal`, the excess bandwidth is shared equally among the logical interfaces.

This example sets the excess bandwidth sharing to proportional at a rate of 100 Mbps with a shaping rate of 80 Mbps.

```
[edit interface-set example-interface-set]
excess-bandwidth-share proportional 100m;
output-traffic-control-profile PIR-80Mbps;
```

Shaping rates established at the logical interface level are used to calculate the MDRR weights used at the interface set level. The 16 MDRR profiles are set to initial values, and the closest profile with rounded values is chosen. By default, the physical port MDRR weights are preset to the full bandwidth on the interface.

Configuring Excess Bandwidth Sharing

IN THIS SECTION

When using the Enhanced Queuing DPC on an MX Series router, there are circumstances when you should configure excess bandwidth sharing and minimum logical interface shaping. This section details some of the guidelines for configuring excess bandwidth sharing.

Excess Bandwidth Sharing and Minimum Logical Interface Shaping

The default excess bandwidth sharing proportional rate is 32.65 Mbps (128 Kbps x 255). In order to have better weighed fair queuing (WFQ) accuracy among queues, the shaping rate configured should be larger than the excess bandwidth sharing proportional rate. Some examples are shown in Table 84 on page 926.

Table 84: Shaping Rates and WFQ Weights

Shaping Rate	Configured Queue Transmit Rate	WFQ Weight	Total Weights
10 Mbps	(30, 40, 25, 5)	(22, 30, 20, 4)	76
33 Mbps	(30, 40, 25, 5)	(76, 104, 64, 13)	257
40 Mbps	(30, 40, 25, 5)	(76, 104.64, 13)	257

With a 10-Mbps shaping rate, the total weights are 76. This is divided among the four queues according to the configured transmit rate. Note that when the shaping rate is larger than the excess bandwidth sharing proportional rate of 32.65 Mbps, the total weights on the logical interface are 257 and the WFQ accuracy is the same.

Selecting Excess Bandwidth Sharing Proportional Rates

A good excess bandwidth sharing proportional rate to configure is to choose the largest CIR (guaranteed rate) among all the logical interfaces (units). If the logical units have PIRs (shaping rates) only, then choose the largest PIR rate. However, this is not ideal if a single logical interface has a large weighed round-robin (WRR) rate. This can skew the distribution of traffic across the queues of the other logical interfaces. To avoid this issue, set the excess bandwidth sharing proportional rate to a lower value on the logical interfaces where the WRR rates are concentrated. This improves the bandwidth sharing accuracy among the queues on the same logical interface. However, the excess bandwidth sharing for the logical interface with the larger WRR rate is no longer proportional.

As an example, consider five logical interfaces on the same physical port, each with four queues, all with only PIRs configured and no CIRs. The WRR rate is the same as the PIR for the logical interface. The excess bandwidth is shared proportionally with a rate of 40 Mbps. The traffic control profiles for the logical interfaces are shown in Table 85 on page 926.

Table 85: Sample Shaping Rates and WFQ Weights

Shaping Rate	Configured Queue Transmit Rate	WFQ Weight	Total Weights
(Unit 0) 10 Mbps	(95, 0, 0, 5)	(60, 0, 0, 3)	63
(Unit 1) 20 Mbps	(25, 25, 25, 25)	(32, 32, 32, 32)	128

Table 85: Sample Shaping Rates and WFQ Weights *(Continued)*

Shaping Rate	Configured Queue Transmit Rate	WFQ Weight	Total Weights
(Unit 2) 40 Mbps	(40, 30, 20, 10)	(102, 77, 51, 26)	255
(Unit 3) 200 Mbps	(70, 10, 10, 10)	(179, 26, 26, 26)	255
(Unit 4) 2 Mbps	(25, 25, 25, 25)	(5, 5, 5, 5)	20

Even though the maximum transmit rate for the queue on logical interface unit 3 is 200 Mbps, the excess bandwidth sharing proportional rate is kept at a much lower value. Within a logical interface, this method provides a more accurate distribution of weights across queues. However, the excess bandwidth is now shared equally between unit 2 and unit 3 (total weight of each = 255).

Mapping Calculated Weights to Hardware Weights

The calculated weight in a traffic control profile is mapped to hardware weight, but the hardware only supports a limited WFQ profile. The weights are rounded to the nearest hardware weight according to the values in Table 86 on page 927.

Table 86: Rounding Configured Weights to Hardware Weights

Traffic Control Profile Number	Number of Traffic Control Profiles	Weights	Maximum Error
1–16	16	1–16 (interval of 1)	50.00%
17–29	13	18–42 (interval of 2)	6.25%
30–35	6	45–60 (interval of 3)	1.35%
36–43	8	64–92 (interval of 4)	2.25%
44–49	6	98–128 (interval of 6)	3.06%
50–56	7	136–184 (interval of 8)	3.13%

Table 86: Rounding Configured Weights to Hardware Weights *(Continued)*

Traffic Control Profile Number	Number of Traffic Control Profiles	Weights	Maximum Error
57–62	6	194–244 (interval of 10)	2.71%
63–63	1	255–255 (interval of 11)	2.05%

From the table, as an example, the calculated weight of 18.9 is mapped to a hardware weight of 18, because 18 is closer to 18.9 than 20 (an interval of 2 applies in the range 18–42).

Allocating Weight with Only Shaping Rates or Unshaped Logical Interfaces

Logical interfaces with only shaping rates (PIRs) or unshaped logical interfaces (units) are given a weight of 10. A logical interface with a small guaranteed rate (CIR) might get an overall weight less than 10. In order to allocate a higher share of the excess bandwidth to logical interfaces with a small guaranteed rate in comparison to the logical interfaces with only shaping rates configured, a minimum weight of 20 is given to the logical interfaces with guaranteed rates configured.

For example, consider a logical interface configuration with five units, as shown in Table 87 on page 928.

Table 87: Allocating Weights with PIR and CIR on Logical Interfaces

Logical Interface (Unit)	Traffic Control Profile	WRR Percentages	Weights
Unit 1	PIR 100 Mbps	95, 0, 0, 5	10, 1, 1, 1
Unit 2	CIR 20 Mbps	25, 25, 25, 25	64, 64, 64, 64
Unit 3	PIR 40 Mbps, CIR 20 Mbps	50, 30, 15, 5	128, 76, 38, 13
Unit 4	Unshaped	95, 0, 0, 5	10, 1, 1, 1
Unit 5	CIR 1 Mbps	95, 0, 0, 5	10, 1, 1, 1

The weights for these units are calculated as follows:

- Select the excess bandwidth sharing proportional rate to be the maximum CIR among all the logical interfaces: 20 Mbps (unit 2).

- Unit 1 has a PIR and unit 4 is unshaped. The weight for these units is 10.
- The weight for unit 1 queue 0 is 9.5 (10 x 95%), which translates to a hardware weight of 10.
- The weight for unit 1 queue 1 is 0 (0 x 0%), but although the weight is zero, a weight of 1 is assigned to give minimal bandwidth to queues with zero WRR.
- Unit 5 has a very small CIR (1 Mbps), and a weight of 20 is assigned to units with a small CIR.
- The weight for unit 5 queue 0 is 19 (20 x 95%), which translates to a hardware weight of 18.
- Unit 3 has a CIR of 20 Mbps, which is the same as the excess bandwidth sharing proportional rate, so it has a total weight of 255.
- The weight of unit 3 queue 0 is 127.5 (255 x 50%), which translates to a hardware weight of 128.

Sharing Bandwidth Among Logical Interfaces

As a simple example showing how bandwidth is shared among the logical interfaces, assume that all traffic is sent on queue 0. Assume also that there is a 40-Mbps load on all of the logical interfaces. Configuration details are shown in Table 88 on page 929.

NOTE: On the MX960 router, bandwidth sharing across high priority and strict-high priority schedulers configured on logical interfaces might not be as expected. This is a hardware limitation.

Table 88: Sharing Bandwidth Among Logical Interfaces

Logical Interface (Unit)	Traffic Control Profile	WRR Percentages	Weights
Unit 1	PIR 100 Mbps	95, 0, 0, 5	10, 1, 1, 1
Unit 2	CIR 20 Mbps	25, 25, 25, 25	64, 64, 64, 64
Unit 3	PIR 40 Mbps, CIR 20 Mbps	50, 30, 15, 5	128, 76, 38, 13
Unit 4	Unshaped	95, 0, 0, 5	10, 1, 1, 1

1. When the port is shaped at 40 Mbps, because units 2 and 3 have a guaranteed rate (CIR) configured, both units 2 and 3 get 20 Mbps of shared bandwidth.

2. When the port is shaped at 100 Mbps, because units 2 and 3 have a guaranteed rate (CIR) configured, each of them can transmit 20 Mbps. On units 1, 2, 3, and 4, the 60 Mbps of excess bandwidth is shaped according to the values shown in Table 89 on page 930.

Table 89: First Example of Bandwidth Sharing

Logical Interface (Unit)	Calculation	Bandwidth
Unit 1	10 / (10+64+128+10) x 60 Mbps	2.83 Mbps
Unit 2	64 / (10+64+128+10) x 60 Mbps	18.11 Mbps
Unit 3	128 / (10+64+128+10) x 60 Mbps	36.22 Mbps
Unit 4	10 (10+64+128+10) x 60 Mbps	2.83 Mbps

However, unit 3 only has 20 Mbps extra (PIR and CIR) configured. This means that the leftover bandwidth of 16.22 Mbps (36.22 Mbps – 20 Mbps) is shared among units 1, 2, and 4. This is shown in Table 90 on page 930.

Table 90: Second Example of Bandwidth Sharing

Logical Interface (Unit)	Calculation	Bandwidth
Unit 1	10 / (10+64+128+10) x 16.22 Mbps	1.93 Mbps
Unit 2	64 / (10+64+128+10) x 16.22 Mbps	12.36 Mbps
Unit 4	10 (10+64+128+10) x 16.22 Mbps	1.93 Mbps

Finally, Table 91 on page 930 shows the resulting allocation of bandwidth among the logical interfaces when the port is configured with a 100-Mbps shaping rate.

Table 91: Final Example of Bandwidth Sharing

Logical Interface (Unit)	Calculation	Bandwidth
Unit 1	2.83 Mbps + 1.93 Mbps	4.76 Mbps

Table 91: Final Example of Bandwidth Sharing *(Continued)*

Logical Interface (Unit)	Calculation	Bandwidth
Unit 2	20 Mbps + 18.11 Mbps + 12.36 Mbps	50.47 Mbps
Unit 3	20 Mbps + 20 Mbps	40 Mbps
Unit 4	2.83 Mbps + 1.93 Mbps	4.76 Mbps

Configuring Customer VLAN (Level 3) Shaping on Enhanced Queuing DPCs

Customer VLAN (level 3) shaping on an MX Series 10-Gigabit Ethernet Enhanced Queuing DPC differs from the customer VLAN (level 3) shaping on an MX Series 1-Gigabit Ethernet Enhanced Queuing DPC. To use the customer VLAN (level 3) shaping on an MX Series 10-Gigabit Ethernet Enhanced Queuing DPC, configure an interface set at the `[edit interfaces interface-set]` hierarchy level. You do not need to configure the interface set while using customer VLAN (level 3) on an MX Series 1-Gigabit Ethernet Enhanced Queuing DPC.

To configure customer VLAN (level 3) shaping on an MX Series 10-Gigabit Ethernet Enhanced Queuing DPC:

1. Configure the interface set at the `[edit interfaces]` hierarchy level.

```
[edit interfaces]
user@host# set interface-set jnpr interface unit 100
user@host# set interface-set jnpr interface xe-1/0/0 unit 101
```

2. Configure the hierarchical scheduler and enable VLAN tagging.

```
[edit interfaces]
user@host# set xe-1/0/0 hierarchical-scheduler
user@host# set xe-1/0/0 vlan-tagging
```

3. Configure the logical interface properties.

```
[edit interfaces]
user@host# set xe-1/0/0 unit 100 vlan-id 100
user@host# set xe-1/0/0 unit 100 family inet address 10.1.0.1/24
user@host# set xe-1/0/0 unit 101 vlan-id 101
user@host# set xe-1/0/0 unit 101 family inet address 10.1.1.1/24
```

4. Configure the traffic control profiles at the `[edit class-of-service]` hierarchy level.

```
[edit class-of-service]
user@host# set traffic-control-profiles profile1 shaping-rate 10g burst-size 2k
user@host# set traffic-control-profiles profile1 guaranteed-rate 10g burst-size 2k
user@host# set traffic-control-profiles profile2 shaping-rate 50m burst-size 2k
user@host# set traffic-control-profiles profile2 guaranteed-rate 50m burst-size 2k
user@host# set traffic-control-profiles profile3 shaping-rate 80m burst-size 3k
user@host# set traffic-control-profiles profile3 guaranteed-rate 80m burst-size 3k
```

5. Configure the output traffic control profiles at the `[edit class-of-service interfaces]` hierarchy level.

```
[edit class-of-service interfaces]
user@host# set interface-set jnpr output-traffic-control-profiles profile1
user@host# set xe-1/0/0 unit 100 output-traffic-control-profiles profile2
user@host# set xe-1/0/0 unit 101 output-traffic-control-profiles profile3
```

To configure customer VLAN (level 3) shaping on an MX Series 1-Gigabit Ethernet Enhanced Queuing DPC:

1. Configure the interface set at the `[edit interfaces]` hierarchy level.

```
[edit interfaces]
user@host# set interface ge-1/0/0 unit 100
user@host# set interface ge-1/0/0 unit 101
```

2. Configure the hierarchical scheduler and enable the VLAN tagging.

```
[edit interfaces]
user@host# set ge-1/0/0 hierarchical-scheduler
user@host# set ge-1/0/0 vlan-tagging
```

3. Configure the logical interface properties.

```
[edit interfaces]
user@host# set ge-1/0/0 unit 100 vlan-id 100
user@host# set ge-1/0/0 unit 100 family inet address 10.1.0.1/24
user@host# set ge-1/0/0 unit 101 vlan-id 101
user@host# set ge-1/0/0 unit 101 family inet address 10.1.1.1/24
```

4. Configure the traffic control profiles at the `[edit class-of-service]` hierarchy level.

```
[edit class-of-service]
user@host# set traffic-control-profiles profile1 shaping-rate 10g burst-size 2k
user@host# set traffic-control-profiles profile1 guaranteed-rate 10g burst-size 2k
user@host# set traffic-control-profiles profile2 shaping-rate 50m burst-size 2k
user@host# set traffic-control-profiles profile2 guaranteed-rate 50m burst-size 2k
user@host# set traffic-control-profiles profile3 shaping-rate 80m burst-size 3k
user@host# set traffic-control-profiles profile3 guaranteed-rate 80m burst-size 3k
```

5. Configure the traffic control profiles at the `[edit class-of-service interfaces]` hierarchy level.

```
[edit class-of-service interfaces]
user@host# set ge-1/0/0 unit 100 output-traffic-control-profiles profile2
user@host# set ge-1/0/0 unit 101 output-traffic-control-profiles profile3
```

RELATED DOCUMENTATION

Enhanced Queuing DPC CoS Properties | 916

Configuring Simple Filters on Enhanced Queuing DPCs

You can configure and apply a simple filter to perform multifield classification on the ingress interfaces of an MX Series router with Enhanced Queuing DPCs. These simple filters can be used to override default CoS classification parameters such as forwarding class or loss priority. Simple filters, in contrast to other firewall filters, only support a subset of the full firewall filter syntax.

To configure a simple filter, include the `simple-filter` statement at the `[edit firewall family inet]` hierarchy level:

```
[edit firewall family inet]
simple-filter filter-name {
    term term-name {
        from {
            ... match-conditions...
        }
        then {
            forwarding-class class-name;
            loss-priority priority;
        }
    }
}
```

The following example configures a simple filter to detect ingress packets from various source addresses (`10.1.1.1/32`, `10.10.10.10/32`, and `10.4.0.0/8`), destination addresses (`10.6.6.6/32`), protocols (`tcp`), and source ports (`400-500`, `http`). The filter then assigns various forwarding classes and loss priorities to the filtered traffic. Finally, the filter is applied to the input side of an Enhanced Queuing DPC interface (`ge-2/3/3`).

```
[edit]
firewall {
    family inet {
        simple-filter sf-for-eq-dpc {
            term 1 {
                from {
                    source-address 10.1.1.1/32;
                    protocol tcp;
                }
                then loss-priority low;
            }
            term 2 {
                from {
                    source-address 10.4.0.0/8;
                    source-port http;
                }
                then loss-priority high;
            }
            term 3 {
                from {
```

```
                destination-address 10.6.6.6/32;
                source-port 400-500;
            }
            then {
                loss-priority low;
                forwarding-class best-effort;
            }
        }
        term 4 {
            from {
                forwarding-class expedited-forwarding;
                source-address 10.10.10.10/32;
            }
            then loss-priority low;
        }
        term 5 {
            from {
                source-address 10.10.10.10/32;
            }
            then loss-priority low;
        }
    }
  }
}
interfaces { # Apply the simple filter above to the input side of the interface.
ge-2/3/3 {
    unit 0 {
        family inet {
            simple-filter {
                input sf-for-eq-dpc;
            }
        }
    }
}
```

RELATED DOCUMENTATION

Simple Filter Overview

How Simple Filters Evaluate Packets

Guidelines for Configuring Simple Filters

Guidelines for Applying Simple Filters

CHAPTER 19

Configuring Class of Service on MICs, MPCs, and MLCs

IN THIS CHAPTER

CoS Features and Limitations on MIC and MPC Interfaces

MIC and MPC interfaces on MX Series 5G Universal Routing Platforms use the Trio chipset-based queuing model, which supports CoS characteristics that are optimized compared to CoS characteristics supported by the standard queuing model. However, some CoS features are not supported or are supported with limitations on MIC and MPC interfaces.

When configuring CoS features on MIC and MPC interfaces on MX Series routers, keep the following limitations in mind.

Table 92: CoS Limitations on MIC and MPC Interfaces

CoS Feature	Limitation on MIC or MPC Interfaces
Classifiers	Interfaces on MPCs support up to 32 classifiers of each type per module.

Table 92: CoS Limitations on MIC and MPC Interfaces *(Continued)*

CoS Feature	Limitation on MIC or MPC Interfaces
BA classifier for MPLS pack ets	When you configure a behavior aggregate (BA) classifier that does not include a specific rewrite rule for MPLS packets, we highly recommend that you enable the default MPLS EXP classifier. Doing so ensures that MPLS exp value is rewritten according to the BA classifier rules configured for forwarding or packet loss priority. For more information, see "Default MPLS EXP Classifier" on page 66. To enable the default MPLS EXP classifier, include the `default` statement at the `[edit class-of-service interfaces` *`interface-name`* `unit` *`logical-unit-number`* `rewrite-rules exp]` hierarchy level.
Rewrite rules	For interfaces on MPCs or on MICs installed in MPCs, you can figure up to 32 rewrite rules: • DSCP rewrite rules • Internet rewrite rules • EXP rewrite rules • IEEE rewrite rules However, if you configure all 32 allowed rewrite rules, the class-of-service process can intermittently fail and generate syslog entries.
Default rewrite rules for MPLS-enabled interfaces	On interfaces other than MIC and MPC interfaces, the default EXP rewrite rule is automatically applied to MPLS-enabled interfaces, even if not configured. On MIC and MPC interfaces, you must explicitly configure EXP rewrite rules to MPLS-enabled interfaces.
Rewrite rules for service VLAN tag CoS bits	For MIC and MPC interfaces for VPLS or bridge domains, rewrite service VLAN tag CoS bits by configuring the rewrite rules on the *core-facing* interface.

Table 92: CoS Limitations on MIC and MPC Interfaces *(Continued)*

CoS Feature	Limitation on MIC or MPC Interfaces
Excess bandwidth sharing	Interfaces on MICs and MPCs do not support the `excess-bandwidth-share` *configuration statement*, which specifies how excess bandwidth at an interface set in a hierarchical scheduler environment is to be shared: proportionally or equally. Instead, you can include the `excess-rate` statement at one of the following hierarchy levels: • `[edit class-of-service schedulers` *`scheduler-name`*`]` • `[edit class-of-service traffic-control-profiles` *`traffic-control-profile-name`*`]`
Layer 1 and Layer 2 overhead	MIC and MPC interfaces take all Layer 1 and Layer 2 overhead bytes into account for all levels of the hierarchy, including preamble, interpacket gaps, frame check sequence, and cyclical redundancy check. Queue statistics also take these overheads into account when displaying byte statistics.
Pairing of load-balancing links	When load balancing EQ MIC interfaces installed in Type 1 MPCs, you should configure odd- and even-numbered interfaces in the form *`interface-fpc/odd`* | *`even`*`/ports`. For example, if one link is `xe-1/0/0`, the other should be `xe-1/1/0`. If you do not configure odd and even load balancing, the system RED-drops packets when sending at line rate. This limitation does not apply to interfaces on EQ MICs installed in Type 2 MPCs.

RELATED DOCUMENTATION

Rate Shaping on MIC and MPC Interfaces | 968

Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 992

Scheduler Delay Buffering on MIC and MPC Interfaces | 1007

Drop Profiles on MIC and MPC Interfaces | 1011

Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview

IN THIS SECTION

Queuing Ethernet Modular Port Concentrators (MPCs) provide a set of dedicated queues for subscriber interfaces configured with hierarchical scheduling or per-unit scheduling.

The dedicated queues offered on these MPCs enable service providers to reduce costs through different scaling configurations. These queuing MPCs enable service providers to reduce the cost per subscriber by allowing many subscriber interfaces to be created with four or eight queues.

This topic describes the overall queue, scheduler node, and *logical interface* scaling for subscriber interfaces created on these MIC and MPC combinations.

Queue Scaling for MPCs

Beginning with Junos OS Release 15.1, MPC2E-3D-NG-Q, MPC3E-3D-NG-Q, MPC5EQ-40G10G, and MPC5EQ-100G10G MPCs support up to five levels of hierarchical queuing. Beginning with Junos OS Release 16.1R1, MPC7 line cards also support five levels of hierarchical queuing. Table 93 on page 941 lists the number of dedicated queues and nodes supported per MPC.

Table 93: Dedicated Queues for MPCs

MPC	Dedicated Queues	Level 4 Nodes	Level 3 Nodes	Level 2 Nodes	Level 1 Nodes (Ports)
MPC2E-3D-NG-Q MPC3E-3D-NG-Q	512,000	64,000	16,000	4000	384

Table 93: Dedicated Queues for MPCs *(Continued)*

MPC	Dedicated Queues	Level 4 Nodes	Level 3 Nodes	Level 2 Nodes	Level 1 Nodes (Ports)
MPC5EQ-40G10G MPC5EQ-100G10G	1 million	128,000	32,000	4000	384
MPC7	512,000	64,000	16000	8000	252
MPC10E on MX10K series platforms	256,000	32,000	8,000	4,000	128

CAUTION: The maximum scaling targets provided in Table 93 on page 941 are based on system level design specifications. Actual realized subscriber or session scale is highly dependent upon the configuration and can be influenced by configuration variables including: the number of routes, the number of enabled services, the number of policy and firewall filters, policers, counters, statistics and access model type. Once you define a configuration, your Juniper account team can help characterize the expected system level scale or scale range for your live deployment.

MPCs vary in the number of Packet Forwarding Engines on board. MPC2E-3D-NG-Q and MPC3E-3D-NG-Q MPCs each have one Packet Forwarding Engine, allowing all 64,000 level 4 (subscriber) nodes to be allocated to a single MIC. MPC5EQ MPCs have two Packet Forwarding Engines, one for each possible MIC, each supporting 64,000 level 4 (subscriber) nodes. MPC7 MPCs also have two Packet Forwarding Engines, one for each possible MIC, each supporting 256,000 dedicated queues and 32,000 level 4 (subscriber) nodes.

NOTE: The nonqueuing MPCs MPC2E-3D-NG, MPC3E-3D-NG, MPC5E-40G10G, and MPC5E-100G10G provide up to eight queues per port in standard configuration. However, each of these MPCs can be configured to provide limited-scale hierarchical class of service (HCoS) and up to 32,000 queues.

Managing Remaining Queues

In Junos OS releases earlier than Release 15.1R4, SNMP traps generate system log messages to notify you:

- When the number of available dedicated queues on the MPC drops below 10 percent. For example:

  ```
  Mar 15 14:55:22.977 host cosd[1963]: COSD_OUT_OF_DEDICATED_QUEUES: Queue usage count for
  interface xe-3/0/0 is at 90 percent
  ```

- When the maximum number of dedicated queues on the MPCs is reached. For example,

  ```
  Mar 15 18:01:59.344 host cosd[3848]: COSD_OUT_OF_DEDICATED_QUEUES: Queue usage count for
  interface xe-3/0/0 is at 100 percent.
  ```

  When the maximum number of dedicated queues is allocated, the system does not provide subsequent subscriber interfaces with a dedicated set of queues. For per-unit scheduling configurations, there are no configurable queues remaining on the MPC.

For hierarchical scheduling configurations, remaining queues are available when the maximum number of dedicated queues is reached on the MPC. Traffic from these logical interfaces is considered unclassified and attached to a common set of queues that are shared by all subsequent logical interfaces. These common queues are the default port queues that are created for every port. You can configure a traffic-control profile and attach that to the interface to provide CoS parameters for the remaining queues. These subscriber interfaces remain with this traffic-control profile, even if dedicated queues become available.

NOTE: Starting in Junos OS Release 15.1R4, the COSD_OUT_OF_DEDICATED_QUEUES functionality is not available for QoS-enabled dynamic subscribers. Starting in Junos OS Release 17.4R1, CoS resource monitoring enables you to set a per-FPC queue threshold of up to 90 percent of resources bound to a scheduling hierarchy; subscriber logins are not allowed when the threshold is reached. However, this threshold applies to all queues, not dedicated queues alone. See Resource Monitoring for Subscriber Management and Services Overview for more information.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
16.1R1	Beginning with Junos OS Release 16.1R1, MPC7 line cards also support five levels of hierarchical queuing.
15.1R1	Beginning with Junos OS Release 15.1, MPC2E-3D-NG-Q, MPC3E-3D-NG-Q, MPC5EQ-40G10G, and MPC5EQ-100G10G MPCs support up to five levels of hierarchical queuing.

RELATED DOCUMENTATION

Hierarchical Class of Service User Guide

Understanding Hierarchical Scheduling

Managing Dedicated and Remaining Queues for Static CoS Configurations on MIC and MPC Interfaces

Understanding Hierarchical Scheduling for MIC and MPC Interfaces

Verifying the Number of Dedicated Queues Configured on MIC and MPC Interfaces

IN THIS SECTION

Purpose

Display the number of dedicated queue resources that are configured for the logical interfaces on a port.

Action

```
user@host#show class-of-service interface ge-1/1/0
Physical interface: ge-1/1/0, Index: 166
Queues supported: 4, Queues in use: 4
Total non-default queues created: 4
  Scheduler map: <default>, Index: 2
  Chassis scheduler map: <default-chassis>, Index: 4

  Logical interface: ge-1/1/0.100, Index: 72, Dedicated Queues: no
    Shaping rate: 32000
    Object              Name                   Type                    Index
    Scheduler-map       <remaining>                                    0
    Classifier          ipprec-compatibility   ip                      13

  Logical interface: ge-1/1/0.101, Index: 73, Dedicated Queues: no
    Shaping rate: 32000
    Object              Name                   Type                    Index
    Scheduler-map       <remaining>                                    0
    Classifier          ipprec-compatibility   ip                      13

  Logical interface: ge-1/1/0.102, Index: 74, Dedicated Queues: yes
    Shaping rate: 32000
    Object                   Name               Type                   Index
    Traffic-control-profile  <control_tc_prof>  Output                 45866
```

RELATED DOCUMENTATION

Managing Dedicated and Remaining Queues for Static CoS Configurations on MIC and MPC Interfaces

Managing Dedicated and Remaining Queues for Dynamic CoS Configurations on MIC and MPC Interfaces

Scaling of Per-VLAN Queuing on Non-Queuing MPCs

Per-VLAN (logical interface) queueing has been introduced on most MPCs supported on the MX platform. Table 94 on page 946 shows the details along with the supported JUNOS release.

Table 94: MPC and MIC support for per-VLAN (logical interface) queuing

MPC	MICs Supported	JUNOS Release
16x10GE MPC	N/A	13.2
MPC3E	2x10GE XFP	13.2
	10x10GE SFPP	13.2
	2x40G QSFPP	13.2
	1x100GE CXP	13.2
	1x100G CFP	13.2
MPC4E-32x10GE SFPP	N/A	13.3
MPC4E-2x100GE+8x10GE SFPP	N/A	13.3
MPC6E	24x10GE SFPP	15.1
	24x10GE SFFP OTN	15.1
	2x100GE CFP2 OTN	15.1
	4x100GE CXP	15.1
MPC5E-10G100G	N/A	13.3R3
MPC5E-10G40G	N/A	13.3R3
MPC2E-3D-NG/MPC3E-3D-NG	20x1GE SFP	15.1

Table 94: MPC and MIC support for per-VLAN (logical interface) queuing *(Continued)*

MPC	MICs Supported	JUNOS Release
	2xGE-XFP	15.1
	40x1GE	15.1
	4xGE-XFP	15.1
	8OC3OC12-4OC48	15.1
	4OC3OC12-1OC48	15.1
	8CHOC3-4CHOC12	15.1
	4CHOC3-1CHOC12	15.1
	8DS3-E3	15.1
	1xOC192-XFP	15.1
	4COC3-1COC12-CE	15.1
	20xGE-SFP-E	15.1
MPC3E-3D-NG	2x10GE XFP	15.1
	10x10GE SFPP	15.1
	2x40G QSFPP	15.1
	1x100GE CXP	15.1

To enable logical interface scheduling, you include the `per-unit-scheduler` statement at the `[edit interfaces` *`interface name`*`]` hierarchy level. When per-unit schedulers are enabled, you can define dedicated schedulers for logical interfaces by including the `scheduler-map` statement at the `[edit class-of-service interfaces` *`interface name`* `unit` *`logical unit number`*`]` hierarchy level. Alternatively, you can include the `scheduler-map` statement at the `[edit class-of-service traffic-control-profiles` *`traffic control profile name`*`]` hierarchy level and then include the `output-traffic-control-profile` statement at the `[edit class-of-service interfaces` *`interface name`* `unit` *`logical unit number`*`]` hierarchy level.

Table 95 on page 948 shows the number of VLANs per port available in both 8-queue and 4-queue mode for 16x10GE, MPC3E, MPC4E and MPC6E.

Table 95: Number of VLANs on 16x10G, MPC3E, MPC4E and MPC6E

MPC	MIC	VLANs/Port – 8-Queue Mode	VLANs/Port – 4-Queue Mode
16X10GE	No	21	44
MPC3E	2x10GE with XFP	20	42
	10X10GE with SFP+	12 per group of 5 ports*	34 per group of 5 ports*
	2X40GE with QSFP+	20	42
	1X100GE with CXP	20	42
32x10GE MPC4E	No	20 per group of 4 ports*	48 per group of 4 ports*
2x100GE + 8x10GE MPC4E	No	26	54
MPC6E	24X10GE	20 per group of 3 ports*	42 per group of 3 ports*
	2X100GE with CFP2 OTN	26	54
	4X100GE MIC with CXP	21	44

*The 10X10GE MIC for the MPC3E, the 32X10GE MPC4E, and the 24X10GE MICs for the MPC6E share VLANs across a port group. You can assign all of the available VLANs to one port within the port group or spread them across the ports in any combination.

Enabling and configuring per-unit schedulers on these interfaces adds additional output to the `show interfaces` *`interface name`* `[detail | extensive]` command. This additional output lists the maximum resources available and the number of configured resources for schedulers. Following is sample output showing the CoS scheduler resource information on a non-queuing line card:

```
root@R1# run show interfaces et-2/2/0 detail

Physical interface: et-2/2/0, Enabled, Physical link is Down
  Interface index: 165, SNMP ifIndex: 550, Generation: 168
  Link-level type: Ethernet, MTU: 1522, Speed: 100Gbps, BPDU Error: None, Loopback: Disabled,
Source filtering: Disabled,
  Flow control: Enabled
  Device flags   : Present Running Down
  Interface flags: Hardware-Down SNMP-Traps Internal: 0x4000
  Link flags     : Scheduler
  CoS queues     : 8 supported, 8 maximum usable queues
  Schedulers     : 0
  Hold-times     : Up 0 ms, Down 0 ms
  Current address: 80:71:1f:10:e6:b4, Hardware address: 80:71:1f:10:e6:b4
  Last flapped   : 2013-05-07 16:17:01 PDT (03:16:41 ago)
  Statistics last cleared: Never
  Traffic statistics:
   Input  bytes  :                    0                    0 bps
   Output bytes  :                    0                    0 bps
   Input  packets:                    0                    0 pps
   Output packets:                    0                    0 pps
   IPv6 transit statistics:
    Input  bytes  :                   0
    Output bytes  :                   0
    Input  packets:                   0
    Output packets:                   0
  Egress queues: 8 supported, 4 in use
  CoS scheduler resource information:
    Maximum units supported per MIC/PIC: 20
    Configured units per MIC/PIC: 1
    Maximum units allowed per port: 20
    Configured units on this port: 1
  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0 best-effort                    0                    0                    0
    1 expedited-fo                   0                    0                    0
    2 assured-forw                   0                    0                    0
    3 network-cont                   0                    0                    0
  Queue number:         Mapped forwarding classes
    0                   best-effort
```

If you enable more VLANs than the previously mentioned MPC/MIC combinations support, VLANs up to the supported numbers receive dedicated queuing resources. The additional VLANs share port queues. Scheduling for port queues cannot be controlled. However, port queues are guaranteed 1 percent of the physical interface bandwidth to avoid queue starving and buffer holdup.

In the case of MPC2E-NG/3E-NG, MPC5E and MPC7E/8E/9E SKUs, the following command needs to be configured to enable “flexible queuing” on the MPC. Configuration of this knob results in a reboot of the MPC. The per-unit-scheduler, hierarchical scheduling and 2 level hierarchical scheduling are supported. There are 32K queues enabled and they can be used for either ingress queueing or egress queueing. The 32K queues are available when all 8 queues are used per IFL.

```
chassis {
     fpc 0 {
         flexible-queuing-mode;        }
            }
```

Table 96 on page 950 shows the number of VLANs per port available in both 8-queue and 4-queue mode for MPC3E-NG/MPC2E-NG, and MPC5E.

Table 96: Number of VLANs on MPC3E-NG/MPC2E-NG, MPC5E

MPC	MIC	VLANs/Port – 8-Queue Mode	VLANs/Port – 4-Queue Mode
MPC3E-NG/MPC2E-NG	Supported MICs	32K	32K
MPC5E	Supported MICs	32K	32K

NOTE: The number of logical interfaces with per-vlan queuing enabled should not exceed line card maximum. If the line card maximum is exceeded, then the queuing behavior is unpredictable. This could mean that some logical interfaces have queues assigned and some do not.

RELATED DOCUMENTATION

per-unit-scheduler

Increasing Available Bandwidth on Rich-Queuing MPCs by Bypassing the Queuing Chip

Queuing MPCs contain a queuing chip that enables rich-queuing features such as hierarchical and per-vlan queuing. By default, all traffic passing through an interface on one of these MPCs also passes through the queuing chip, which decreases the available bandwidth of the interface. If you do not require hierarchical or per-vlan queuing on a particular interface of a queuing MPC, you can bypass the queuing chip to increase the available bandwidth.

To bypass the queuing chip on an interface on a queuing MPC:

1. Ensure that neither `per-unit-scheduler` nor `hierarchical-scheduler` is configured on the interface.

   NOTE: It is not possible to bypass the queuing chip on an interface if per-unit or hierarchical scheduling is configured on that interface.

2. Ensure that *flexi-queuing-mode* is enabled.
3. Enable `bypass-queuing-chip` on the interface.

   For example:

```
[edit interfaces]
user@router# set interface- name bypass-queuing-chip
```

4. Commit your changes.

```
[edit interfaces]
user@router# show
interface-name {
    bypass-queueing-chip;
}
```

5. Verify your changes.

```
user@router> show interfaces interface-name
Physical interface: interface-name, Enabled, Physical link is Up
  Interface index: 147, SNMP ifIndex: 524
```

```
Link-level type: Ethernet, MTU: 1514, MRU: 1522, LAN-PHY mode, Speed: 1000mbps,
BPDU Error: None, MAC-REWRITE Error: None, Loopback: Disabled, Source filtering: Disabled,
Flow control: Enabled, Auto-negotiation: Enabled, Remote fault: Online
Pad to minimum frame size: Disabled
Device flags   : Present Running
Interface flags: SNMP-Traps Internal: 0x4000
Link flags     : None
CoS queues     : 8 supported, 4 maximum usable queues
Schedulers     : 0, Queuing Chip Bypassed
Current address: 00:21:59:0f:35:31, Hardware address: 00:21:59:0f:35:31
Last flapped   : 2014-04-29 14:10:18 PDT (02:27:46 ago)
Input rate     : 0 bps (0 pps)
Output rate    : 0 bps (0 pps)
Active alarms  : None
Active defects : None
Interface transmit statistics: Disabled
```

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
18.2R1	Starting with Junos OS 18.2R1, you can enable this option on vMX routers to save a vCPU when scheduling is not needed on an interface.

RELATED DOCUMENTATION

bypass-queuing-chip

Flexible Queuing Mode

IN THIS SECTION

You can configure the non-hierarchical quality-of-service (non-HQoS) MPCs to support port-based flexible queuing. By default, the non-HQoS MPCs do not support queuing. To enable queuing, you must upgrade these MPCs through an add-on license. After you upgrade these MPCs, they can support a flexible queuing capability of up to 32,000 queues per port and per card, including queues on both ingress and egress interfaces. Channelized MICs are supported on non-HQoS MPCs only when flexible queuing is configured.

Flexible Queuing Mode Overview

The queuing component on non-HQoS MPCs is disabled by default to save power. When flexible queuing is enabled on a non-HQoS MPC, the MPC is restarted with the queuing component enabled. The MPC is powered on only if the PEM has sufficient power to bring up the MPC with the queuing component enabled. The MPC remains offline if the required power is not available.

You can enable flexible queuing on the non-HQoS MPCs by including the `flexible-queuing-mode` statement at the `[edit chassis fpc]` hierarchy level. When queuing is configured, the power consumed by the queuing components at the configured ambient temperature is considered when power is allocated for the MPC.

NOTE: The following MICs are supported on non-HQoS MPCs only when flexible queuing is enabled:

- MIC-3D-8CHOC3-4CHOC12
- MIC-3D-4CHOC3-2CHOC12
- MIC-4COC3-2COC12-G
- MIC-2COC3-1COC12-G

Table 97 on page 953 lists the MPCs that support flexible queueing and the supported Junos OS release for these MPCs.

Table 97: MPCs and the Junos OS Release that Support Flexible Queuing

MPCs	First Supported Junos OS Release
MPC2E-3D-NG	15.1R1

Table 97: MPCs and the Junos OS Release that Support Flexible Queuing *(Continued)*

MPCs	First Supported Junos OS Release
MPC3E-3D-NG	15.1R1
MPC5E	14.1R1
MPC7E-MRATE	15.1F4
MPC7E-10G MPC8E MPC9E	15.1F5

Upgrading non-HQoS MPCs to Support Flexible Queuing

You can enable flexible queuing on a non-HQoS MPC to support a maximum of up to 32,000 queues per port and per card, including queues on both ingress and egress interfaces.

This topic describes how to enable flexible queuing on a non-HQoS MPC.

To configure flexible queuing on non-HQoS MPCs:

1. Run the `set chassis fpc` *`slot-number`* `flexible-queuing-mode` configuration mode command.

   For example, to configure flexible queuing on an MPC in slot 2:

   ```
   [edit]
   user@router# set chassis fpc 2 flexible-queuing-mode
   ```

   NOTE: When flexible queuing is enabled, the MPC is restarted with the queuing component enabled. The MPC comes online only if the power entry module (PEM) has sufficient power to bring up the MPC with the queuing component enabled. The MPC remains offline if the required power is not available in the PEM.

2. Review your configuration and issue the `commit` command.

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

```
[edit]
  'chassis fpc'
    WARNING: FPC configuration for flexible-queuing is changed. FPC would undergo reboot to
enable flexible-queuing. FPC would come online only if power available is sufficient to
enable queuing components.
commit complete
```

Disabling Flexible Queuing for non-HQoS MPCs to Optimize Power Utilization

You can optimize power utilization by disabling flexible queuing on a non-HQoS MPC.

This topic describes how to disable flexible queuing on a non-HQoS MPC.

1. Run the `delete chassis fpc` *`slot-number`* `flexible-queuing-mode` command at the `[edit chassis]` hierarchy level.

   For example, to disable flexible queuing on an MPC in slot 2:

   ```
   [edit]
   user@router# delete chassis fpc 2 flexible-queuing-mode
   ```

2. Review your configuration and issue the `commit` command.

   ```
   [edit]
   user@router# commit
   commit complete

   [edit]
   user@router#
   ```

RELATED DOCUMENTATION

flexible-queuing-mode

Multifield Classifier for Ingress Queuing

The multifield classifier for ingress queuing is an implementation point for firewall filters configured with specific traffic shaping actions. These filters allow you to set the forwarding class and packet loss priority for packets, or drop the packets prior to ingress queue selection. The filters are applied as ingress queue filters. You can then use CoS commands to select ingress queue, set rate limiting and so forth.

Firewall filters configured at the protocol family level are able to distinguish specific types of traffic from other types by matching on multiple fields within the packet header. The number and types of matches available depend on which protocol family is used in the filter. Before the introduction of the ingress queuing filter, these firewall filters could only be applied to traffic after the ingress queue had been selected based solely on the behavior aggregate (BA). With the introduction of the ingress queuing filter, firewall filters can be used to set forwarding classification and packet loss priority based on multiple fields within the packet header prior to forwarding queue selection. CoS functions provide traffic classification options and the ability to assign that classified traffic to specific forwarding queues.

NOTE: Ingress queuing filters are only available when the traffic manager mode is set to `ingress-and-egress` at the `[edit chassis fpc` *`fpc-id`* `pic` *`pic-id`* `traffic-manager mode]` hierarchy level.

The `ingress-queuing-filter` configuration statement is used at the `[edit interfaces` *`interface-name`* `unit` *`unit-number`* `family` *`family-name`*`]` hierarchy level to designate a previously configured firewall filter to be used as an ingress queuing filter. The following list shows which protocol families are compatible with the `ingress-queuing-filter` statement:

- `bridge`
- `ccc`
- `inet`
- `inet6`
- `mpls`
- `vpls`

The named firewall filter is a normal firewall filter that must be configured with at least one of the following actions: `accept`, `discard`, `forwarding-class`, and `loss-priority`.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
16.1	Beginning with Junos OS Release 16.1, the multifield classifier for ingress queuing is an implementation point for firewall filters configured with specific traffic shaping actions.

RELATED DOCUMENTATION

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 60

ingress-queuing-filter

Example: Configuring a Filter for Use as an Ingress Queuing Filter

Example: Configure a Filter for Use as an Ingress Queuing Filter

IN THIS SECTION

This example shows how to configure a firewall filter for use as an ingress queuing filter. The ingress queuing filter assists in traffic shaping operations by enabling you to set the forwarding class and packet loss priority, or drop the packet before ingress queue selection. The firewall filter must be configured within one of the following protocol families: `bridge`, `ccc`, `inet`, `inet6`, `mpls`, or `vpls` and have one or more of the following actions: `accept`, `discard`, `forwarding-class`, and `loss-priority`.

NOTE: Although the ingress queuing filter can be used with MX Series routers, it is used only on those MX Series routers that have MPCs. An error is generated at commit if the ingress queuing filter is applied to an interface on any other type of port concentrator.

Requirements

This example uses the following hardware and software components:

- An MX Series router with MPC

In order for ingress queuing filters to function, `ingress-and-egress` must be configured as the `traffic-manager` mode at the `[edit chassis fpc` *`slot`* `pic` *`slot`* `traffic-manager mode]` hierarchy level.

Overview

In this example, you create a firewall filter named `iqfilter1` in the `inet` protocol family that sets the loss priority and forwarding class of packets coming from the 192.168.2.0/24 network. You then apply the `iqfilter1` filter to the ge-0/0/0.0 logical interface as an ingress queuing filter.

To configure a firewall filter and apply it for use as an ingress queuing filter involves:

- Creating a firewall filter named `iqfilter1` in the `inet` protocol family with the following two actions: `forwarding-class` and `loss-priority`.
- Applying the firewall filter to the ge-0/0/0.0 interface as an ingress queuing filter.

Configuration

IN THIS SECTION

CLI Quick Configuration

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

```
set firewall family inet filter iqfilter1 term t1 from address 192.168.2.0/24
set firewall family inet filter iqfilter1 term t1 then loss-priority low
set firewall family inet filter iqfilter1 term t1 then forwarding-class expedited-forwarding
set interfaces ge-0/0/0 unit 0 family inet ingress-queuing-filter iqfilter1
```

Configuring the Firewall Filter and Applying It to an Interface as an Input Queuing Filter

Step-by-Step Procedure

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

To configure the firewall filter, `iqfilter1`, and apply it to logical interface ge-0/0/0 unit 0:

1. Create a firewall filter named `iqfilter1`.

   ```
   [edit firewall family inet]
   user@router# set  filter iqfilter1 term t1 from address 192.168.2.0/24
   user@router# set  filter iqfilter1 term t1 then loss-priority low
   user@router# set  filter iqfilter1 term t1 then forwarding-class expedited-forwarding
   ```

2. Apply the firewall filter to the logical interface.

   ```
   [edit]
   user@router# set interfaces ge-0/0/0 unit 0 family inet ingress-queuing-filter iqfilter1
   ```

Results

From configuration mode, confirm your configuration by entering the `show firewall` and the `show interfaces ge-0/0/0.0` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@router# show firewall
 family inet {
    filter iqfilter1 {
        term t1 {
            from {
                address {
                    192.168.0.0/24;
                }
            }
            then {
                loss-priority low;
                forwarding-class expedited-forwarding;
            }
        }
    }
}
user@router# show interfaces ge-0/0/0.0
family inet {
```

```
    ingress-queuing-filter iqfilter1;
}
```

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

```
user@router# commit
```

RELATED DOCUMENTATION

Multifield Classifier for Ingress Queuing on MX Series Routers with MPC

ingress-queuing-filter

Ingress Queuing Filter with Policing Functionality

IN THIS SECTION

On MPCs that support ingress queuing, you can implement policer actions, along with other filter actions, on traffic before the traffic is assigned to ingress queues. Ingress queuing policing filters allow you to rate limit traffic as well as count and set the forwarding class and packet loss priority for packets prior to ingress queue selection. You can then use CoS commands to select ingress queuing parameters.

Understanding the Ingress Queuing Policing Filter

The ingress queuing policing filter (`iq-policing-filter`) function similarly and at the same point as the ingress policing filter (`ingress-queuing-filter`), but provides the added benefit of accepting almost all filter actions, including policing and counting actions. The ingress queuing policing filter is also more efficient, requiring fewer system resources.

NOTE: Ingress queuing filters are only available when the traffic manager mode is set to `ingress-and-egress` at the `[edit chassis fpc fpc-id pic pic-id traffic-manager mode]` hierarchy level.

The `iq-policing-filter` configuration statement is used at the `[edit interfaces interface-name unit unit-number family family-name]` hierarchy level to designate a previously configured firewall filter to be used as an ingress queuing policing filter. The following list shows which protocol families are compatible with the `iq-policing-filter` statement:

- `bridge`
- `inet`
- `vpls`

The named firewall filter is a normal firewall filter that must be configured with at least one of the following actions: `count accept, discard, forwarding-class,loss-priority` and `policers`.

SEE ALSO

iq-policing-filter

Example: Configuring a Filter for Use as an Ingress Queuing Policing Filter

IN THIS SECTION

This example shows how to configure a firewall filter for use as an ingress queuing policing filter. The ingress queuing filter assists in ingress traffic policing operations by allowing you to rate limit traffic prior to ingress queue selection. The firewall filter must be configured within one of the following protocol families: `bridge`, `inet`, or `vpls`.

You can only use the ingress queuing policing filter on devices that support ingress queuing. An error is generated at commit if the ingress queuing filter is applied to an interface on any other type of port concentrator.

Requirements

This example uses the following hardware and software components:

- An MX Series router with an MPC that supports ingress queuing

In order for ingress queuing filters to function, `ingress-and-egress` must be configured as the `traffic-manager` mode at the `[edit chassis fpc *slot* pic *slot* traffic-manager mode]` hierarchy level.

Overview

In this example, you create a firewall filter named `vpls_iqp_filter` in the `vpls` protocol family that counts and polices voice and best effort traffic. You then apply the `vpls_iqp_filter` filter to the xe-0/0/0.0 logical interface as an ingress queuing policing filter.

To configure a firewall filter and apply it for use as an ingress queuing filter involves:

- Creating a firewall filter named `vpls_iqp_filter` in the `vpls` protocol family with the following actions: `count`, `forwarding- class` and `policer`.
- Applying the firewall filter to the xe-0/0/0.0 interface as an ingress queuing policing filter.

Configuration

IN THIS SECTION

CLI Quick Configuration

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

```
set firewall family vpls filter vpls_iqp_filter interface-specific
set firewall family vpls filter vpls_iqp_filter term VoiceSum from learn-vlan-1p-priority 5
set firewall family vpls filter vpls_iqp_filter term VoiceSum then count VoiceSum
set firewall family vpls filter vpls_iqp_filter term VoiceSum then forwarding-class Voice
set firewall family vpls filter vpls_iqp_filter term VoiceSum then next term
set firewall family vpls filter vpls_iqp_filter term Voice from learn-vlan-1p-priority 5
set firewall family vpls filter vpls_iqp_filter term Voice then policer Voice-IN
set firewall family vpls filter vpls_iqp_filter term Voice then count Voice
set firewall family vpls filter vpls_iqp_filter term Voice then accept
set firewall family vpls filter vpls_iqp_filter term BestEffortSum then count BestEffortSum
set firewall family vpls filter vpls_iqp_filter term BestEffortSum then next term
set firewall family vpls filter vpls_iqp_filter term BestEffort then policer BestEffort-IN
set firewall family vpls filter vpls_iqp_filter term BestEffort then count BestEffort
set firewall family vpls filter vpls_iqp_filter term BestEffort then accept
set firewall family vpls filter vpls_iqp_filter policer pol-vpls if-exceeding bandwidth-limit
400m
set firewall family vpls filter vpls_iqp_filter policer pol-vpls if-exceeding burst-size-limit
40m
set firewall family vpls filter vpls_iqp_filter policer pol-vpls then discard
set firewall family vpls filter vpls_iqp_filter policer Voice-IN if-exceeding bandwidth-limit
100m
set firewall family vpls filter vpls_iqp_filter policer Voice-IN if-exceeding burst-size-limit
10m
set firewall family vpls filter vpls_iqp_filter policer Voice-IN then loss-priority high
set firewall family vpls filter vpls_iqp_filter policer BestEffort-IN if-exceeding bandwidth-
limit 350m
set firewall family vpls filter vpls_iqp_filter policer BestEffort-IN if-exceeding burst-size-
limit 30m
set firewall family vpls filter vpls_iqp_filter policer BestEffort-IN then loss-priority high
set interfaces xe-0/0/0 unit 0 family vpls iq-policing-filter vpls_iqp_filter
```

Configuring the Firewall Filter and Applying It to an Interface as an Input Queuing Policing Filter

Step-by-Step Procedure

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

To configure the firewall filter, `vpls_iqp_filter`, and apply it to logical interface xe-0/0/0 unit 0:

1. Create a firewall filter named `vpls_iqp_filter`.

   ```
   [edit firewall family vpls filter vpls_iqp_filter]
   user@router# set interface-specific
   ```

```
user@router# set term VoiceSum from learn-vlan-1p-priority 5
user@router# set term VoiceSum then count VoiceSum
user@router# set term VoiceSum then forwarding-class Voice
user@router# set term VoiceSum then next term
user@router# set term Voice from learn-vlan-1p-priority 5
user@router# set term Voice then policer Voice-IN
user@router# set term Voice then count Voice
user@router# set term Voice then accept
user@router# set term BestEffortSum then count BestEffortSum
user@router# set term BestEffortSum then next term
user@router# set term BestEffort then policer BestEffort-IN
user@router# set term BestEffort then count BestEffort
user@router# set term BestEffort then accept
user@router# set policer pol-vpls if-exceeding bandwidth-limit 400m
user@router# set policer pol-vpls if-exceeding burst-size-limit 40m
user@router# set policer pol-vpls then discard
user@router# set policer Voice-IN if-exceeding bandwidth-limit 100m
user@router# set policer Voice-IN if-exceeding burst-size-limit 10m
user@router# set policer Voice-IN then loss-priority high
user@router# set policer BestEffort-IN if-exceeding bandwidth-limit 350m
user@router# set policer BestEffort-IN if-exceeding burst-size-limit 30m
user@router# set policer BestEffort-IN then loss-priority high
```

2. Apply the firewall filter to the logical interface.

```
[edit interfaces xe-0/0/0]
user@router# set unit 0 family vpls iq-policing-filter vpls_iqp_filter
```

Results

From configuration mode, confirm your configuration by entering the `show firewall` and the `show interfaces xe-0/0/0.0` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@router# show firewall family vpls filter vpls_iqp_filter
```

```
interface-specific;
term VoiceSum {
    from {
        learn-vlan-1p-priority 5;
    }
    then {
        count VoiceSum;
        forwarding-class Voice;
        next term;
    }
}
term Voice {
    from {
        learn-vlan-1p-priority 5;
    }
    then {
        policer Voice-IN;
        count Voice;
        accept;
    }
}
term BestEffortSum {
    then {
        count BestEffortSum;
        next term;
    }
}
term BestEffort {
    then {
        policer BestEffort-IN;
        count BestEffort;
        accept;
    }
}
policer pol_vpls {
    if-exceeding {
        bandwidth-limit 400m;
        burst-size-limit 40m;
    }
    then discard;
}
policer Voice-IN {
    if-exceeding {
        bandwidth-limit 100m;
        burst-size-limit 10m;
    }
    then loss-priority high;
}
policer BestEffort-IN {
    if-exceeding {
        bandwidth-limit 350m;
        burst-size-limit 30m;
    }
    then loss-priority high;
}
```

```
user@router# show interfaces xe-0/0/0 unit 0
family vpls {
    iq-policing-filter vpls_iqp_filter;
}
```

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

```
user@router# commit
```

SEE ALSO

iq-policing-filter

RELATED DOCUMENTATION

iq-policing-filter

Ingress Rate Limiting on MPCs

On MPCs that support ingress queueing, you can perform rate limiting on incoming packets based on the forwarding class and packet loss priority (PLP) defined for each packet at ingress. You can define the ingress forwarding class either through behavior aggregate (BA) classification or through multifield (MF) ingress-queuing-filter classification.

A packed entering an interface that has ingress queuing and ingress rate-limiting enabled has the following path through the device:

```
Packet in -> BA classification -> MF classification -> Ingress rate-limiting -> Ingress queuing -
> Loopback through the interface -> BA classification -> MF classification -> Firewall filters -
> Routing -> Egress pipeline
```

As the packet enters the interface, BA and MF classification are used to determine the forwarding class and PLP for the packet. Ingress rate-limiting is applied to the packet based on the forwarding class and PLP just determined. If no BA classifier is defined for the packet, the default BA classifier is used. Use of MF classification is optional and overrides any BA classification, including default BA classification.

Ingress rate limiting can be applied to physical and logical interfaces as well as interface sets.

To configure ingress rate-limiting:

1. Configure the `traffic-manager` statement with `ingress-and-egress` mode:

   ```
   [edit chassis fpc slot-number pic pic-number]
   user@router# set traffic-manager mode ingress-and-egress;
   ```

2. Configure a rate-limited scheduler. For example:

   ```
   [edit class-of-service]
   user@router# set schedulers sched_RL transmit-rate percent 70;
   user@router# set schedulers sched_RL transmit-rate rate-limit;
   ```

3. Apply the scheduler to a scheduler map. For example:

   ```
   [edit class-of-service]
   user@router# set scheduler-maps SMap_RL forwarding-class expedited-forwarding scheduler
   sched_RL;
   ```

4. Apply the scheduler map as an `input-scheduler-map` to a physical or logical interface or interface set. For example:

   ```
   [edit class-of-service]
   user@router# set interfaces interface-set my-set input-scheduler-map SMap_RL;
   ```

```
user@router# set interfaces ge-8/0/2 input-scheduler-map SMap_RL;
user@router# set interfaces ge-8/0/3 unit 0 input-scheduler-map SMap_RL;
```

NOTE: Alternatively, apply the scheduler map to a traffic control profile, then apply the traffic control profile as an `input-traffic-control-profile` to a physical or logical interface or interface set.

RELATED DOCUMENTATION

Multifield Classifier for Ingress Queuing on MX Series Routers with MPC

Configuring Ingress Hierarchical CoS | **459**

Rate Shaping on MIC and MPC Interfaces

IN THIS SECTION

This topic covers the following information:

Granularity of Rate Shaping on MIC and MPC Interfaces

Interfaces hosted on MIC and MPC line cards have a certain granularity in the application of configured shaping rates. In other words, the observed hardware value might not exactly match the user-configured value. Nevertheless, the derived values are as close to the configured values as allowed.

Table 98 on page 969 lists the shaping granularity for each MPC port type. The derived shaping rate granularity ranges from 250 Kbps for coarse-grained queuing on the basic hardware up to 1.5 Kbps for fine-grained queuing on the enhanced queuing hardware.

Table 98: Shaping Rate Granularity for MPC Ports

MIC or MPC Port in an MX Series Router		Shaping Rate Granularity	
Line Card Type	Port Speed	Port Level, Queue Level	Logical Interface Level, Interface Set Level
Non-Queuing MPC	1 Gbps / 10 Gbps	250 Kbps	n/a
Queuing MIC or MPC	1 Gbps	2.4 Kbps	9.6 Kbps
	10 Gbps	9.6 Kbps	38.4 Kbps
Enhanced Queuing MPCs	1 Gbps	1.5 Kbps	6 Kbps
	10 Gbps	6 Kbps	24 Kbps

NOTE: The shaping rate granularity for MX Series routers with the MPC3E and MPC4E is approximately 293-300 Kbps. For routers with other MPCs (Trio-based FPCs), the shaping rate granularity is 250 Kbps. The predefined shaping rates for these MPCs are the next multiple of these shaping rate granularity values. The expected deviation from the predefined shaping rates is 5 to 10 percent.

Accounting for Layer 1 and Layer 2 Overhead in Egress Rate-Shaping Statistics

In calculating egress rate-shaping statistics for shaped-session packets on the egress side of MIC and MPC interfaces, the system adds 20 bytes per packet by default.

To configure an explicit overhead value to use for calculating egress rate-shaping statistics, include the `egress-shaping-overhead` statement at the `[edit chassis fpc` *`slot-number`* `pic` *`pic-number`* `traffic-manager]` hierarchy level. You can specify an offset value from –63 bytes through 192 bytes per egress packet.

RELATED DOCUMENTATION

CoS Features and Limitations on MIC and MPC Interfaces | 938

Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 992

Per-Priority Shaping on MIC and MPC Interfaces Overview

Per-priority shaping enables you to configure a separate shaping rate for each of the five priority levels supported by MIC and MPC interfaces. The main use of per-priority shaping rates is to ensure that higher priority services such as voice and video do not starve lower priority services such as data.

There are five scheduler priorities:

- Guaranteed high (GH)
- Guaranteed medium (GM)
- Guaranteed low (GL)
- Excess high (EH)
- Excess low (EL)

The five scheduler priorities support a shaping rate for each priority:

- Shaping rate priority high (GH)
- Shaping rate priority medium (GM)
- Shaping rate priority low (GL)
- Shaping rate excess high (EH)
- Shaping rate excess low (EL)

On MPC7E (MPC7E-MRATE and MPC7E-10G), MPC8E (MX2K-MPC8E), MPC9E(MX2K-MPC9E), and PTX series, when you enable the enhanced priority mode feature, additional scheduler priorities and shaping rates are supported. For more information on the enhanced priority mode feature, see *enhanced-priority-mode*.

The additional scheduler priorities supported when you enable the enhanced priority mode feature:

- Shaping rate priority strict high (GHL)
- Shaping rate priority medium low (GML)
- Shaping rate excess medium high (EMH)
- Shaping rate excess medium low (EML)

When you enable the enhanced priority mode feature, the queue priorities are mapped to the priorities of the MPCs :

Table 99: Shaping Priority and Default Excess Shaping Priority

Configured Priority	Priority Supported on the MPC	Default Excess Priority
Strict-High	GH	EH
High	GHL	EH
Medium-High	GM	EL
Medium-Low	GML	EL
Low	GL	EM

If each service is represented by a forwarding class queued at a separate priority, then assigning a per-priority shaping rate to higher priority services accomplishes the goal of preventing the starvation of lower priority services.

To configure per-priority shaping rates, include the `shaping-rate-excess-high` *rate* `<burst-size` *burst*`>`, `shaping-rate-excess-low` *rate* `<burst-size` *burst*`>`, `shaping-rate-priority-high` *rate* `<burst-size` *burst*`>`, `shaping-rate-priority-low` *rate* `<burst-size` *burst*`>`, or `shaping-rate-priority-medium` *rate* `<burst-size` *burst*`>` at the `[edit class-of-service traffic-control-profiles` *tcp-name*`]` hierarchy level and apply the traffic control profile at the `[edit interfaces]` hierarchy level. You can specify the rate in absolute values, or by using `k` (kilo-), `m` (mega-) or `g` (giga-) units.

You can include one or more of the per-priority shaping statements in a traffic control profile:

```
[edit class-of-service]
traffic-control-profiles {
    tcp-ge-port {
        shaping-rate-excess-high

         rate <burst-size bytes>;
        shaping-rate-excess-low

         rate <burst-size bytes>;
        shaping-rate-priority-high
```

```
            rate <burst-size bytes>;
          shaping-rate-priority-low

            rate <burst-size bytes>;
          shaping-rate-priority-medium

            rate <burst-size bytes>;
     }
 }
```

NOTE: To use per-priority shaping on a physical interface on the MX104 router, you must enable hierarchical scheduling on the interface with the **set hierarchical-scheduler** statement at the [`edit interface` *`interface-name`*] hierarchy level.

BEST PRACTICE: When planning your implementation, consider the following behavior. You can configure independent burst-size values for each rate, but the system uses the maximum burst-size value configured in each rate family. For example, the system uses the highest configured value for the guaranteed rates (GH and GM) or the highest value of the excess rates (EH and EM).

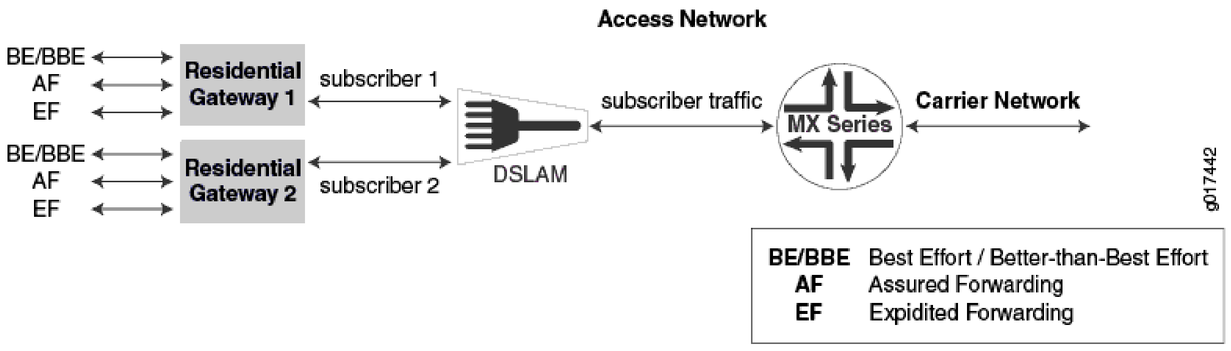
There are several important points about per-priority shaping rates:

- Per-priority shaping rates are only supported on MIC and MPC interfaces (with the exception of the 10-Gigabit Ethernet MPC with SFP+).
- Per-priority shaping is only available for level 1 and level 2 scheduler nodes. (For more information on hierarchical schedulers, see "Configuring Hierarchical Schedulers for CoS" on page 446.)
- Per-priority shaping rates are supported when level 1 or level 2 scheduler nodes have static or dynamic interfaces above them.
- Per-priority shaping rates are supported on aggregated Ethernet (AE) interfaces.
- Per-priority shaping rates are only supported in traffic control profiles.

Per-priority shaping rates can be helpful when the MX Series 5G Universal Routing Platform is in a position between subscriber traffic on an access network and the carrier network, playing the role of a broadband services router. In that case, the MX Series router provides quality-of-service parameters on the subscriber access network so that each subscriber receives a minimum bandwidth (determined by the guaranteed rate) and a maximum bandwidth (determined by the shaping rate). This allows the devices closer to the carrier network to operate more efficiently and more simply and reduces operational network expenses because it allows more centralized network management.

One architecture for using per-priority shaping on the MX Series router is shown in Figure 66 on page 973. In the figure, subscribers use residential gateways with various traffic classes to support voice, video, and data services. The MX Series router sends this traffic from the carrier network to the digital subscriber line access multiplexer (DSLAM) and from the DSLAM on to the residential gateway devices.

Figure 66: Architecture for MIC and MPC Interface Per-Priority Shaping

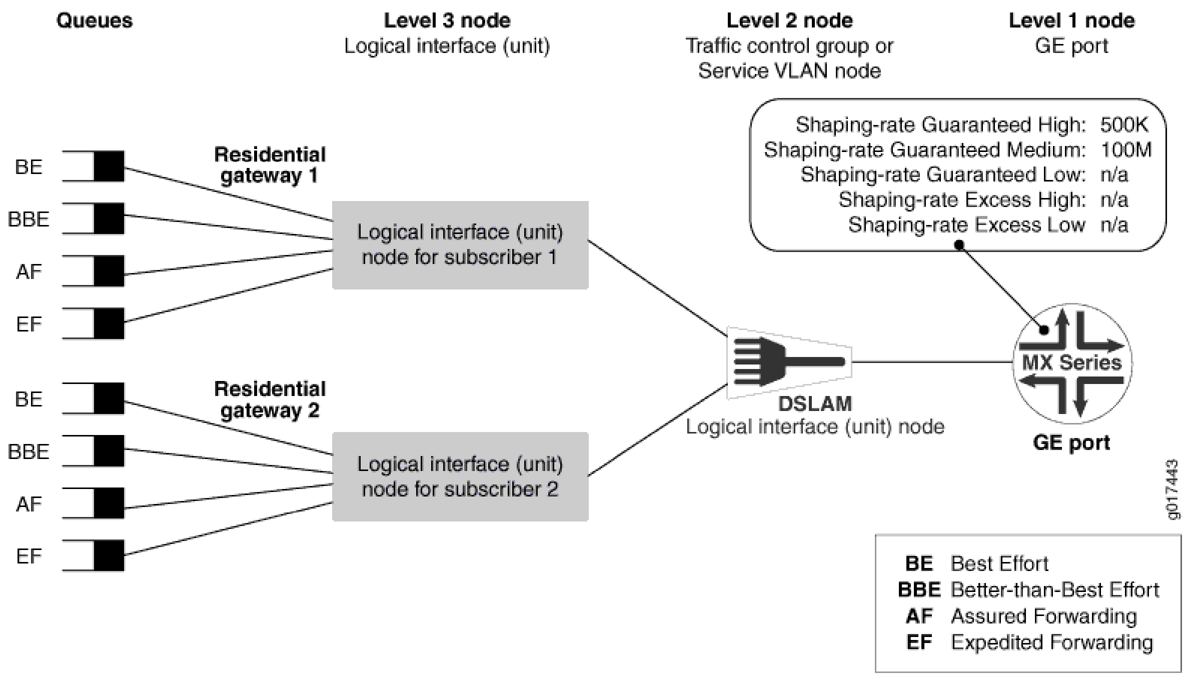


One way that the MX Series router can provide service classes for this physical network topology is shown in Figure 67 on page 974. In the figure, services such as voice and video are placed in separate forwarding classes and the services at different priority levels. For example:

- All expedited-forwarding queues are voice services at a priority level of guaranteed high.
- All assured-forwarding queues are video services at a priority level of guaranteed medium.
- All better-than-best-effort queues are services at a priority level of excess high.
- All best-effort queues are services at a priority level of excess low.

NOTE: This list covers only one possible configuration. Others are possible and reasonable, depending on the service provider’s goals. For example, best-effort and better-than-best-effort traffic can have the same priority level, with the better-than-best-effort forwarding class having a higher scheduler weight than the best-effort forwarding class. For more information on forwarding classes, see "Configuring a Custom Forwarding Class for Each Queue" on page 291.

Figure 67: Scheduling Hierarchy for Per-Priority Shaping


Queues
Level 3 node
Logical interface (unit)
Level 2 node
Traffic control group or Service VLAN node
Level 1 node
GE port
Shaping-rate Guaranteed High: 500K
Shaping-rate Guaranteed Medium: 100M
Shaping-rate Guaranteed Low: n/a
Shaping-rate Excess High: n/a
Shaping-rate Excess Low n/a
BE
BBE
AF
EF
Residential gateway 1
Logical interface (unit) node for subscriber 1
BE
BBE
AF
EF
Residential gateway 2
Logical interface (unit) node for subscriber 2
DSLAM
Logical interface (unit) node
MX Series
GE port
g017443
BE Best Effort
BBE Better-than-Best Effort
AF Assured Forwarding
EF Expedited Forwarding

Aggregated voice traffic in this topology is shaped by applying a high-priority shaper to the port. Aggregated video traffic is shaped in the same way by applying a medium-priority shaper to the port. As long as the sum of the high- and medium-priority shapers is less than the port speed, some bandwidth is reserved for best-effort and better-than-best-effort traffic. So assured-forwarding and expedited-forwarding voice and video cannot starve best-effort and better-than-best-effort data services. One possible set of values for high-priority (guaranteed high) and medium-priority (guaranteed medium) traffic is shown in Figure 67 on page 974.

BEST PRACTICE: We recommend that you do not shape delay-sensitive traffic such as voice traffic because it adds delay (latency). Service providers often use connection admission control (CAC) techniques to limit aggregated voice traffic. However, establishing a shaping rate for other traffic guards against CAC failures and can be useful in pacing extreme traffic bursts.

Per-priority shaping statements:

```
[edit class-of-service]
traffic-control-profile {
    tcp-for-ge-port {
        shaping-rate-priority-high 500k;
        shaping-rate-priority-medium 100m;
    }
}
```

Apply (attach) the traffic control profile to the physical interface (port) at the `[edit class-of-services interfaces]` hierarchy level:

```
[edit class-of-service]
interfaces {
    ge-1/0/0 {
        output-traffic-control-profile tcp-for-ge-port;
    }
}
```

Traffic control profiles with per-priority shaping rates can only be attached to interfaces that support per-priority shaping.

You can apply per-priority shaping to levels other than the level 1 physical interface (port) of the scheduler hierarchy. Per-priority shaping can also be applied at level 2, the interface set level, which would typically represent the digital subscriber link access multiplexer (DSLAM). At this level you could use per-priority shaping to limit to total amount of video traffic reaching a DSLAM, for example.

You apply (attach) the traffic control profile to an interface set at the `[edit class-of-services interfaces]` hierarchy level:

```
[edit class-of-service]
interfaces {
    interface-set svlan-1 {
        output-traffic-control-profile tcp-for-ge-port;
    }
}
```

NOTE: Although you can configure both input and output traffic control profiles, only output traffic control profiles are supported for per-priority shaping.

You can configure per-priority shaping for the traffic remaining with the `output-traffic-control-profile-remaining` statement on a physical port (a level 2 node) but not for an interface set (a level 3 node).

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
16.1R1	On MPC7E (MPC7E-MRATE and MPC7E-10G), MPC8E (MX2K-MPC8E), MPC9E(MX2K-MPC9E), and PTX series, when you enable the enhanced priority mode feature, additional scheduler priorities and shaping rates are supported.

RELATED DOCUMENTATION

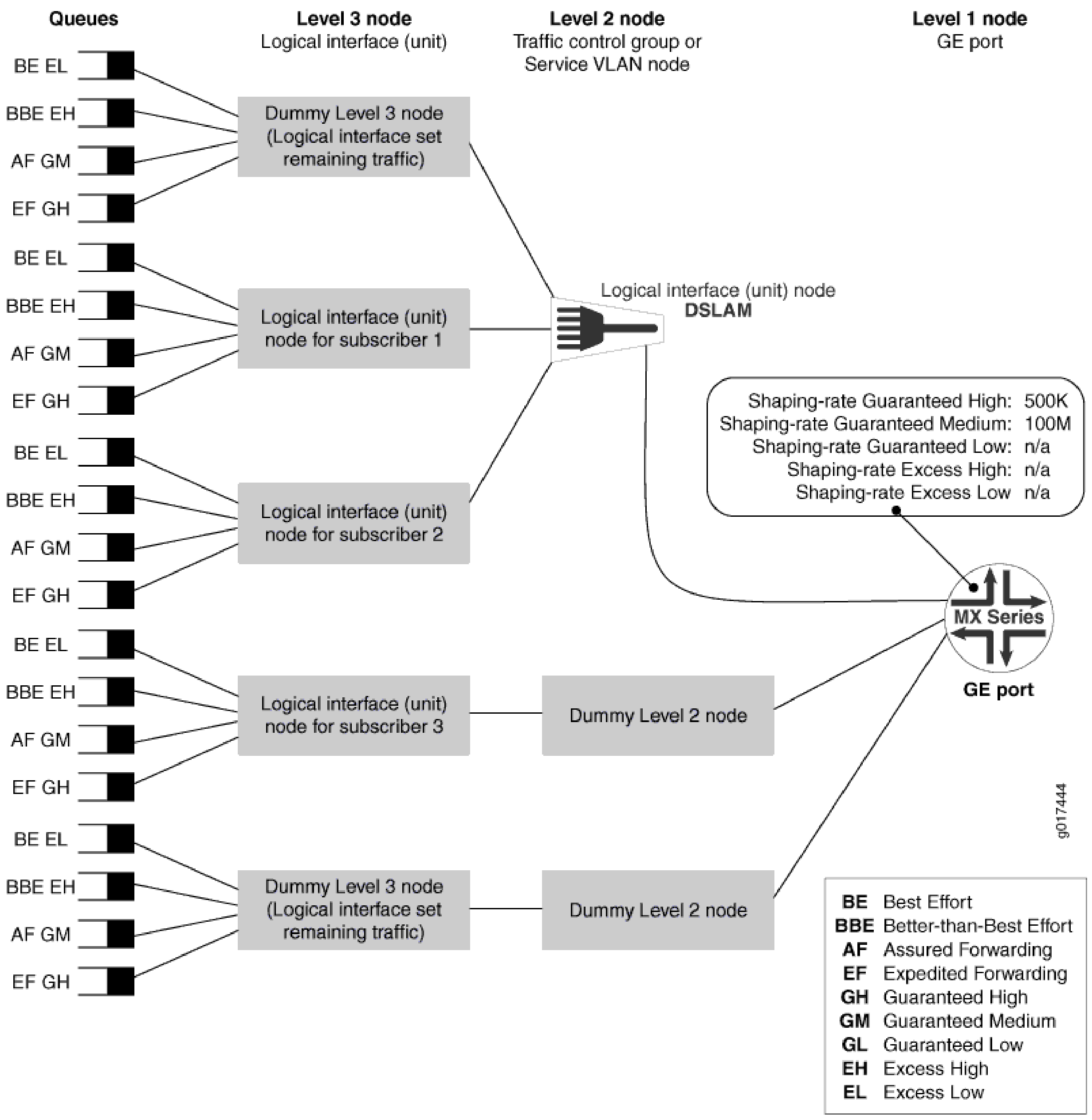
Excess Bandwidth Distribution on MIC and MPC Interfaces Overview

enhanced-priority-mode

Example: Configuring Per-Priority Shaping on MIC and MPC Interfaces

In practice, per-priority shaping is used with other traffic control profiles to control traffic as a whole. Consider the traffic control profile applied to the physical interface (port), as shown in Figure 68 on page 977.

Figure 68: Example of MIC and MPC Interface Scheduling Hierarchy


Queues
Level 3 node
Logical interface (unit)
Level 2 node
Traffic control group or Service VLAN node
Level 1 node
GE port
BE EL
BBE EH
AF GM
EF GH
Dummy Level 3 node (Logical interface set remaining traffic)
Logical interface (unit) node for subscriber 1
Logical interface (unit) node for subscriber 2
Logical interface (unit) node for subscriber 3
Dummy Level 2 node
Logical interface (unit) node
DSLAM
Shaping-rate Guaranteed High: 500K
Shaping-rate Guaranteed Medium: 100M
Shaping-rate Guaranteed Low: n/a
Shaping-rate Excess High: n/a
Shaping-rate Excess Low n/a
MX Series
GE port
g017444
BE Best Effort
BBE Better-than-Best Effort
AF Assured Forwarding
EF Expedited Forwarding
GH Guaranteed High
GM Guaranteed Medium
GL Guaranteed Low
EH Excess High
EL Excess Low

This example is more complex than those used before. In addition to a pair of subscribers in an interface set (DSLAM), the figure now adds the following:

- A dummy level 3 scheduler node (`interface-set-remaining-traffic`) that provides scheduling for interface set members that do not have explicit class-of-service parameters configured.
- A subscriber (Subscriber 3) that is not a member of an interface set. A dummy level 2 node connects Subscriber 3’s level 3 node to level 1, making it appear to be at level 2.

- A dummy level 3 scheduler node (`port-remaining-traffic`) in order to provide queues for traffic that does not have explicit class-of-service parameters configured.
- A dummy level 2 scheduler node to connect level 1 and level 3 scheduler nodes. This dummy level 2 scheduler node is internal only.

This example uses a gigabit Ethernet interface with five logical interface units, each one representing one of the level 3 nodes in Figure 68 on page 977.

From the top of the figure to the bottom, the level 3 nodes are:

- Unit 3 is scheduled as a “dummy” level 3 node because unit 3 is a member of an interface set (`ifset-1`) but there is no explicit CoS configuration.
- Unit 1 is scheduled as a logical interface node for subscriber 1 because unit 1 is a member of an interface set (`ifset-1`) and has an explicit CoS configuration under the `[edit class-of-service interfaces]` hierarchy.
- Unit 2 is scheduled as a logical interface node for subscriber 2 because unit 2 is a member of an interface set (`ifset-1`) and has an explicit CoS configuration under the `[edit class-of-service interfaces]` hierarchy.
- Unit 4 is scheduled as a logical interface node for subscriber 3 because unit 4 is not a member of an interface set but has an explicit CoS configuration under the `[edit class-of-service interfaces]` hierarchy level.
- Unit 5 is scheduled by another “dummy” level 3 node, this one for remaining traffic at the port level, because unit 5 is not a member of an interface set and has no explicit CoS configuration.

In this example, per-priority shaping is applied at the physical port level. The example uses three priorities, but other parameters are possible. The example does not use shaping rates, transmit rates, excess priorities, or other options for reasons of simplicity. The example uses five forwarding classes and leaves out a network control forwarding class that would typically be included in real configurations.

The example configuration is presented in several parts:

- Interfaces configuration
- Class-of-service forwarding classes and traffic control profiles configuration
- Class-of-service interfaces configuration
- Class-of-service schedulers and scheduler map configuration

Interfaces configuration:

```
[edit]
interfaces {
    # A three member interface-set.
    interface-set ifset-1 {
        interface ge-1/1/0 {
            unit 1;
            unit 2;
            unit 3;
        }
    }
    # A ge port configured for "hierarchical-scheduling" and
    # vlans.  5 vlans are configured for the 5 level-3 scheduler
    # nodes
    #
    ge-1/1/0 {
        hierarchical-scheduler;
        vlan-tagging;
        unit 1 {
            vlan-id 1;
        }
        unit 2 {
            vlan-id 2;
        }
        unit 3 {
            vlan-id 3;
        }
        unit 4 {
            vlan-id 4;
        }
        unit 5 {
            vlan-id 5;
        }
    }
}
```

Class-of-service forwarding classes and traffic control profiles configuration:

```
[edit class-of-service]
forwarding-classes {
    queue 0 BE priority low;
    queue 1 BBE priority low;
    queue 2 AF priority low;
    queue 3 EF priority high;
}
traffic-control-profiles {
    tcp-if-portd {
        shaping-rate-priority-high 500k;
        shaping-rate-priority-medium 100m;
    }
    tcp-if-port-rem {
        scheduler-map smap-1;
    }
    tcp-ifset-rem {
        scheduler-map smap-1;
    }
    tcp-if-unit {
        scheduler-map smap-1;
        shaping-rate 10m;
    }
}
```

Class-of-service interfaces configuration:

```
[edit class-of-service]
interfaces {
    interface-set ifset-1 {
        output-traffic-control-profile-remaining tcp-ifset-rem;
    }
    ge-1/1/0 {
        output-traffic-control-profile tcp-if-port;
        output-traffic-control-profile-remaining tcp-if-port-rem;
        unit 1 {
            output-traffic-control-profile tcp-if-unit;
        }
        unit 2 {
            output-traffic-control-profile tcp-if-unit;
        }

        # Unit 3 present in the interface config and interface-set
        # config, but is absent in this CoS config so that we can
        # show traffic that uses the interface-set
        # remaining-traffic path.

        unit 4 {
            output-traffic-control-profile tcp-if-unit;
        }

        # Unit 5 is present in the interface config, but is absent
        # in this CoS config so that we can show traffic that
        # uses the if-port remaining-traffic path.
    }
}
```

Class-of-service schedulers and scheduler map configuration:

```
[edit class-of-service]
scheduler-maps {
    smap-1 {
        forwarding-class BE scheduler sched-be;
        forwarding-class BBE scheduler sched-bbe;
        forwarding-class AF scheduler sched-af;
    forwarding-class EF scheduler sched-ef;
}
schedulers {
    sched-be {
        priority low;
    }
    sched-bbe {
        priority low;
    }
    sched-af {
        priority medium-high;
    }
    sched-ef {
        priority high;
    }
}
```

You can configure both a shaping rate and a per-priority shaping rate. In this case, the legacy `shaping-rate` statement specifies the maximum rate for all traffic scheduled through the scheduler. Therefore, the per-priority shaping rates must be less than or equal to the overall shaping rate. So if there is a `shaping-rate 400m` statement configured in a traffic control profile, you cannot configure a higher value for a per-priority shaping rate (such as `shaping-rate-priority-high 500m`). However, the sum of the per-priority shaping rates can exceed the overall shaping rate: for `shaping-rate 400m` you can configure both `shaping-rate-priority-high 300m` and `shaping-rate-priority-low 200m` statements.

Generally, you cannot configure a shaping rate that is smaller than the guaranteed rate (which is why it is guaranteed). However, no such restriction is placed on per-priority shaping rates unless all shaping rates are for priority high or low or medium traffic.

This configuration is allowed (per-priority rates smaller than guaranteed rate):

```
[edit class-of-service]
traffic-control-profile {
    tcp-for-ge-port {
        guaranteed-rate 500m;
        shaping-rate-priority-high 400m;
        shaping-rate-priority-medium 300m;
        shaping-rate-excess-high 100m;
    }
}
```

However, this configuration generates an error (no excess per-priority rate, so the node can never achieve its guaranteed rate):

```
[edit class-of-service]
traffic-control-profile {
    tcp-for-ge-port {
        guaranteed-rate 301m;
        shaping-rate-priority-high 100m;
        shaping-rate-priority-medium 100m;
        shaping-rate-priority-low 100m;
    }
}
```

You verify configuration of per-priority shaping with the `show class-of-service traffic-control-profile` command. This example shows shaping rates established for the high and medium priorities for a traffic control profile named `tcp-ge-port`.

```
user@host# **show class-of-service traffic-control-profile**
Traffic control profile: tcp-ae, Index: 22093
   Shaping rate: 3000000000
   Scheduler map: <default>
```

```
Traffic control profile: tcp-ge-port, Index: 22093
   Shaping rate priority high: 1000000000
   Shaping rate priority medium: 9000000000
   Scheduler map: <default>
```

There are no restrictions on or interactions between per-priority shaping rates and the excess rate. An excess rate (a weight) is specified as a percentage or proportion of excess bandwidth.

Table 100 on page 983 shows where traffic control profiles containing per-priority shaping rates can be attached for both per-unit schedulers and hierarchical schedulers.

Table 100: Applying Traffic Control Profiles

Type of Traffic Control Profile	Per-unit Allowed?	Hierarchical Allowed?
Port level `output-traffic-control-profile` with per-priority shaping	Yes	Yes
Port level `output-traffic-control-profile-remaining` with per-priority shaping	No	Yes
Port level `output-traffic-control-profile` and `output-traffic-control-profile-remaining` with per-priority shaping	No	Yes
Port level `input-traffic-control-profile` with per-priority shaping	No	No
Port level `input-traffic-control-profile-remaining` with per-priority shaping	No	No
Interface set `output-traffic-control-profile` with per-priority shaping	No	Yes
Interface set `output-traffic-control-profile-remaining` with per-priority shaping	No	No
Interface set `input-traffic-control-profile` with per-priority shaping	No	No
Interface set `input-traffic-control-profile-remaining` with per-priority shaping	No	No
Logical interface level `output-traffic-control-profile` with per-priority shaping	No	No

Table 100: Applying Traffic Control Profiles *(Continued)*

Type of Traffic Control Profile	Per-unit Allowed?	Hierarchical Allowed?
Logical interface level `input-traffic-control-profile` with per-priority shaping	No	No

RELATED DOCUMENTATION

Per-Priority Shaping on MIC and MPC Interfaces Overview | 970

Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates

The overhead accounting feature enables you to account for downstream traffic that has different encapsulations or downstream traffic from cell-based equipment, such as ATM switches.

You can configure the overhead accounting feature to shape downstream traffic based on frames or cell shaping mode.

You can also account for the different byte sizes per encapsulation by configuring a byte adjustment value for the shaping mode.

To configure the shaping mode and byte adjustment value for static CoS configurations:

1. Specify the shaping mode.

   Frame shaping mode is enabled by default.

   ```
   [edit class-of-service traffic-control-profiles profile-name]
   user@host# set overhead-accounting (frame-mode | cell-mode)
   ```

2. (Optional) Specify a byte adjustment value.

   ```
   [edit class-of-service traffic-control-profiles profile-name
   user@host# set overhead-accounting bytes byte-value]
   ```

BEST PRACTICE: We recommend that you specify a byte adjustment value that represents the difference between the customer premise equipment (CPE) protocol overhead and the B-RAS protocol overhead.

The available range is –120 through 124 bytes. The system rounds up the byte adjustment value to the nearest multiple of 4. For example, a value of 6 is rounded to 8, and a value of –10 is rounded to –8.

RELATED DOCUMENTATION

Bandwidth Management for Downstream Traffic in Edge Networks Overview

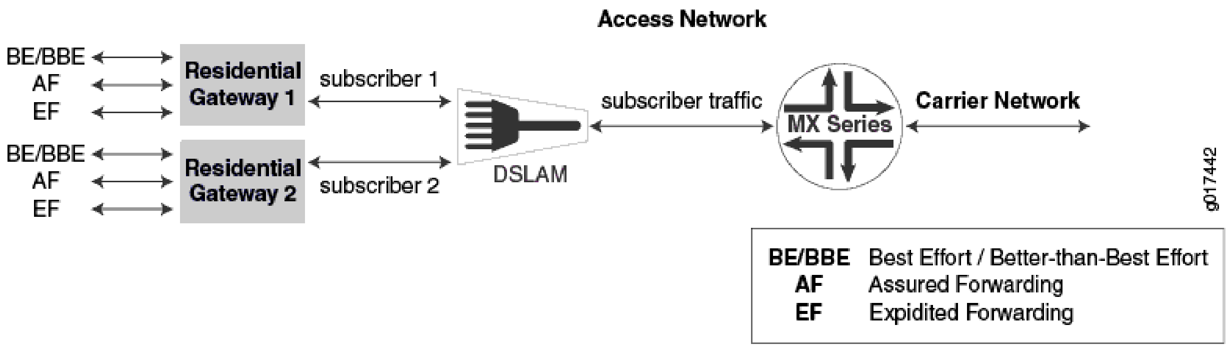
Example: Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates

IN THIS SECTION

This topic describes two scenarios for which you can configure static shaping parameters to account for packet overhead in a downstream network.

Figure 69 on page 986 shows the sample network that the examples reference.

Figure 69: Sample Network Topology for Downstream Traffic



Managing Traffic with Different Encapsulations

In this example, the MX Series router shown in Figure 69 on page 986 sends stacked VLAN frames to the DSLAM, and the DSLAM sends single-tagged VLAN frames to the residential gateway.

To accurately shape traffic at the residential gateway, the MX Series router must account for the different frame sizes. The difference between the stacked VLAN (S-VLAN) frames sent by the router and the single-tagged VLAN frames received at the residential gateway is a 4-byte VLAN tag. The residential gateway receives frames that are 4 bytes less.

To account for the different frame sizes, the network administrator configures the frame shaping mode with –4 byte adjustment:

1. The network administrator configure the traffic shaping parameters and attaches them to the interface.

   Enabling the overhead accounting feature affects the resulting shaping rate, guaranteed rate, and excess rate parameters, if they are configured.

```
[edit]
class-of-service {
    traffic-control-profiles {
        tcp-example-overhead-accounting-frame-mode {
            shaping-rate 10m;
            shaping-rate-priority-high 4m;
            guaranteed-rate 2m;
            excess-rate percent 50;
            overhead-accounting frame-mode bytes -4;
            }
        }
        interfaces {
            ge-1/0/0 {
                output-traffic-control-profile tcp-example-overhead-accounting-frame-mode;
            }
        }
    }
}
```

2. The network administrator verifies the adjusted rates.

```
user@host#show class-of-service traffic-control-profile
Traffic control profile: tcp-example-overhead-accounting-frame-mode, Index: 61785
Shaping rate: 10000000
Shaping rate priority high: 4000000
Excess rate 50
Guaranteed rate: 2000000
Overhead accounting mode: Frame Mode
Overhead bytes: –4
```

Managing Downstream Cell-Based Traffic

In this example, the DSLAM and residential gateway shown in Figure 69 on page 986 are connected through an ATM cell-based network. The MX Series router sends Ethernet frames to the DSLAM, and the DSLAM sends ATM cells to the residential gateway.

To accurately shape traffic at the residential gateway, the MX Series router must account for the different physical network characteristics.

To account for the different frame sizes, the network administrator configures the cell shaping mode with –4 byte adjustment:

1. Configure the traffic shaping parameters and attach them to the interface.

   Enabling the overhead accounting feature affects the resulting shaping rate, guaranteed rate, and excess rate parameters, if they are configured.

```
[edit]
class-of-service {
    traffic-control-profiles {
        tcp-example-overhead-accounting-cell-mode {
            shaping-rate 10m;
            shaping-rate-priority-high 4m;
            guaranteed-rate 2m;
```

```
                excess-rate percent 50;
                overhead-accounting cell-mode;
                }
            }
            interfaces {
                ge-1/0/0 {
                    output-traffic-control-profile tcp-example-overhead-accounting-cell-mode;
                }
            }
        }
    }
```

2. Verify the adjusted rates.

```
user@host#show class-of-service traffic-control-profile
Traffic control profile: tcp-example-overhead-accounting-cell-mode, Index: 61785
Shaping rate: 10000000
Shaping rate priority high: 4000000
Excess rate 50
Guaranteed rate: 2000000
Overhead accounting mode: Cell Mode
Overhead bytes: 0
```

To account for ATM segmentation, the MX Series router adjusts all of the rates by 48/53 to account for ATM AAL5 encapsulation. In addition, the router accounts for cell padding, and internally adjusts each frame by 8 bytes to account for the ATM trailer.

RELATED DOCUMENTATION

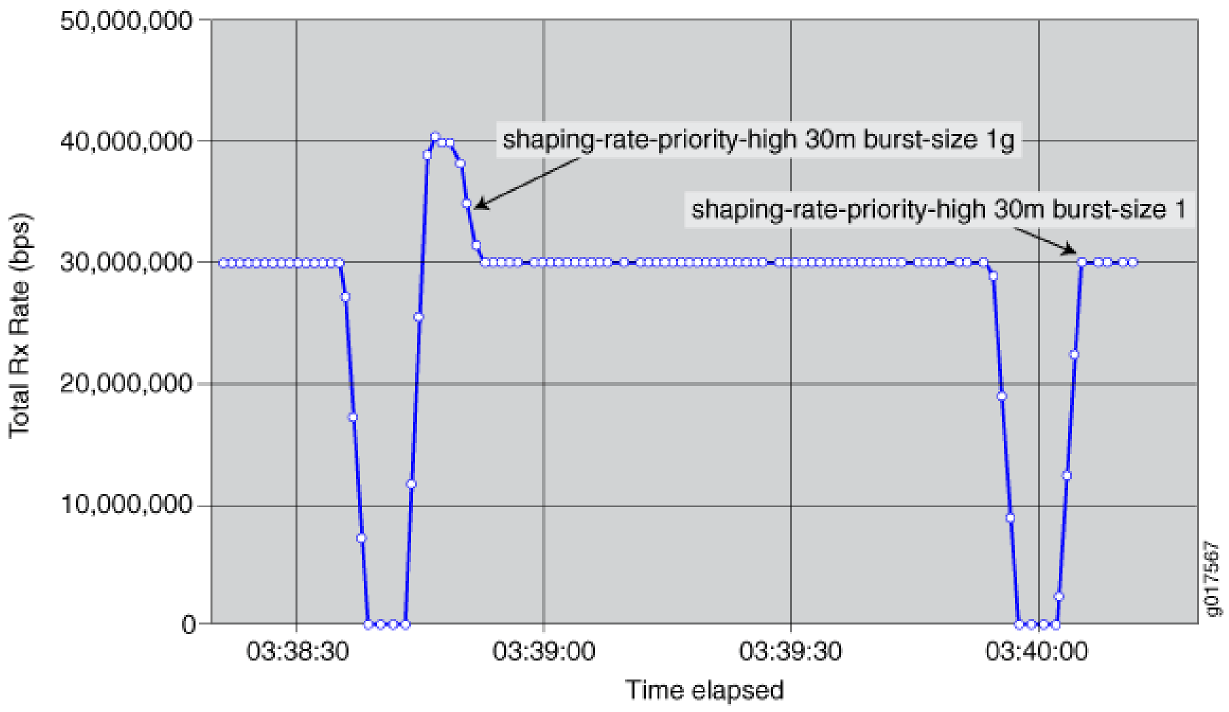
Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates

Traffic Burst Management on MIC and MPC Interfaces Overview

IN THIS SECTION

You can manage the impact of bursts of traffic on your network by configuring a burst-size value with the shaping rate or the guaranteed rate. The value is the maximum bytes of rate credit that can accrue for an idle queue or scheduler node. When a queue or node becomes active, the accrued rate credits enable the queue or node to catch up to the configured rate.

Figure 70: Sample Burst Shaping Rates



In Figure 70 on page 989, the network administrator configures a large burst-size value for the shaping rate, then configures a small burst-size value. The larger burst size is subject to a maximum value. The smaller burst size is subject to a minimum value that enables the system to achieve the configured rates.

In both configurations, the scheduler node can burst beyond its shaping rate for a brief interval. The burst of traffic beyond the shaping rate is more noticeable with the larger burst size than the smaller burst size.

Guidelines for Configuring the Burst Size

Typically, the default burst-size (100 ms) for both scheduler nodes and queues on MIC and MPC interfaces is adequate for most networks. However, if you have intermediate equipment in your network that has very limited buffering and is intolerant of bursts of traffic, you might want to configure a lower value for the burst size.

Use caution when selecting a different burst size for your network. A burst size that is too high can overwhelm downstream networking equipment, causing dropped packets and inefficient network operation. Similarly, a burst size that is too low can prevent the network from achieving your configured rate.

When configuring a burst size, keep the following considerations in mind:

- The system uses an algorithm to determine the actual burst size that is implemented for a node or queue. For example, to reach a shaping rate of 8 Mbps, you must allocate 1MB of rate credits every second. A shaping rate of 8 Mbps with a burst size of 500,000 bytes of rate-credit per seconds enables the system to transmit at most 500,000 bytes, or 4 Mbps. The system cannot implement a burst size that prevents the rate from being achieved.

  For more information, see "How the System Calculates the Burst Size" on page 991.

- There are minimum and maximum burst sizes for each platform, and different nodes and queue types have different scaling factors. For example, the system ensures the burst cannot be set lower than 1 Mbps for a shaping rate of 8 Mbps. To smoothly shape traffic, rate credits are sent much faster than once per second. The interval at which rate credits are sent varies depending on the platform, the type of rate, and the scheduler level.

- When you have configured adjustments for the shaping rate (either by percentage or through an application such as ANCP or Multicast OIF), the system bases the default and minimum burst-size calculations on the adjusted shaping rate.

- When you have configured cell shaping mode to account for ATM cell tax, the system bases the default and minimum burst-size calculations on the post-tax shaping rate.

- The guaranteed rate and shaping rate share the value specified for the burst size. If the guaranteed rate has a burst size specified, that burst size is used for the shaping rate; if the shaping rate has a burst size specified, that bursts size is used for the guaranteed rate. If you have specified a burst size for both rates, the system uses the lesser of the two values.

- The burst size configured for the guaranteed rate cannot exceed the burst-size configured for the shaping rate. Starting in Junos OS Release 15.1, the CLI no longer generates a commit error when the guaranteed-rate burst size is statically configured to be more than the shaping-rate burst size. This behavior changed with the advent of enhanced subscriber management. The system logs an error when the guaranteed-burst rate is higher, whether it is configured statically, dynamically with predefined variables, or by means of a change of authorization request.

- If you have not configured a guaranteed rate, logical interfaces and interface sets receive a default guaranteed rate from the port speed. Queues receive a default guaranteed rate from the parent *logical interface* or interface set.
- Burst-size is not supported with `per-priority-shaping`.

How the System Calculates the Burst Size

When calculating the burst size, the system uses an exponent of a power of two. For example:

```
Shaping-rate in bps * 100 ms / (8 bits/byte * 1000 ms/s) = 1,875,000 bytes
```

The system then rounds this value up. For example, the system uses the following calculation to determine the burst size for a scheduler node with a shaping rate of 150 Mbps:

```
Max (Shaping rate, Guaranteed rate) bps * 100 ms / (8 bits/byte * 1000 ms/s) = 1,875,000 bytes
```

```
Rounded up to the next higher power of two = 2,097,150 (which is 2**21, or 0x200000)
```

The system assigns a single burst size to each of the following rate pairs:

- Shaping rate and guaranteed rate
- Guaranteed high (GH) and guaranteed medium (GM)
- Excess high (EH) and excess low (EL)
- Guaranteed low (GL)

To calculate the burst size for each pair, the system:

- Uses the configured burst-size if only one of the pair is configured.
- Uses the lesser of the two burst sizes if both values are configured.
- Uses the next lower power of two.
- To calculate the minimum burst size, the system uses the greater of the two rates.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
15.1	Starting in Junos OS Release 15.1, the CLI no longer generates a commit error when the guaranteed-rate burst size is statically configured to be more than the shaping-rate burst size.

RELATED DOCUMENTATION

Per-Priority Shaping on MIC and MPC Interfaces Overview

Managing Excess Bandwidth Distribution on Static Interfaces on MICs and MPCs

Understanding Hierarchical Scheduling for MIC and MPC Interfaces

IN THIS SECTION

Scheduler Node Scaling for MIC and MPC Interfaces

In per-unit scheduling, the logical interfaces share a common level 2 node (one per port). In hierarchical-scheduling, each logical interface has its own level 2 node. Thus, scaling is limited by the number of level 2 nodes.

To better control system resources in hierarchical-scheduling mode, you can limit the number of scheduler node levels to two. In this case, all logical interfaces and interface sets with CoS scheduling policy share a single level 2 node. Consequently, the maximum number of logical interfaces with CoS scheduling policies is increased (the interface sets must be at level 3).

To configure scheduler node scaling, include the `hierarchical-scheduler` statement and set the `maximum-hierarchy-levels` option to `2` at the `[edit interfaces xe-`*`fpc/pic/port`*`]` hierarchy level.

```
[edit interfaces]
xe-2/0/0 {
    hierarchical-scheduler {
        maximum-hierarchy-levels 2;
    }
}
```

NOTE: The `maximum-hierarchy-levels` option supports level 3 interface sets but not level 2 interface sets. If you configure level 2 interface sets with the `maximum-hierarchy-levels` option, you generate Packet Forwarding Engine errors.

Hierarchical Scheduling Priority Levels for MIC and MPC Interfaces

The queuing model used by MIC and MPC interfaces supports three priority levels for guaranteed scheduling priority and two lower priority levels for excess scheduling priority. You can configure a queue with one guaranteed priority and one excess priority. For example, you can configure a queue for guaranteed low (GL) as the guaranteed priority and configure excess high (EH) as the excess priority.

You can associate a guaranteed level with only one excess level. You can associate an excess level with any number of guaranteed priority levels, including none.

Interface nodes maintain their guaranteed priority level (for example, guaranteed high, GH) as long as they do not exceed their guaranteed bandwidth. If the queue bandwidth exceeds the guaranteed rate, then the priority drops to the excess priority (for example, excess high, EH). Because excess level priorities are lower than their guaranteed counterparts, the bandwidth guarantees for each of the other levels can be maintained.

Guaranteed Bandwidth and Weight of an Interface Node on MIC and MPC Interfaces

The queuing model used by MIC and MPC interfaces separates the concepts of *guaranteed bandwidth* and *weight* of an interface node, although the two terms are often used interchangeably. The guaranteed bandwidth for an interface node is the bandwidth the node can use, independent of what is happening at the other nodes of the scheduling hierarchy. The weight of an interface node, on the other hand, is a value that determines how *excess bandwidth* is used. The weight of a node comes into play when other nodes at the same hierarchical scheduling level use less than the sum of their guaranteed bandwidths

For some application traffic types (such as constant bit rate voice, where there is little concern about excess bandwidth), the guaranteed bandwidth dominates the node. For other types of application traffic (such as bursty data, where a well-defined bandwidth is not always possible), the concept of weight dominates the node.

Hierarchical Scheduling for MIC and MPC Interfaces in Oversubscribed PIR Mode

In contrast to the Intelligent Queuing Enhanced (IQE) and Intelligent Queuing 2 Enhanced (IQ2E) PICs, the interfaces on MICs and MPCs set the guaranteed rate to zero in oversubscribed peak information rate (PIR) mode for the per-unit scheduler. Also, the configured rate is scaled down to fit the oversubscribed value. For example, if there are two logical interface units with a shaping rate of 1 Gbps each on a 1-Gbps port (which is, therefore, oversubscribed 2 to 1), then the guaranteed rate on each unit is scaled down to 500 Mbps (scaled down by 2).

With hierarchical schedulers in oversubscribed PIR mode, the guaranteed rate for every logical interface unit is set to zero. This means that the queue transmit rates are always oversubscribed.

Because in oversubscribed PIR mode the queue transmit rates are always oversubscribed, the following are true:

- If the queue transmit rate is set as a percentage, then the guaranteed rate of the queue is set to zero; but the excess rate (weight) of the queue is set correctly.
- If the queue transmit rate is set as an absolute value and if the queue has guaranteed high or medium priority, then traffic up to the queue's transmit rate is sent at that priority level. However, for guaranteed low traffic, that traffic is demoted to the excess low region. This means that best-effort traffic well within the queue's transmit rate gets a lower priority than out-of-profile excess high traffic. This differs from the IQE and IQ2E PICs.

RELATED DOCUMENTATION

Per-Unit Scheduling and Hierarchical Scheduling for MPC Interfaces | **999**

CoS Features and Limitations on MIC and MPC Interfaces | **938**

Jitter Reduction in Hierarchical CoS Queues | **1014**

Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | **757**

CoS Three-Level Hierarchical Scheduling on MPLS Pseudowire Subscriber Interfaces

Configuring Ingress Hierarchical CoS on MIC and MPC Interfaces

Use Feature Explorer to confirm platform and release support for specific features.

You can configure ingress CoS parameters, including hierarchical schedulers, on supported interfaces. In general, the supported configuration statements apply to per-unit schedulers or to hierarchical schedulers.

NOTE: Junos OS does not support ingress queuing and ingress hierarchical CoS on AE interfaces. You can, however, configure standard CoS classification and rewrite rules on AE interfaces.

To configure ingress CoS for per-unit schedulers, include the following statements at the [edit class-of-service interfaces *interface-name*] hierarchy level:

```
[edit class-of-service interfaces interface-name]
input-excess-bandwidth-share (proportional value | equal);
input-scheduler-map map-name;
input-shaping-rate rate;
input-traffic-control-profile profile-name;
unit logical-unit-number;
    input-scheduler-map map-name;
    input-shaping-rate (percent percentage | rate);
    input-traffic-control-profile profile-name;
```

To configure ingress CoS for hierarchical schedulers, include the interface-set *interface-set-name* statement at the [edit class-of-service interfaces] hierarchy level:

```
[edit class-of-service interfaces]
interface-set interface-set-name {
    input-traffic-control-profile profile-name;
    input-traffic-control-profile-remaining profile-name;
    interface interface-name {
        input-excess-bandwidth-share (proportional value | equal);
        input-traffic-control-profile profile-name;
        input-traffic-control-profile-remaining profile-name;
        unit logical-unit-number;
    }
}
```

By default, ingress CoS features are disabled interfaces. To enable ingress CoS on an interface, configure the traffic-manager statement with ingress-and-egress mode as shown in the following example:

```
chassis {
    fpc 7 {
        pic 0 {
            traffic-manager {
                mode ingress-and-egress;
            }
        }
    }
}
```

Configured CoS features on the ingress are independent of CoS features on the egress.

NOTE: HQoS MPC cards installed in MX240, MX480, MX960, MX2008, MX2010, and MX2020 routers have a hardware limitation with an ingress queuing CoS "ingress-and-egress" configuration.

Ingress queuing can be enabled for a maximum of 10 ports per MIC Slot, resulting in 20 ports per MPC2E-3D-NG HQoS and MPC3E-3D-NG HQoS line card with 10 ports per MIC slot. In the XM chip there are 16 loopback streams allocated per port group for PG0 and PG1, where PG0 is mapped to MIC slot 0 and PG1 is mapped to MIC slot 1. On enabling ingress queuing on a PIC slot, one loopback stream from the XM chip is allocated per interface from the respective port group. Because there are only 16 loopback streams, out of which 2 are used by default and 4 are used for tunnel interfaces and inline services, 10 streams are left for ingress CoS.

Change History Table

Feature support is determined by the platform and release you are using. Use Feature Explorer to determine if a feature is supported on your platform.

Release	Description
17.4R2	Starting with Junos 17.4R2 on supported devices, you can have precise control over which ports have ingress CoS enabled by configuring `traffic-manager` at the port level.

RELATED DOCUMENTATION

mode (Layer 2 Tunneling Protocol Shaping)

Configuring a CoS Scheduling Policy on Logical Tunnel Interfaces

You can configure a CoS scheduling policy on a logical tunnel interface (LT ifl). Logical tunnel interfaces can be used to terminate a pseudowire into a virtual routing and forwarding (VRF) instance. If an lt device is used to terminate a pseudowire, CoS scheduling policies can be applied on the lt interface to manage traffic entering the pseudowire. You accomplish this by configuring the hierarchical-scheduler attribute on the physical interface.

NOTE: It is important to first commit the hierarchical-scheduler configuration under the logical tunnel physical interface (LT ifd), and subsequently add and commit the class-of-service configuration.

NOTE: The `output-traffic-control` statement applies only to the LT ifl that is part of an L3 VRF instance.

NOTE: On QX chip-based MICs and MPCs only, LT ifls do not support shaping of Layer 2 pseudowires.

The following example shows two pseudowires (pw1 and pw2) over lt-1/0/10. pw1 carries data, voice, and video traffic, and pw2 carries only data and voice traffic. All pseudowire traffic is restricted to 800m bps. The shaping rate for traffic over pw1 is 400m bps and the shaping rate for traffic over pw2 is 400m bps.

```
[edit interfaces]
lt-1/0/10 {
    hierarchical-scheduler;
}

[edit class-of-service schedulers]
data_sch {
    buffer-size remainder;
    priority low;
}
voice_sch {
    transmit-rate 6k;
    priority strict-high;
}
video_sch {
    shaping-rate 1m;
    priority medium-low;
}
[edit class-of-service scheduler-maps]
pw1-smap {
    forwarding-class be scheduler data_sch;
    forwarding-class ef scheduler voice_sch;
    forwarding-class af scheduler video_sch;
```

```
}
pw2-smap {
    forwarding-class be scheduler data_sch;
    forwarding-class ef scheduler voice_sch;
}
[edit class-of-service traffic-control-profiles]
pw1-tcp {
    scheduler-map pw1-smap;
    shaping-rate 400m;
}
pw2-tcp {
    scheduler-map pw2-smap;
    shaping-rate 400m;
}
all-pw-tcp {
    shaping-rate 800m;
}
lt-ifd-remain {
    shaping-rate 10m;
}

[edit class-of-service interfaces]
lt-1/0/10 {
    output-traffic-control-profile all-pw-tcp;
    output-traffic-control-profile-remaining lt-ifd-remain;
unit 1 {
    output-traffic-control-profile pw1-tcp;
}
unit 3 {
    output-traffic-control-profile pw2-tcp;
}
```

RELATED DOCUMENTATION

CoS Scheduling Policy on Logical Tunnel Interfaces Overview | **1026**

Hierarchical CoS for Metro Ethernet Environments | **446**

Configuring Logical Tunnel Interfaces

CoS on Ethernet Pseudowires in Universal Edge Networks Overview | **1025**

Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks | **1026**

Per-Unit Scheduling and Hierarchical Scheduling for MPC Interfaces

IN THIS SECTION

Scheduling Models Supported for MPC Interfaces

Interfaces hosted on Modular Port Concentrator (MPC) line cards in MX Series 5G Universal Routing Platforms support the following models of class-of-service (CoS) scheduling, depending on MPC type:

Limited Scale Per-Unit Scheduling MPCs

Per-unit scheduling features on a limited scale are supported for interfaces hosted on some MPCs that do not have a dedicated queuing chip.

On MPCs that support per-unit scheduling, the following scheduling capabilities are available:

- Four or eight egress queues per unit.
- Delay buffer capacities of 100 ms by default, and up to 200 ms maximum delay.
- Rate shaping of the ports and their queues.
- Guaranteed rate enforced at the queues.

The per-unit scheduling features also support pre-classification of incoming packets to protect high priority packets in the event of congestion. Such features include ingress DSCP rewrite and per-VLAN classification, ingress and egress policing, and rewrites.

Hierarchical Scheduling MPCs

MPCs that provide a dedicated queuing chip support hierarchical scheduling.

Hierarchical scheduling MPCs support all per-unit scheduling functionality plus fine-grained queuing abilities over four or five levels of hierarchical scheduling:

- Hierarchical scheduling with ports, interface sets, and logical interfaces.

- Shaping—Committed Information Rate (CIR) and a peak information rate (PIR)—at all scheduling levels, including queues.
- Three normal- priority levels and two excess- priority levels configurable at all scheduling levels, including queues.
- Per-priority shaping of traffic at Level 1 or Level 2.
- Shaping for unconfigured customer VLANs (C-VLANs) and for service VLANs (S-VLANs).

Scheduler Node Levels for MPC Interfaces

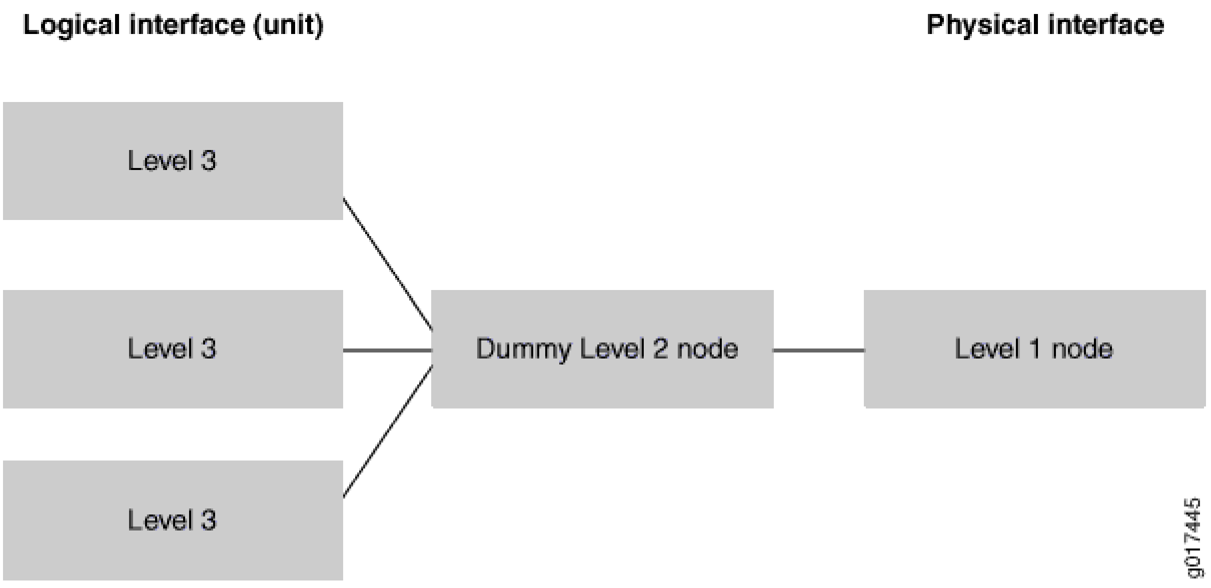
Interfaces hosted on MPCs support different scheduler node levels, depending on the MPC type:

Scheduler Node Levels for Per-Unit Scheduling MPCs

For an interface hosted on a per-unit scheduling MPC, each logical interface has its own dedicated level 3 node, and all logical interfaces share a common level 2 node (one per port).

Figure 71 on page 1000 illustrates scheduler node levels for an interface hosted on a per-unit queuing MPC.

Figure 71: Scheduler Node Levels for Per-Unit Scheduling MPCs



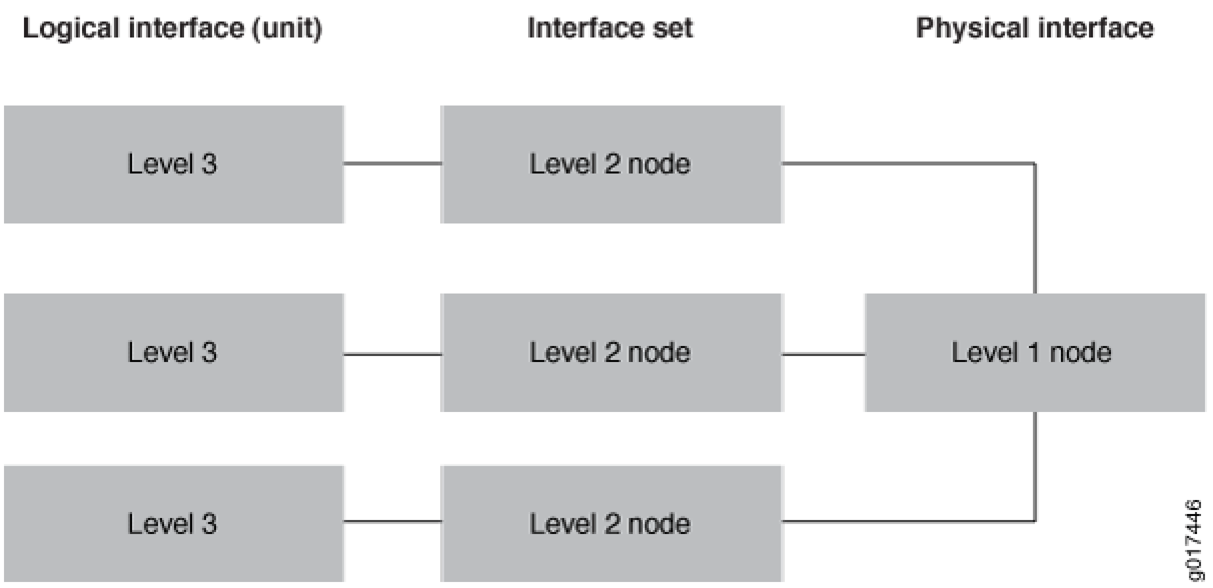
For interfaces hosted on per-unit scheduling MPCs, the level 2 node is always a dummy node.

Scheduler Node Levels for Hierarchical Scheduling MPCs

The queuing model used by interfaces hosted on hierarchical scheduling MPCs supports up to five levels of scheduler nodes: the queue itself (level 5), session logical interface (ppp or dhcp) (level 4), customer VLAN (C-VLAN) (level 3), the interface set or service VLAN (S-VLAN) collection (level 2), and the physical interface or port (level 1).

Figure 72 on page 1001 illustrates the scheduler node levels for an interface hosted on a hierarchical scheduling MIC or MPC.

Figure 72: Scheduler Node Levels for Interfaces on Hierarchical Scheduling MPCs



The figure depicts scheduler nodes for an interface that does not include interface sets and for which traffic control profiles are applied to the logical interfaces only.

NOTE: If an interface set has a CoS scheduling policy but none of its child logical interfaces has a CoS scheduling policy, then the interface set is considered to be a leaf node and has one level 2 and one level 3 node.

RELATED DOCUMENTATION

Understanding Hierarchical Scheduling for MIC and MPC Interfaces | **992**

CoS Three-Level Hierarchical Scheduling on MPLS Pseudowire Subscriber Interfaces

MX Series MPC Overview

MPCs Supported by MX Series Routers

MX Series MIC Overview

MICs Supported by MX Series Routers

MX5, MX10, MX40, and MX80 Modular Interface Card Description

Managing Dedicated and Remaining Queues for Static CoS Configurations on MIC and MPC Interfaces

IN THIS SECTION

This topic describes how to manage dedicated and remaining queues for static subscriber interfaces configured at the `[edit class-of-service]` hierarchy.

Configuring the Maximum Number of Queues for MIC and MPC Interfaces

30-Gigabit Ethernet Queuing MPCs and 60-Gigabit Ethernet Queuing and Enhanced Queuing MPCs support a dedicated number of queues when configured for hierarchical scheduling and per-unit scheduling configurations.

To scale the number of subscriber interfaces per queue, you can modify the number of queues supported on the MIC.

To configure the number of queues:

1. Specify that you want to configure the MIC.

```
user@host# edit chassis fpc slot-number pic pic-number
```

2. Configure the number of queues.

```
[edit chassis fpc slot-number pic pic-number]
user@host# set max-queues-per-interface (8 | 4)
```

SEE ALSO

Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview | 941

Configuring Remaining Common Queues on MIC and MPC Interfaces

30-Gigabit Ethernet Queuing MPCs and 60-Gigabit Ethernet Queuing and Enhanced Queuing MPCs support a dedicated set of queues when configured with hierarchical scheduling.

When the number of dedicated queues is reached on the module, there can be queues remaining. Traffic from these logical interfaces are considered unclassified and attached to a common set of queues that are shared by all subsequent logical interfaces.

You can configure traffic shaping and scheduling resources for the remaining queues by attaching a special traffic-control profile to the interface. This feature enables you to provide the same shaping and scheduling to remaining queues as the dedicated queues.

To configure the remaining queues on a MIC or MPC interface:

1. Configure CoS parameters in a traffic-control profile.

    ```
    [edit class-of-service]
    user@host# edit traffic-control-profiles profile-name
    ```

2. Enable hierarchical scheduling for the interface.

    ```
    [edit interfaces interface-name]
    user@host# set hierarchical-scheduler
    ```

3. Attach the traffic control profiles for the dedicated and remaining queues to the port on which you enabled hierarchical scheduling.

    To provide the same shaping and scheduling parameters to dedicated and remaining queues, reference the same traffic-control profile.

    a. Attach the traffic-control profile for the dedicated queues on the interface.

        ```
        [edit class-of-service interfaces interface-name]
        user@host# set output-traffic-control-profile profile-name
        ```

b. Attach the traffic-control profile for the remaining queues on the interface.

```
[edit class-of-service interfaces interface-name]
user@host# set output-traffic-control-profile-remaining profile-name
```

SEE ALSO

Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview | **941**

RELATED DOCUMENTATION

Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview

Verifying the Number of Dedicated Queues Configured on MIC and MPC Interfaces

Hierarchical CoS for Metro Ethernet Environments | **446**

Configuring Interface Sets | **355**

Excess Bandwidth Distribution on MIC and MPC Interfaces Overview

Service providers often used tiered services to provide bandwidth for excess traffic as traffic patterns vary. By default, excess bandwidth between a configured guaranteed rate and shaping rate is shared equally among all queues on MIC and MPC interfaces, which might not be optimal for all subscribers to a service.

You can adjust this distribution by configuring the rates and priorities for the excess bandwidth.

By default, when traffic exceeds the shaping or guaranteed rates, the system demotes traffic with guaranteed high (GH) priority and guaranteed medium (GM) priority. You can disable this priority demotion for the MIC and MPC interfaces in your router.

RELATED DOCUMENTATION

Managing Excess Bandwidth Distribution on Static Interfaces on MICs and MPCs

Managing Excess Bandwidth Distribution for Dynamic CoS on MIC and MPC Interfaces

Per-Priority Shaping on MIC and MPC Interfaces Overview

Traffic Burst Management on MIC and MPC Interfaces Overview

Bandwidth Management for Downstream Traffic in Edge Networks Overview

IN THIS SECTION

In a subscriber access network, traffic with different encapsulations can be passed downstream to other customer premise equipment (CPE) through the MX Series router. Managing the bandwidth of downstream ATM traffic to Ethernet interfaces can be especially difficult because of the different Layer 2 encapsulations.

The downstream network is not necessarily the directly attached network. In typical broadband network gateway (BNG) configurations, the directly attached network is an Ethernet access network, which provides access to either another frame-based network, or a cell-based network.

The *overhead accounting* feature enables you to shape traffic based on whether the downstream network is a frame-based network, like Ethernet, or a cell-based network, like ATM. It assigns a byte adjustment value to account for different encapsulations.

This feature is available on MIC and MPC interfaces.

Effective Shaping Rate

The shaping-rate, also known as peak information rate (PIR), is the maximum rate for a scheduler node or queue.

The true rate of a subscriber at the access-loop/CPE is a function of:

- The shaping-rate in effect for the subscriber’s household, in bits per second.
- Whether the subscriber is connected to a frame-based or cell-based network.
- Number of bytes in each frame that are accounted for by the shaper.

NOTE: Chassis egress-shaping-overhead is not included in the effective rate. `Egress-shaping-overhead` accounts for the physical interface overhead (ISO OSI Layer 1). Effective shaping-rate is a Layer 2 (ISO OSI) rate.

Shaping Modes

There are two modes used for adjusting downstream traffic:

- *Frame shaping mode* is useful for adjusting downstream traffic with different encapsulations. Shaping is based on the number of bytes in the frame, without regard to cell encapsulation or padding overhead. Frame is the default shaping mode on the router.
- *Cell shaping mode* is useful for adjusting downstream cell-based traffic. In cell shaping mode, shaping is based on the number of bytes in cells, and accounts for the cell encapsulation and padding overhead.

  When you specify cell mode, the resulting traffic stream conforms to the policing rates configured in downstream ATM switches, reducing the number of packet drops in the Ethernet network.

  To account for ATM segmentation, the router adjusts all of the rates by 48/53 to account for 5-byte ATM AAL5 encapsulation. In addition, the router accounts for cell padding, and internally adjusts each frame by 8 bytes to account for the ATM trailer.

Byte Adjustments

When the downstream traffic has different byte sizes per encapsulation, it is useful to configure a *byte adjustment* value to adjust the number of bytes per packet to be included in or excluded from the shaping mechanism. This value represents the number of bytes that are encapsulated and decapsulated by the downstream equipment. For example, to properly account for a 4-byte header stripped by the downstream network, set the overhead-accounting bytes to -4. To properly account for a 12-byte header added by the downstream network, set the overhead-accounting bytes to 12.

We recommend that you specify a byte adjustment value that represents the difference between the CPE protocol overhead and B-RAS protocol overhead.

The system rounds up the byte adjustment value to the nearest multiple of 4. For example, a value of 6 is rounded to 8, and a value of –10 is rounded to –8.

You do not need to configure a byte adjustment value to account for the downstream ATM network. However, you can specify the byte value to account for additional encapsulations or decapsulations in the downstream network.

Relationship with Other CoS Features

Enabling the overhead accounting feature affects the resulting shaping rates, guaranteed rate, and excess rate parameters, if they are configured.

The overhead accounting feature also affects the egress shaping overhead feature that you can configure at the chassis level. We recommend that you use the egress shaping-overhead feature to account for the Layer 2 overhead of the outgoing interface, and use the overhead-accounting feature to account for downstream traffic with different encapsulations and cell-based networks.

When both features are configured, the total byte adjustment value is equal to the adjusted value of the overhead-accounting feature plus the value of the egress-shaping-overhead feature. For example, if the configured byte adjustment value is 40, and the router internally adjusts the size of each frame by 8, the adjusted overhead accounting value is 48. That value is added to the egress shaping overhead of 24 for a total byte adjustment value of 72.

RELATED DOCUMENTATION

Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates

Configuring Dynamic Shaping Parameters to Account for Overhead in Downstream Traffic Rates

Setting Shaping Rate and Overhead Accounting Based on PPPoE Vendor-Specific Tags

Scheduler Delay Buffering on MIC and MPC Interfaces

To control congestion at the output stage, you can configure the delay-buffer bandwidth. Scheduler delay-buffer bandwidth provides packet buffer space to absorb burst traffic up to the specified duration of delay. After the specified delay buffer becomes full, packets with 100 percent drop probability are dropped from the head of the buffer.

The buffer size is the amount of time in milliseconds of port bandwidth that a queue can use to continue to transmit packets during periods of congestion, before the buffer runs out and packets begin to drop.

For MIC and MPC interfaces the default maximum queue buffer size is:

- 500 ms for traffic rates below 1 Gbps.
- 100 ms for traffic rates of 1 Gbps and faster.
- 100 ms for all tunnel interfaces configured on MIC and MPC interfaces

You can configure an explicit buffer size ranging from 4 KB to 256 MB, depending on the MIC or MPC model. However, MIC and MPC interfaces do not support the large delay buffer size configuration statement `q-pic-large-buffer`.

Interfaces hosted on MIC and MPC line cards have a certain granularity in the application of configured delay buffer parameters. In other words, the observed hardware value might not exactly match the user-configured value. Nevertheless, the derived values are as close to the configured values as allowed.

When you configure an explicit buffer size, there are 256 points available and the closest point is chosen. High-priority and medium-priority queues use 64 points, and the low-priority queues uses 128.

RELATED DOCUMENTATION

- CoS Features and Limitations on MIC and MPC Interfaces | **938**
- Rate Shaping on MIC and MPC Interfaces | **968**
- Understanding Hierarchical Scheduling for MIC and MPC Interfaces | **992**
- Drop Profiles on MIC and MPC Interfaces | **1011**

Managing Excess Bandwidth Distribution on Static Interfaces on MICs and MPCs

Service providers often used tiered services that must provide bandwidth for excess traffic as traffic patterns vary. By default, excess bandwidth between a configured guaranteed rate and shaping rate is shared equally among all queues, which might not be optimal for all subscribers to a service.

To manage excess bandwidth:

1. Configure the parameters for the interface.

   a. Configure the shaping rate.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set shaping-rate (percent percentage | rate) <burst-size bytes>
```

> **TIP**: On MIC and MPC interfaces, the guaranteed rate and the shaping rate share the value specified for the burst size. If the guaranteed rate has a burst size specified, it is used for the shaping rate; if the shaping rate has a burst size specified, it is used

for the guaranteed rate. If you have specified a burst for both rates, the system uses the lesser of the two values.

b. Configure the excess rate.

You can configure an excess rate for all priorities of traffic.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set excess-rate (percent percentage | proportion value)
```

Optionally, you can configure an excess rate specifically for high- and low-priority traffic. When you configure the `excess-rate` statement for an interface, you cannot also configure the `excess-rate-low` and `excess-rate-high` statements.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set excess-rate-high (percent percentage | proportion value)
user@host# set excess-rate-low (percent percentage | proportion value)
```

BEST PRACTICE: We recommend that you configure either a percentage or a proportion of the excess bandwidth for all schedulers with the same parent in the hierarchy. For example, if you configure interface 1.1 with twenty percent of the excess bandwidth, configure interface 1.2 with eighty percent of the excess bandwidth.

2. (Optional) Configure parameters for the queue.

a. Configure the shaping rate.

```
[edit class-of-service scheduler scheduler-name]
user@host#set shaping-rate (rate | $junos-cos-scheduler-shaping-rate) <burst-size bytes>
```

b. Configure the excess rate.

```
[edit class-of-service scheduler scheduler-name]
user@host#set excess-rate (percent percentage | proportion value)
```

c. (Optional) Configure the priority of excess bandwidth for the queue.

```
[edit class-of-service scheduler scheduler-name]
user@host#set excess-priority (low | medium-low | medium-high | high | none)
```

TIP: For queues, you cannot configure the excess rate in these cases:

- When the `transmit-rate exact` statement is configured. In this case, the shaping rate is equal to the transmit rate and the queue does not operate in the excess region.
- When the scheduling priority is configured as `strict-high`. In this case, the queue gets all available bandwidth and never operates in the excess region.

By default, when traffic exceeds the shaping or guaranteed rates, the system demotes traffic configured with guaranteed high (GH) priority and guaranteed medium (GM) priority. To disable priority demotion, specify the `none` option. You cannot configure this option for queues configured with `transmit-rate` expressed as a percent and when the parent's guaranteed rate is set to zero.

For example, the following statements establish a traffic control profile with a shaping rate of 80 Mbps and an excess rate of 100 percent.

```
[edit class-of-service traffic-control-profiles]
tcp-example-excess {
    shaping-rate 80m;
    excess-rate percent 100;
}
```

The following statements establish a scheduler with an excess rate of 5 percent and a low priority for excess traffic.

```
[edit class-of-service scheduler]
example-scheduler {
    excess-priority low;
    excess-rate percent 5;
}
```

RELATED DOCUMENTATION

Excess Bandwidth Distribution on MIC and MPC Interfaces Overview

Hierarchical CoS for Metro Ethernet Environments | 446

Drop Profiles on MIC and MPC Interfaces

IN THIS SECTION

This topic covers the following Information

Drop Profiles on Enhanced Queuing MIC and MPC Interfaces

Enhanced queuing (EQ) interfaces on MICs and MPCs support drop profiles as follows:

- Up to 255 drop profiles
- Up to 128 tail-drop priorities for guaranteed low (GL) priorities
- Up to 64 tail-drop priorities for guaranteed high and medium priorities

Implicit Scaling of WRED Profiles

You can oversubscribe scheduler delay buffers by configuring more delay-buffer memory than the system can support. If you oversubscribe the scheduler delay buffers for MIC and MPC interfaces, the system implicitly scales down the configured weighted random early detection (WRED) profiles so that packets are dropped more aggressively from the relatively full queues. This automatic adjustment creates buffer space for packets in the relatively empty queues and provides a sense of fairness among the delay buffers.

RELATED DOCUMENTATION

CoS Features and Limitations on MIC and MPC Interfaces | 938

Intelligent Oversubscription on MIC and MPC Interfaces Overview

On the MIC and MPC interfaces on MX Series routers, as on other types of interface hardware, arriving packets are assigned to one of two preconfigured traffic classes (network control and best effort) based on their header types and destination media access control (MAC) address. Oversubscription, the situation when the incoming packet rate is much higher than the Packet Forwarding Engine and system can handle, can cause key packets to be dropped and result in a flurry of resends, making the problem worse. However, MIC and MPC interfaces handle oversubscription more intelligently and drops lower priority packets when oversubscription occurs. Protocols such as routing protocols are classified as network control. Protocols such as telnet, FTP, and SSH are classified as best effort. No configuration is necessary.

The following frames and packets are assigned to the network control traffic class:

- ARPs: Ethertype `0x0806` for ARP and `0x8035` for dynamic RARP
- IEEE 802.3ad Link Aggregation Control Protocol (LACP): Ethertype `0x8809` and `0x01` or `0x02` (subtype) in first data byte
- IEEE 802.1ah: Ethertype `0x8809` and subtype `0x03`
- IEEE 802.1g: Destination MAC address `0x01-80-C2-00-00-02` with Logical Link Control (LLC) `0xAAAA03` and Ethertype `0x08902`
- PVST: Destination MAC address `0x01-00-0C-CC-CC-CD` with LLC `0xAAAA03` and Ethertype `0x010B`
- xSTP: Destination MAC address `0x01-80-C2-00-00-00` with LLC `0x424203`
- GVRP: Destination MAC address `0x01-80-C2-00-00-21` with LLC `0x424203`
- GMRP: Destination MAC address `0x01-80-C2-00-00-20` with LLC `0x424203`
- IEEE 802.1x: Destination MAC address `0x01-80-C2-00-00-03` with LLC `0x424203`
- Any per-port `my-mac` destination MAC address
- Any configured global Integrated Bridging and Routing (IRB) `my-mac` destination MAC address

In addition, the following Layer 3 control protocols are assigned to the network control traffic class:

- IGMP query and report: Ethertype `0x0800` and carrying an IPv4 protocol or IPv6 next header field set to 2 (IGMP)
- IGMP DVMRP: IGMP field version = `1` and type = `3`
- IPv4 ICMP: Ethertype `0x0800` and IPv4 protocols = `1` (ICMP)
- IPv6 ICMP: Ethertype `0x86DD` and IPv6 next header field = `0x3A` (ICMP)
- IPv4 or IPv6 OSPF: Ethertype `0x0800` and IPv4 protocol field or IPv6 next header field = `89` (OSPF)
- IPv4 or IPv6 VRRP: IPv4 Ethertype `0x0800` or IPv6 Ethertype `0x86DD` and IPv4 protocol field or IPv6 next header field = `112` (VRRP)
- IPv4 or IPv6 RSVP: IPv4 Ethertype `0x0800` or IPv6 Ethertype `0x86DD` and IPv4 protocol field or IPv6 next header field = `46` or `134`
- IPv4 or IPv6 PIM: IPv4 Ethertype `0x0800` or IPv6 Ethertype `0x86DD` and IPv4 protocol field or IPv6 next header field = `103`
- IPv4 or IPv6 IS-IS: IPv4 Ethertype `0x0800` or IPv6 Ethertype `0x86DD` and IPv4 protocol field or IPv6 next header field = `124`
- IPv4 router alert: IPv4 Ethertype `0x0800` and IPv4 option field = `0x94` (router alert)

Also, the following Layer 4 control protocols are assigned to the network control traffic class:

- IPv4 and IPv6 BGP: IPv4 Ethertype `0x0800` or IPv6 Ethertype `0x86DD`, TCP port = `179`, and carrying an IPv4 protocol or IPv6 next header field set to 6 (TCP)
- IPv4 and IPv6 LDP: IPv4 Ethertype `0x0800` or IPv6 Ethertype `0x86DD`, TCP or UDP port = `646`, and carrying an IPv4 protocol or IPv6 next header field set to 6 (TCP) or 17 (UDP)
- IPv4 UDP/L2TP control frames: IPv4 Ethertype `0x0800`, UDP port = `1701`, and carrying an IPv4 protocol field set to 17 (UDP)
- DHCP: Ethertype `0x0800`, IPv4 protocol field set to 17 (UDP), and UDP destination port = `0x43` (DHCP service) or `0x44` (DHCP host)
- IPv4 or IPv6 UDP/BFD: Ethertype `0x0800`, UDP port = `3784`, and IPv4 protocol field or IPv6 next header field set to 17 (UDP)

Finally, any PPP encapsulation (Ethertype `0x8863` (PPPoE Discovery) or `0x8864` (PPP0E Session Control)) is assigned to the network control traffic class (queue 3).

NOTE: These classifications are preconfigured.

Jitter Reduction in Hierarchical CoS Queues

IN THIS SECTION

Queue Jitter as a Function of the Maximum Number of Queues

Each queuing chip on a Modular Interface Card (MIC) or Modular Port Concentrator (MPC) internally hosts a *rate wheel thread* that updates the *shaper credits* into the *shapers* available at each level of scheduling hierarchy. At each hierarchy level, the length of this update period determines two key characteristics of scheduling:

- The minimum buffer needed for the queue to pass packets without dropping.
- The degree of *jitter* encountered in the queue.

At each hierarchy level, the length of the rate wheel update period is dependent upon the number of entities enabled for that node level. Because traffic is queued at Level 5 (queues) and scheduled upwards to Level 1 (the port), the number of entities (queues) enabled at Level 5 determines the number of entities (logical interfaces, interface-sets, or ports) enabled at the other levels of the scheduling hierarchy. By extension, the number of queues enabled for a given scheduler node hierarchy determines the length of the update period at all hierarchy levels. Consequently, limiting the maximum number of queues supported by a hierarchical queuing MIC or MPC can reduce jitter in the queues. To configure the maximum number of queues allowed per hierarchical queuing MIC or MPC, include the `max-queues` statement at the `[edit chassis fpc` *`slot-number`*`]` hierarchy level.

Default Maximum Queues for Hierarchical Queuing MICs and MPCs

The QX chip on a MIC or MPC consists of two symmetrical halves, and each half supports a maximum of 64 K queues (128 K queues per QX chip). The 2-port and 4-port 10-Gigabit Ethernet MICs with XFP and the MPC1_Q line cards have one chipset and can support a maximum of 128 K queues, distributed across the two partitions of the single QX chip. The MPC2 Q and MPC2 EQ line cards have two chipsets and can support a maximum of 256 K queues, distributed across the four partitions of the two QX chips.

Table 101 on page 1015 lists the maximum number of queues supported by default and the corresponding rate wheel update period for each hierarchical queuing MIC or MPC.

Table 101: Default Maximum Queues and Corresponding Rate Wheel Update Periods

Router Model	Hierarchical Queuing MIC or MPC	Maximum Queues	Rate Wheel Update Period
MX5, MX10, MX40, and MX80 modular	**2-port 10-Gigabit Ethernet MIC with XFP** The chassis base board hosts one chipset-based Packet Forwarding Engine process that operates in standalone mode. The single QX chip is composed of two partitions that each support 64 K queues for egress ports.	128 K	1.6 ms
MX240, MX480, MX960, MX2010, and MX2020	**MPC1 Q** The MPC1 Q line card hosts one chipset-based Packet Forwarding Engine process that operates in fabric mode. The single QX chip is composed of two partitions that each support 64 K queues for egress ports.	128 K	1.6 ms
	MPC2 Q The MPC2 Q line card hosts two chipset-based Packet Forwarding Engine processes that operate in fabric mode. The two QX chips are composed of four partitions that each support 64 K queues for egress ports.	256 K	1.6 ms
	MPC2 EQ The MPC2 EQ line card hosts two chipset-based Packet Forwarding Engine processes that operate in fabric mode. The two QX chips are composed of four partitions that each support 64 K queues for egress ports.	256 K	2.6 ms

You can configure hierarchical queuing MICs and MPCs to support a reduced maximum number of queues. Doing so reduces the rate wheel update period used by the QX chip, which in turn reduces jitter in the queues for the egress interfaces hosted on the line card.

Shaping Rate Granularity as a Function of the Rate Wheel Update Period

Reducing the length of the QX chip rate wheel update period, in addition to reducing jitter in the hierarchical scheduling queues, also indirectly increases the shaping granularity.

For a given port line rate and scheduling hierarchy level, the shaping granularity is a function of the minimum shaper credit size and the rate wheel update period in effect as a result of the number of queues supported by the line card.

```
shaping granularity  =  minimum shaper credit size  /  rate wheel update period
```

Table 102 on page 1016 shows how shaping granularity is calculated for non-enhanced hierarchical queuing MIC and MPC line cards with default values for minimum shaper credit size and for rate wheel update period.

Table 102: Default Shaping Granularities on Non-Enhanced Queuing MICs and MPCs

Port Type	Hierarchy Level	Non-Enhanced Queuing MIC or MPC Defaults		Calculation of Shaping Granularity
		Minimum Credit	Update Period	
1 Gbps Queuing	Level 1 (port), Level 4 (queues)	`4 bytes = 32 bits`	`13.33 ms = 0.01333 sec`	`32 bits / 0.01333 sec =` **2.4 Kbps**
	Level 2, Level 3	`16 bytes = 128 bits`	`1.66 ms = 0.00166 sec`	`128 bits / 0.01333 sec =` **9.6 Kbps**
10 Gbps Queuing	Level 1 (port), Level 4 (queues)	`16 bytes = 128 bits`	`13.33 ms = 0.01333 sec`	`128 bits / 0.01333 sec =` **9.6 Kbps**
	Level 2, Level 3	`64 bytes = 512 bits`	`1.66 ms = 0.00166 sec`	`512 bits / 0.01333 sec =` **38.4 Kbps**

RELATED DOCUMENTATION

Example: Reducing Jitter in Hierarchical CoS Queues | 1017

Per-Unit Scheduling and Hierarchical Scheduling for MPC Interfaces | 999

Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 992

Example: Reducing Jitter in Hierarchical CoS Queues

IN THIS SECTION

This example shows how to reduce jitter in the output queues for VLAN ports hosted on a hierarchical queuing MPC.

Requirements

This example uses the following Juniper Networks hardware and Junos OS software:

- MX960 router in an IPv4 network running any supported Junos OS release.
- Available Gigabit Ethernet port hosted on FPC slot 2, PIC slot 0, port 0.
- Available Gigabit Ethernet port hosted on port 0 of a Gigabit Ethernet Modular Interface Card (MIC) in PIC slot 0 of a Modular Port Concentrator (MPC) in FPC slot 5.

Before you begin configuring this example, make sure that the maximum number of queues allowed for the hierarchical queuing MPC in slot 5 has not yet been configured. When you enter the `show chassis fpc 5` command from configuration mode, the `max-queues` statement should not display.

Overview

In this example you configure hierarchical scheduling on a VLAN port hosted on a hierarchical queuing MPC. To reduce jitter in the queues for all egress ports hosted on the MPC, reduce the maximum number of queues allowed for MPC.

Configuration

IN THIS SECTION

CLI Quick Configuration

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

```
set interfaces xe-2/0/0 per-unit-scheduler
set interfaces xe-2/0/0 flexible-vlan-tagging
set interfaces xe-2/0/0 unit 0 vlan-id 1
set interfaces xe-2/0/0 unit 0 family inet address 10.1.1.1/24
set interfaces xe-2/0/0 unit * classifiers ieee-802.1 ieee_jitter
set interfaces xe-5/0/0 per-unit-scheduler
set interfaces xe-5/0/0 flexible-vlan-tagging
set interfaces xe-5/0/0 unit 0 vlan-id 1
set interfaces xe-5/0/0 unit 0 family inet address 10.2.1.1/24
set class-of-service-interfaces xe-5/0/0 unit * output-traffic-control-profile tcp
set class-of-service forwarding-classes queue 0 be
set class-of-service forwarding-classes queue 1 ef
set class-of-service forwarding-classes queue 2 af
set class-of-service forwarding-classes queue 3 nc
set class-of-service schedulers be_sch priority low
set class-of-service schedulers ef_sch priority low
set class-of-service schedulers af_sch priority strict-high
set class-of-service schedulers nc_sch priority low
set class-of-service classifiers ieee_jitter forwarding-class be loss-priority low code-points
000
set class-of-service classifiers ieee_jitter forwarding-class ef loss-priority low code-points
001
set class-of-service classifiers ieee_jitter forwarding-class af loss-priority low code-points
010
set class-of-service classifiers ieee_jitter forwarding-class nc loss-priority low code-points
011
set class-of-service scheduler-maps smap_jitter forwarding-class be scheduler be_sch
set class-of-service scheduler-maps smap_jitter forwarding-class ef scheduler ef_sch
set class-of-service scheduler-maps smap_jitter forwarding-class af scheduler af_sch
set class-of-service scheduler-maps smap_jitter forwarding-class nc scheduler nc_sch
set class-of-service traffic-control-profiles tcp  scheduler-map smap_jitter
set class-of-service traffic-control-profiles tcp  shaping-rate 6g
```

Baseline Configuration

Step-by-Step Procedure

Configure hierarchical scheduling at `xe-5.0.0`.

1. To configure the VLAN 1 input and output at `xe-2/0/0.0` and `xe-5/0/0.0`:

```
[edit]
user@host# set interfaces xe-2/0/0 per-unit-scheduler
user@host# set interfaces xe-2/0/0 flexible-vlan-tagging
user@host# set interfaces xe-2/0/0 unit 0 vlan-id 1
user@host# set interfaces xe-2/0/0 unit 0 family inet address 10.1.1.1/24

user@host# set interfaces xe-5/0/0 per-unit-scheduler
user@host# set interfaces xe-5/0/0 flexible-vlan-tagging
user@host# set interfaces xe-5/0/0 unit 0 vlan-id 1
user@host# set interfaces xe-5/0/0 unit 0 family inet address 10.2.1.1/24
```

2. Map each of four queues to a forwarding class.

```
[edit]
user@host# set class-of-service forwarding-classes queue 0 be
user@host# set class-of-service forwarding-classes queue 1 ef
user@host# set class-of-service forwarding-classes queue 2 af
user@host# set class-of-service forwarding-classes queue 3 nc
```

3. Assign a packet-scheduling priority value to each forwarding class.

```
[edit]
user@host# set class-of-service schedulers be_sch priority low
user@host# set class-of-service schedulers ef_sch priority low
```

```
user@host# set class-of-service schedulers af_sch priority strict-high
user@host# set class-of-service schedulers ef_sch priority low
```

4. Customize the default IEEE 802.1p classifier (BA classifier based on Layer 2 header) by defining different values for iEEE 802.1p code points.

```
[edit]
user@host# set class-of-service classifiers ieee_jitter forwarding-class be loss-priority low
code-points 000
user@host# set class-of-service classifiers ieee_jitter forwarding-class ef loss-priority low
code-points 001
user@host# set class-of-service classifiers ieee_jitter forwarding-class af loss-priority low
code-points 010
user@host# set class-of-service classifiers ieee_jitter forwarding-class nc loss-priority low
code-points 011
```

5. Apply the BA classifier to the input of the logical units on `xe-2/0/0`.

```
[edit]
user@host# set interfaces xe-2/0/0 unit * classifiers ieee-802.1 ieee_jitter
```

6. Configure the scheduler map `smap_jitter` to map the forwarding classes to the schedulers.

```
[edit]
user@host# set class-of-service scheduler-maps smap_jitter forwarding-class be scheduler
be_sch
user@host# set class-of-service scheduler-maps smap_jitter forwarding-class ef scheduler
ef_sch
user@host# set class-of-service scheduler-maps smap_jitter forwarding-class af scheduler
af_sch
user@host# set class-of-service scheduler-maps smap_jitter forwarding-class nc scheduler
nc_sch
```

7. Configure the traffic control profile `tcp` to combine the scheduler map `smap_jitter` (that maps the forwarding classes to the schedulers for port-based scheduling) with a shaping rate (for hierarchical scheduling).

```
[edit]
user@host# set class-of-service traffic-control-profiles tcp scheduler-map smap_jitter
user@host# set class-of-service traffic-control-profiles tcp shaping-rate 6g
```

8. Apply the traffic control profile to the router output at `xe-5/0/0`.

```
[edit]
user@host# set class-of-service-interfaces xe-5/0/0 unit * output-traffic-control-profile tcp
```

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

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

Results

Confirm your configuration by entering `show interfaces` and `show cloass-of-service` commands from configuration mode. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
[edit]
user@host# show interfaces
xe-2/0/0 {
    per-unit-scheduler;
    flexible-vlan-tagging;
    unit 0 {
        vlan-id 1;
        family inet {
            address 10.1.1.1/24;
        }
    }
}
xe-5/0/0 {
    per-unit-scheduler;
    flexible-vlan-tagging;
    unit 0 {
        vlan-id 1;
        family inet {
            address 10.2.1.1/24;
        }
    }
}
```

```
[edit]
user@host# **show class-of-service**
classifiers {
    ieee-802.1 ieee_jitter {
        forwarding-class be {
            loss-priority low code-points 000;
        }
        forwarding-class ef {
            loss-priority low code-points 001;
        }
        forwarding-class af {
            loss-priority low code-points 010;
        }
        forwarding-class nc {
            loss-priority low code-points 011;
        }
    }
}
forwarding-classes {
    queue 0 be;
    queue 1 ef;
    queue 2 af;
    queue 3 nc;
}
traffic-control-profiles {
    tcp {
        scheduler-map smap_jitter;
        shaping-rate 6g;
    }
}
interfaces {
    xe-2/0/0 {
        unit * {
            classifiers {
                ieee-802.1 ieee_jitter;
            }
        }
    }
    xe-5/0/0 {
        unit * {
            output-traffic-control-profile tcp;
        }
    }
}
scheduler-maps {
    smap_jitter {
        forwarding-class be scheduler be_sch;
        forwarding-class ef scheduler ef_sch;
        forwarding-class af scheduler af_sch;
        forwarding-class nc scheduler nc_sch;
    }
}
schedulers {
    be_sch {
        priority low;
    }
    ef_sch {
        priority low;
    }
    af_sch {
        priority strict-high;
    }
    nc_sch {
        priority low;
    }
}
```

Verification

IN THIS SECTION

Confirm that the configuration is working properly

Measuring End-to-End Jitter to Establish the Baseline

Purpose

Establish a baseline measurement by noting the amount of jitter that occurs when the hierarchical queuing line card hosting the egress port is configured with the default maximum number of queues.

Action

To measure jitter:

1. Pass traffic through the VLAN.
2. Measure the variation in packet delay for selected packets in the data flow.

Configuring Jitter Reduction

Purpose

Reduce jitter in the VLAN port output queues.

Action

1. Configure a reduced maximum number of queues for egress ports on the hierarchical queuing MPC in slot 5, thereby reducing the jitter in the port queues.

   ```
   [edit]
   user@host# set chassis fpc 5 max-queue 64k
   ```

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

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

Measuring End-to-End Jitter to Verify Jitter Reduction

Purpose

Measure the amount of jitter that occurs when the hierarchical queuing line card hosting the egress port is configured with a reduced maximum number of queues.

Action

To measure jitter:

1. Pass traffic through the VLAN.

2. Measure the variation in packet delay for selected packets in the data flow.

RELATED DOCUMENTATION

Jitter Reduction in Hierarchical CoS Queues | 1014

max-queues

CoS on Ethernet Pseudowires in Universal Edge Networks Overview

You can apply *rewrite rules* and classifiers to an Ethernet pseudowire on MIC and MPC interfaces on MX Series routers. In an edge network, the pseudowire can represent a single customer.

To create the pseudowires, you use logical tunnel (LT) interfaces that connect two virtual routing forwarding (VRF) instances. To provide CoS to the LT interface, you can apply classifiers and rewrite rules. Rewrite rules enable you to rewrite packet header information by specifying various CoS values, including DiffServ code point (DSCP) and IP precedence.

NOTE: Scheduling is not supported on LT interfaces in the current release.

For example, a VPLS instance is connected to a Layer 3 routing instance. The logical tunnel labeled `lt-9/0/0.0` is configured with `vpls` as the family, and `lt-9/0/0.1` is configured with `inet` as the family. You can apply a rewrite rule and classifier for DSCP to `lt-9/0/0.1`, which can represent a business subscriber.

RELATED DOCUMENTATION

Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks | **1026**

CoS Scheduling Policy on Logical Tunnel Interfaces Overview

You can configure a CoS scheduling policy on a logical tunnel interface (LT ifl). Logical tunnel interfaces can be used to terminate a pseudowire into a virtual routing and forwarding (VRF) instance. If an LT device is used to terminate a pseudowire, CoS scheduling policies can be applied on the LT interface to manage traffic entering the pseudowire. You accomplish this by configuring the hierarchical-scheduler attribute on the physical interface.

This feature is supported on MIC and MPC interfaces.

RELATED DOCUMENTATION

Configuring a CoS Scheduling Policy on Logical Tunnel Interfaces | **996**

Hierarchical CoS for Metro Ethernet Environments | **446**

CoS on Ethernet Pseudowires in Universal Edge Networks Overview | **1025**

Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks | **1026**

Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks

You can configure rewrite rules and classifiers to logical tunnel (LT) interfaces that are configured to represent Ethernet pseudowires.

This feature is supported on MIC and MPC interfaces.

To configure CoS on an LT interface configured for an Ethernet pseudowire:

1. Configure a pair of LT interfaces to represent a pseudowire.

To apply rewrite rules and classifiers to the pseudowire, you must assign one of the LT interfaces to the `inet` family.

```
[edit]
user@host#edit interfaces lt-fpc/pic/port
user@host#edit unit logical-unit-number
user@host#set encapsulation encapsulation
user@host#set family (inet | inet6 | iso | mpls)
user@host#set peer-unit unit-number
```

2. Configure the rewrite rule.

   The available rewrite rule types for an LT interface are `dscp` and `inet-precedence`.

```
[edit class-of-service]
user@host#edit rewrite-rules (dscp | inet-precedence) rewrite-name
user@host#edit forwarding-class class-name
user@host#set loss-priority class-name code-point (alias | bits)
```

3. Configure the classifier.

   The available classifier types for an LT interface are `dscp` and `inet-precedence`.

```
[edit class-of-service]
user@host#edit classifiers (dscp | inet-precedence) classifier-name
user@host#edit forwarding-class class-name
user@host#set loss-priority class-name code-points [aliases] [bit-patterns]
```

4. Apply the rewrite rule and classifier to the LT interface that you assigned to the `inet` family.

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
user@host#set rewrite-rule (dscp | inet-precedence) (rewrite-name | default) protocol
protocol-types
user@host# set classifiers (dscp | inet-precedence) (classifier-name | default)
```

RELATED DOCUMENTATION

CoS on Ethernet Pseudowires in Universal Edge Networks Overview | **1025**

Rewriting Packet Headers to Ensure Forwarding Behavior | **558**

How Behavior Aggregate Classifiers Prioritize Trusted Traffic | **60**

CoS for L2TP LNS Inline Services Overview

IN THIS SECTION

You can apply hierarchical scheduling and per-session shaping to Layer 2 Tunnel Protocol (L2TP) network server (LNS) inline services using a static or dynamic CoS configuration.

Guidelines for Applying CoS to the LNS

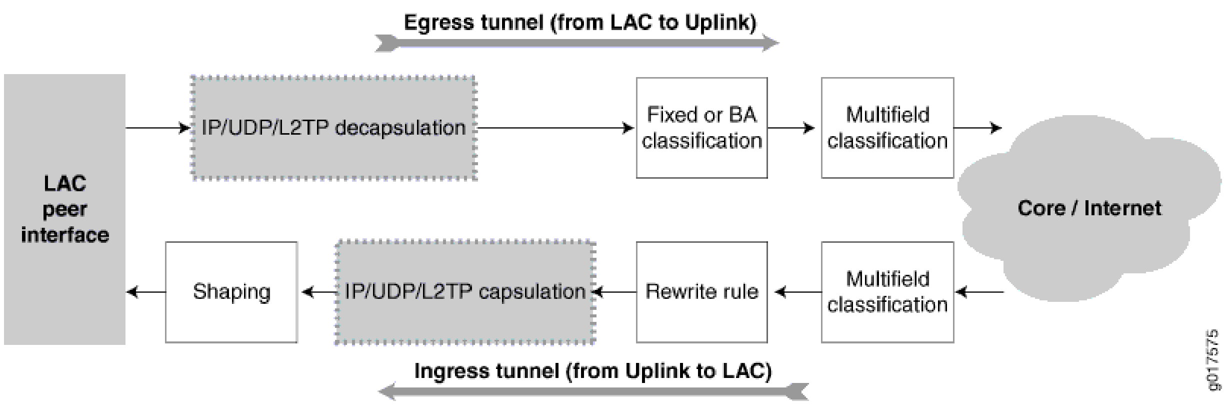
In L2TP configurations, IP, UDP, and L2TP headers are added to packets arriving at a PPP subscriber interface on the L2TP access concentrator (LAC) before being tunneled to the LNS.

When a service interface is configured for an L2TP LNS session, it has an *inner* IP header and an outer IP header. You can configure CoS for an LNS session that corresponds to the inner IP header only. The *outer* IP header is used for L2TP tunnel processing only.

However, we recommend that you configure classifiers and rewrite-rules to transfer the ToS (type of service) value from the inner IP header to the outer IP header of the L2TP packet.

Figure 73 on page 1028 shows the classifier and *rewrite rules* that you can configure on an LNS inline service.

Figure 73: Processing of CoS Parameters in an L2TP LNS Inline Service



Egress tunnel (from LAC to Uplink)
LAC peer interface
IP/UDP/L2TP decapsulation
Fixed or BA classification
Multifield classification
Core / Internet
Shaping
IP/UDP/L2TP capsulation
Rewrite rule
Multifield classification
Ingress tunnel (from Uplink to LAC)
g017575

By default, the shaping calculation on the service interface includes the L2TP encapsulation.

RELATED DOCUMENTATION

Configuring Static CoS for an L2TP LNS Inline Service | 1029

Configuring Dynamic CoS for an L2TP LNS Inline Service

Configuring Static CoS for an L2TP LNS Inline Service

Before you begin, configure the L2TP LNS inline service interface. See *Configuring an L2TP LNS with Inline Service Interfaces*.

You can configure hierarchical scheduling for an L2TP LNS inline service and manage the IP header values using rewrite rules and classifiers.

To configure static CoS for an L2TP LNS inline service:

1. Configure the hierarchical scheduler for the service interface (si) interface.

```
[edit interfaces si-fpc/port/pic ]
user@host# set hierarchical-scheduler maximum-hierarchy-levels 2
```

BEST PRACTICE: To enable Level 3 nodes in the LNS scheduler hierarchy and to provide better scaling, we recommend that you also specify a maximum of two hierarchy levels.

2. Configure the LNS to reflect the IP ToS value in the inner IP header to the outer IP header.

```
[edit services l2tp tunnel-group name]
user@host# set tos-reflect
```

3. Configure the classifier for egress traffic from the LAC:

   a. Define the fixed or behavior aggregate (BA) classifier.

   - To configure a fixed classifier:

```
[edit class-of-service interfaces si-fpc/port/pic  unit logical-unit-number]
user@host# set forwarding-class class-name
```

- To configure a BA classifier:

```
[edit class-of-service]
user@host# set classifiers (dscp | dscp-ipv6 | inet-precedence) classifier-name
forwarding-class class-name loss-priority level code-points [ aliases ] [ bit-patterns]
```

b. Apply the classifier to the service interface.

- To apply the classifier for the DSCP or DSCP IPv6 value:

```
[edit class-of-service interfaces si-fpc/port/pic  unit logical-unit-number classifiers]
user@host# set dscp (classifier-name | default)
user@host# set dscp-ipv6 (classifier-name | default)
```

- To apply the classifier for the ToS value:

```
[edit class-of-service interfaces si-fpc/port/pic  unit logical-unit-number classifiers]
user@host# set inet-precedence (classifier-name | default)
```

4. Configure and apply a rewrite-rule to ingress traffic to the LAC:

a. Configure the rewrite rule with the forwarding class and the loss priority value.

```
[edit class-of-service]
user@host# set rewrite-rules (dscp | dscp-ipv6 | inet-precedence) rewrite-name forwarding-
class class-name loss-priority level code-point (alias | bits)
```

b. Apply the rewrite rule to the service interface.

- To apply the rewrite rule for the DSCP or DSCP IPv6 value:

```
[edit class-of-service interfaces si-fpc/port/pic  unit logical-unit-number rewrite-
rules]
user@host# set dscp(rewrite-name | <default>) protocol protocol-types
user@host# set dscp-ipv6 (rewrite-name | <default>)
```

- To apply the rewrite rule for the ToS value:

```
[edit class-of-service interfaces si-fpc/port/pic  unit logical-unit-number rewrite-
rules]
user@host# set inet-precedence (rewrite-name | <default>) protocol protocol-types
```

5. Apply the traffic-control profile.

```
[edit class-of-service interfaces si-fpc/port/pic  unit logical-unit-number]
user@host# set output-traffic-control-profile profile-name
```

BEST PRACTICE: To limit bandwidth for tunneled sessions with default CoS configurations, we recommend that you also configure CoS for the remaining traffic on the static service interface.

```
[edit class-of-service interfaces si-fpc/port/pic ]
user@host# set output-traffic-control-profile-remaining profile-name
```

RELATED DOCUMENTATION

CoS for L2TP LNS Inline Services Overview | **1028**

Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates

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

On supported platforms, you can configure 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 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 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 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

Understanding swap-by-poppush

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

Understanding Transparent Tag Operations and IEEE 802.1p Inheritance

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

swap-by-poppush

Understanding IEEE 802.1p Inheritance push and swap from a Transparent Tag | **1031**

Understanding swap-by-poppush

Understanding Transparent Tag Operations and IEEE 802.1p Inheritance

CoS on Application Services Modular Line Card Overview

IN THIS SECTION

The Application Services Modular Line Card (AS-MLC) is designed to run services for real-time traffic. It consists of three main components:

- Application Services Modular Carrier Card (AS-MCC)
- Application Services Modular Processing Card (AS-MXC)
- Application Services Modular Storage Card (AS-MSC)

The AS-MLC supports CoS features to ensure *quality of service* (QoS) for real-time traffic that is sensitive to latency on a network. The AS-MLC supports the following CoS features:

- Code-point Aliases—On the AS-MLC, you can use the code-point alias name for CoS components such as classifiers and drop-profile maps.

- Classification—On the AS-MLC, the traffic flowing from the AS-MXC toward the AS-MCC supports three types of classification:
  - Behavior Aggregate (BA)—You can configure BA classifiers on the aggregated logical interfaces to classify traffic flowing from the AS-MXC toward the AS-MCC. With BA classification you can set the forwarding class and loss priority of a packet based on its code points. The AS-MLC only supports IP classification (classification based on ToS and DSCP) and classification is supported for the IPv4 family only. The Media Flow Controller application sets appropriate DSCP/ToS code-point in the packet that is evaluated by the BA classifier on the AS-MCC to classify the packet.
  - Multifield Classification—With multifield classifiers you can set the class and loss priority based on one or more of the following packet header fields: destination address, destination port, DSCP, IP protocol, and source address.
  - Fixed Classification—Fixed classification can be configured on logical interfaces by specifying a forwarding class to be applied to all packets received by the *logical interface*, regardless of the packet contents.
- Scheduling—Schedulers enable you to define the buffer sizes, delay buffer size, drop profile map, excess priority, excess rate percentage, output-traffic-control profile, priority, scheduler-map, shaping rate, transmit rate, and RED drop profiles to apply to a particular queue for packet transmission.

The AS-MLC provides CoS features in the following deployment scenarios:

- HTTP Reverse Proxy— In HTTP reverse proxy configurations, the service provider provides services to a set of domains (content providers) that buy content caching capability from the service provider. Clients connect to content providers through virtual IP (VIP) addresses. Service providers in the reverse proxy scenario generally deploy the routers with AS-MLC hardware to honor service requests (such as caching) from the domain users.
- HTTP Transparent Proxy—In HTTP transparent proxy configurations, the service provider implements the AS-MLC to improve its own caching capability and reduce the load on its own network. Implementing caching on a router with an AS-MLC improves the retrieval speeds for data and optimizes the back-end network utilization.
- Mixed Mode—In mixed mode both reverse proxy and transparent proxy are configured on the same router.

CoS Implementation in HTTP Reverse Proxy Scenario

In the reverse proxy configuration, the AS-MXC provides content to multiple domains. The Media Flow Controller application on an AS-MXC implements DiffServ by setting the DSCP or IP precedence value for the IP packets traversing from the AS-MXC to an AS-MCC on the AS-MLC hardware. The Modular Carrier Card uses these values to classify the packet and provide a suitable level of service.

The Media Flow Controller application detects the domain it serves and marks the DSCP values or the IP precedence bit value based on how important the traffic corresponding to that particular domain is. The service provider operator also sets a BA classifier on the aggregated interfaces on the AS-MCC.

Unlike a firewall, the Media Flow Controller application implemented on the AS-MLC hardware marks the DSCP/IP precedence values based on the application layer protocols. This feature ensures that important traffic flowing from the AS-MXC gets a higher priority and is processed accordingly. For example, if you implement MPEG on the egress, the drop preference for each frame can be different such that the P and B frames (which require more processing) are dropped before the I frames. This approach results in a better quality video for the end user.

End-to-end CoS policies ensure that traffic arriving at an interface has the right DSCP values and the traffic is prioritized based on the forwarding class and PLP values.

CoS Implementation in Transparent Proxy Scenario

In the HTTP transparent proxy configuration, the service provider deploys the AS-MLC hardware to reduce its own traffic instead of serving a particular domain. The Media Flow Controller application marks the DSCP bits based on its own requirements rather than the requirements of the domains. Besides this difference, the CoS implementation for the egress interface is similar to the reverse proxy configuration scenario. The incoming packets follow the CoS policies applied at the WAN interface.

CoS Implementation in Mixed-Mode Scenario

In mixed mode both reverse proxy and transparent proxy configuration coexist on the same AS MLC hardware. In such a scenario, reverse proxy is configured on an aggregated interface and transparent proxy is configured on a regular interface with the Media Flow Controller application marking the appropriate DSCP values for both the configurations. The individual CoS implementation in both the scenarios remains similar to the implementation discussed in "CoS Implementation in HTTP Reverse Proxy Scenario" on page 1036 and "CoS Implementation in Transparent Proxy Scenario" on page 1037

CHAPTER 20

Configuring Class of Service on Aggregated, Channelized, and Gigabit Ethernet Interfaces

IN THIS CHAPTER

Limitations on CoS for Aggregated Interfaces

Ethernetinterfaces can be aggregated.

There are some restrictions when you configure CoS on aggregated Ethernet interfaces:

- Chassis scheduling is not supported on aggregated interfaces, because a chassis scheduler applies to the entire PIC and not just to one interface.
- An aggregated interface is a pseudo-interface. Therefore, CoS queues are not associated with the aggregated interface. Instead, CoS queues are associated with the member link interfaces of the aggregated interface.
- When you apply CoS parameters to the aggregated interface, they are applied to the CoS queues of the member link interfaces. You do not, however, apply CoS classifiers and *rewrite rules* directly to the member link interfaces.
- You cannot apply a scheduler map to a member link of an aggregate interface.

- Rate-based CoS components such as scheduler, shaper, and policer are not supported on mixed rate aggregated Ethernet links. However, the default CoS settings are supported by default on the mixed rate aggregated Ethernet links.
- Ingress queuing is not supported on aggregated interfaces.

When the scheduler map of the aggregate interface has schedulers configured for absolute transmit rate, the scheduler for the member link interfaces is scaled to the speed of each member link interface. Each member link interface has an automatic scheduler map that is not visible in the CLI. This scheduler map is allocated when the member link is added to the aggregate interface and is deleted when the member link is removed from the aggregate interface.

- If you configure the scheduler transmit rate of the aggregate interface as an absolute rate, the software uses the following formula to scale the transmit rate of each member link:

```
transmit rate of member link interface =
(configured transmit rate of aggregate interface /
total speed of aggregate interface) *
(total speed of member link interface / total configured percent) * 100
```

- If you configure the scheduler transmit rate of the aggregate interface as a percentage, the software uses the following formula to scale the transmit rate of each member link:

```
transmit rate percent of member link interface =
(configured transmit rate percent of aggregate interface /
total configured percent) * 100
```

The total configured percent is the sum of the configured transmit rate of all schedulers in terms of percentage of the total speed of the aggregate interface.

- All the other parameters for the schedulers, including priority, drop profile, and buffer size, are copied without change from the scheduler of the aggregated interface to the member link interfaces.
- The configuration related to the logical interfaces, including classifiers and rewrite rules, is copied from the aggregated *logical interface* configuration to the member link logical interfaces.
- For the scheduler map applied to an aggregated interface, if you configure a transmission rate in absolute terms, then the traffic of all the member link interfaces might be affected if any of the member link interfaces go up or down.

When applying CoS configurations to bundles, you must apply the CoS configuration directly to the bundle, not to the physical ports that are part of the bundle. The device may give you a false commit if you apply a CoS configuration directly to a physical port that is part of a bundle. This limitation applies if you attempt to configure a physical port that is already a member of a bundle or if you attempt to add a physical port to a bundle that already has a CoS configuration applied to it.

If you want to add a physical port to a bundle that already has a CoS configuration, you must:

1. Remove the CoS configuration from the port.
2. Commit your changes on the device.
3. Add the port to the bundle. The CoS configurations that are present on the bundle will be applied to the port you are adding to the bundle.
4. Commit you changes on the device.

In addition, if you want to remove a physical port from a bundle and ensure the physical port has the appropriate CoS configurations, you must:

1. Remove the port from the bundle.
2. Commit your changes on the device.
3. Apply the applicable CoS configuration to the port.
4. Commit your changes on the device.

RELATED DOCUMENTATION

Examples: Configuring CoS on Aggregated Interfaces | 1042

Policer Support for Aggregated Ethernet Interfaces Overview

Aggregated interfaces support single-rate policers, three-color marking policers, two-rate three-color marking policers, hierarchical policers, and percentage-based policers. By default, policer bandwidth and burst-size applied on aggregated bundles is not matched to the user-configured bandwidth and burst-size.

You can configure interface-specific policers applied on an aggregated Ethernet bundle or an aggregated SONET bundle to match the effective bandwidth and burst-size to user-configured values. The `shared-bandwidth-policer` statement is required to achieve this match behavior.

This capability applies to all interface-specific policers of the following types: single-rate policers, single-rate three-color marking policers, two-rate three-color marking policers, and hierarchical policers. Percentage-based policers match the bandwidth to the user-configured values by default, and do not require shared-bandwidth-policer configuration. The `shared-bandwidth-policer` statement causes a split in burst-size for percentage-based policers.

The following usage scenarios are supported:

- Interface policers used by the following configuration:

  ```
  [edit] interfaces (aeX | asX) unit unit-num family  family policer [input | output | arp]
  ```

- Policers and three-color policers (both single-rate three-color marking and two-rate three-color marking) used inside interface-specific filters; that is, filters that have an interface-specific keyword and are used by the following configuration:

  ```
  [edit] interfaces (aeX | asX) unit unit-num family family filter [input | output]
  ```

- Common-edge service filters, which are derived from CLI-configured filters and thus inherit interface-specific properties. All policers and three-color policers used by these filters are also affected.

The following usage scenarios are not supported:

- Policers and three-color policers used inside filters that are not interface specific; such a filter is meant to be shared across multiple interfaces.

- Any implicit policers or policers that are part of implicit filters; for example, the default ARP policer applied to an aggregate Ethernet interface. Such a policer is meant to be shared across multiple interfaces.

- Prefix-specific action policers.

To configure this feature, include the `shared-bandwidth-policer` statement at the following hierarchy levels: `[edit firewall policer *policer-name*]`, `[edit firewall three-color-policer *policer-name*]`, or `[edit firewall hierarchical-policer *policer-name*]`.

RELATED DOCUMENTATION

shared-bandwidth-policer

Examples: Configuring CoS on Aggregated Interfaces

This example illustrates how CoS scheduler parameters are configured and applied to aggregated interfaces.

Applying Scaling Formula to Absolute Rates

Configure queues as follows when the total speed of member link interfaces is 100 Mbps (the available bandwidth is 100 Mbps):

```
[edit class-of-service]
schedulers {
    be {
        transmit-rate 10m;
    }
    af {
        transmit-rate 20m;
    }
    ef {
        transmit-rate 80m;
    }
    nc {
        transmit-rate 30m;
    }
}
```

The total configured transmit rates of the aggregated interface is `10m` + `20m` + `80m` + `30m` = 140 Mbps, meaning the transmit rate is overconfigured by 40 percent. Therefore, the software scales down the configuration to match the 100 Mbps of available bandwidth, as follows:

```
be = (10/140) * 100 = 7 percent of 100 Mbps = 7 Mbps
af = (20/140) * 100 = 14 percent of 100 Mbps = 14 Mbps
ef = (80/140) * 100 = 57 percent of 100 Mbps = 57 Mbps
nc = (30/140) * 100 = 21 percent of 100 Mbps = 21 Mbps
```

Applying Scaling Formula to Mixture of Percent and Absolute Rates

Configure the following mixture of percent and absolute rates:

```
[edit class-of-service]
schedulers {
    be {
        transmit-rate 20 percent;
    }
    af {
        transmit-rate 40 percent;
    }
    ef {
        transmit-rate 150m;
    }
    nc {
        transmit-rate 10 percent;
    }
}
```

Assuming 300 Mbps of available bandwidth, the configured percentages correlate with the following absolute rates:

```
schedulers {
    be {
        transmit-rate 60m;
    }
    af {
        transmit-rate 120m;
    }
    ef {
        transmit-rate 150m;
    }
    nc {
        transmit-rate 30m;
    }
}
```

The software scales the bandwidth allocation as follows:

```
be = (60/360) * 100 = 17 percent of 300 Mbps = 51 Mbps
af = (120/360) * 100 = 33 percent of 300 Mbps = 99 Mbps
ef = (150/360) * 100 = 42 percent of 300 Mbps = 126 Mbps
nc = (30/360) * 100 = 8 percent of 300 Mbps = 24 Mbps
```

Configuring an Aggregated Ethernet Interface

Configure an aggregated Ethernet interface with the following scheduler map:

```
[edit class-of-service]
scheduler-maps {
    aggregated-sched {
        forwarding-class be scheduler be;
        forwarding-class af scheduler af;
        forwarding-class ef scheduler ef;
        forwarding-class nc scheduler nc;
    }
}
schedulers {
    be {
        transmit-rate percent 10;
        buffer-size percent 25;
    }
    af {
        transmit-rate percent 20;
        buffer-size percent 25;
    }
    ef {
        transmit-rate 80m;
        buffer-size percent 25;
    }
    nc {
        transmit-rate percent 30;
        buffer-size percent 25;
    }
}
```

In this case, the transmission rate for the member link scheduler map is as follows:

- `be`—7 percent
- `af`—14 percent
- `ef`—57 percent
- `nc`—21 percent

If you add an Ethernet interface to the aggregate, the aggregate bandwidth is 200 Mbps, and the transmission rate for the member link scheduler map is as follows:

- `be`—10 percent

- `af`—20 percent
- `ef`—40 percent
- `nc`—30 percent

RELATED DOCUMENTATION

Limitations on CoS for Aggregated Interfaces | 1038

Hierarchical Schedulers on Aggregated Ethernet Interfaces Overview

On MX Series routers, you can apply hierarchical schedulers on aggregated ethernet bundles using interface sets. This feature enables you to configure a group of virtual LANs (VLANs) and control their bandwidth. This feature is supported at egress only.

You can configure interface sets for aggregated Ethernet (AE) interfaces created under static configurations. You can configure class-of-service parameters on AE interfaces, in either link-protect or non-link-protect mode. You can configure these parameters at the AE physical interface level. The CoS configuration is fully replicated for all AE member links in link-protect mode. You can control the way these parameters are applied to member links in non-link-protect mode by configuring the AE interface to operate in scaled mode or replicate mode.

The link membership list and scheduler mode of the interface set is inherited from the underlying aggregated Ethernet interface over which the interface set is configured. When an aggregated Ethernet interface operates in link protection mode, or if scheduler mode is configured to replicate member links, the scheduling parameters of the interface set are copied to each of the member links.

If the scheduler mode of the aggregated Ethernet interface is set to scale member links, the scheduling parameters are scaled based on the number of active member links (scaling factor is 1/A where A is the number of active links in the bundle) and applied to each of the AE interface member links.

To configure an interface set, include the `interface-set` statement at the `[edit class-of-service interfaces]` hierarchy level.

To apply scheduling and queuing parameters to the interface set, include the `output-traffic-control-profile` *`profile-name`* statement at the `[edit class-of-service interfaces` *`interface-name`* `interface-set` *`interface-set-name`*`]` hierarchy level.

To apply an output traffic scheduling and shaping profile for the remaining traffic to the *logical interface* or interface set, include the `output-traffic-control-profile-remaining` *`profile-name`* statement at the `[edit` `class-of-service interfaces` *`interface-name`*`]` hierarchy level or the `[edit class-of-service interfaces` *`interface-name`* `interface-set` *`interface-set-name`*`]` hierarchy level.

RELATED DOCUMENTATION

Configuring Hierarchical Schedulers on Aggregated Ethernet Interfaces | 1046

output-traffic-control-profile-remaining

Controlling Remaining Traffic | 388

Configuring Hierarchical Schedulers on Aggregated Ethernet Interfaces

The following example shows the creation of an interface set for aggregated Ethernet interfaces in a static Ethernet configuration.

To configure interface sets for aggregated Ethernet (AE) interfaces created under static configurations:

1. Create the AE interfaces.

```
[edit]
user@host# show chassis | display set
set chassis aggregated-devices ethernet device-count 10
```

2. Configure the AE physical interfaces and member links.

   user@host# show interfaces | display set

```
set interfaces ge-5/2/0 gigether-options 802.3ad ae0
set interfaces ge-5/2/1 gigether-options 802.3ad ae0
set interfaces ae0 hierarchical-scheduler maximum-hierarchy-levels 2
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 1 vlan-id 101
set interfaces ae0 unit 2 vlan-id 102
set interfaces ae0 unit 3 vlan-id 103
set interfaces ae0 unit 4 vlan-id 104
```

3. Configure the interface set.

```
set interfaces interface-set ifset1-ae0 interface ae0 unit 0
set interfaces interface-set ifset1-ae0 interface ae0 unit 1
```

4. Configure class-of-service parameters for the interface sets.

```
set class-of-service interfaces interface-set ifset1-ae0 output-traffic-control-profile tcp
```

NOTE: You also need to configure the parameters of the traffic control profile. For more information, see the Related Documentation section on this page.

5. Configure scheduler mode.

```
set class-of-service interfaces ae0 member-link-scheduler scale
```

RELATED DOCUMENTATION

Configuring Traffic Control Profiles for Shared Scheduling and Shaping | **350**

Hierarchical Schedulers on Aggregated Ethernet Interfaces Overview | **1045**

Configuring Traffic Control Profiles for Shared Scheduling and Shaping | **350**

Example: Configuring Scheduling Modes on Aggregated Interfaces

You can configure class-of-service parameters, such as queuing or shaping parameters on aggregated interfaces, in either link-protect or non-link-protect mode. You can configure these parameters for per-unit schedulers, hierarchical schedulers, or shaping at the physical and logical interface level. You can control the way these parameters are applied by configuring the aggregated interface to operate in `scale` or `replicate` mode.

You can configure the applied parameters for aggregated interfaces operating in non-link-protected mode. In link-protected mode, only one link in the bundle is active at a time (the other link is a backup link) so schedulers cannot be scaled or replicated. In non-link-protected mode, all the links in the bundle are active and send traffic; however, there is no backup link. If a link fails or is added to the bundle in non-link-protected mode, the links' traffic is redistributed among the active links.

To set the scheduling mode for aggregated interfaces, include the `scale` or `replicate` option of the `member-link-scheduler` statement at the `[edit class-of-service interfaces ae`*`n`*`]` hierarchy level, where *n* is the configured number of the interface:

```
[edit class-of-service interfaces aen]
member-link-scheduler (replicate | scale);
```

By default, if you do not include the `member-link-scheduler` statement, scheduler parameters are applied to the member links in the `scale` mode (also called “equal division mode”).

The aggregated Ethernet interfaces are otherwise configured as usual. For more information on configuring aggregated Ethernet interfaces, see the Junos OS Network Interfaces Library for Routing Devices.

The following examples set `scale` mode on the `ae0` interface and `replicate` mode on the `ae1` interface.

```
[edit class-of-service]
interfaces ae0 {
    member-link-scheduler scale;
}

[edit class-of-service]
interfaces ae1 {
    member-link-scheduler replicate;
}
```

NOTE: The `member-link-scheduler` statement only appears for aggregated interfaces. You configure this statement for aggregated interfaces in non-link-protected mode. For more information about link protection modes, see the *Network Interfaces Configuration Guide*.

Aggregated interfaces support both hierarchical and per-unit schedulers.

When interface parameters are using the `scale` option of the `member-link-scheduler` statement, the following parameters under the `[edit class-of-service traffic-control-profiles` *`traffic-control-profile-name`*`]` configuration are scaled on egress when hierarchical schedulers are configured:

- `shaping-rate` (PIR)
- `guaranteed-rate` (CIR)
- `delay-buffer-rate`

When interface parameters are using the `scale` option of the `member-link-scheduler` statement, the following parameters under the `[edit class-of-service schedulers` *`scheduler-name`*`]` configuration are scaled on egress when per-unit schedulers are configured:

- `transmit-rate`
- `buffer-size`

NOTE: You cannot apply a hierarchical scheduler at the interface set level for an `ae` interface. (Interface sets cannot be configured under an `ae` interface.)

The following configuration parameters are not supported on `ae` interfaces in non-link-protection mode:

- Input scheduler maps
- Input traffic control profiles
- Input shaping rates

The following configuration conventions are also not supported:

- Scaling of the `input-traffic-control-profile-remaining` statement.
- The `scheduler-map-chassis` statement and the `derived` option for the `ae` interface. Chassis scheduler maps should be applied under the physical interfaces.
- Dynamic and demux interfaces are not supported as part of the `ae` bundle.

Depending on the whether the `scale` or `replicate` option is configured, the `member-link-scheduler` statement operates in either scaled mode (also called “equal division mode”) or replicated mode, respectively.

In scaled mode, a VLAN can have multiple flows that can be sent over multiple member links of the `ae` interface. Likewise, a member link can receive traffic from any VLAN in the `ae` bundle. In scaled mode, the physical interface bandwidth is divided equally among all member links of the `ae` bundle.

In scaled mode, the following scheduler parameter values are divided equally among the member links:

- When the parameters are configured using traffic control profiles, then the parameters scaled are the shaping rate, guaranteed rate, and delay buffer rate.
- When the parameters are configured using scheduler maps, then the parameters scaled are the transmit rate and buffer size. Shaping rate is also scaled if you configure it in bits per second (bps). Shaping rate is not scaled if you configure it as a percentage of the available interface bandwidth.

For example, consider an `ae` bundle between routers R1 and R2 consisting of three links. These are `ge-0/0/1`, `ge-0/0/2` and `ge-0/0/3` (`ae0`) on R1; and `ge-1/0/0`, `ge-1/0/1`, and `ge-1/0/2` (`ae2`) on R2. Two logical interfaces (units) are also configured on the `ae0` bundle on R1: `ae0.0` and `ae0.1`.

On `ae0`, traffic control profiles on R1 are configured as follows:

- `ae0` (the physical interface level) has a PIR of 450 Mbps.
- `ae0.0` (VLAN 100 at the logical interface level) has a PIR of 150 Mbps and a CIR of 90 Mbps.
- `ae0.1` (VLAN 200 at the logical interface level) has a PIR of 90 Mbps and a CIR of 60 Mbps.

In scaled mode, the `ae0` PIR is first divided among the member physical interfaces. Because there are three members, each receives 450 / 3 = 150 Mbps as a derived value. So the scaled PIR for the members interfaces is 150 Mbps each.

However, there are also two logical interfaces (`ae0.0` and `ae0.1`) and VLANs (100 and 200) on `ae0`. Traffic can leave on any of the three physical interfaces (`ge-0/0/1`, `ge-0/0/2`, or `ge-0/0/3`) in the bundle. Therefore, two derived logical interfaces are added to the member links to represent the two VLANs.

There are now six logical interfaces on the physical interfaces of the links making up the `ae` bundle, one set for VLAN 100 and the other for VLAN 200:

- `ge-0/0/1.0` and `ge-0/0/1.1`
- `ge-0/0/2.0` and `ge-0/0/2.1`
- `ge-0/0/3.0` and `ge-0/0/3.1`

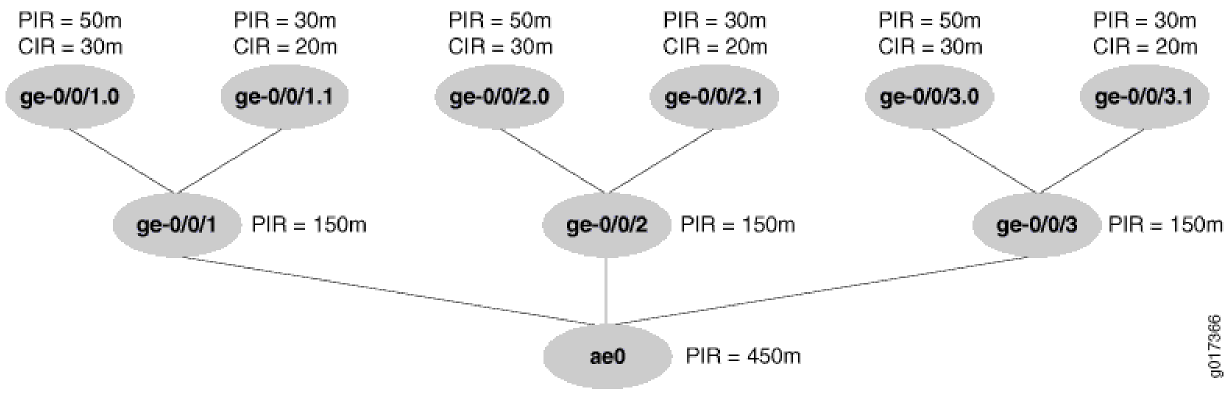
The traffic control profile parameters configured on `ae0.0` are divided across all the underlying logical interfaces (the unit 0s). In the same way, the traffic control profile parameters configured on `ae0.1` are divided across all the underlying logical interfaces (the unit 1s).

Therefore, the derived values of the scaled parameters on the interfaces are:

- For `ge-0/0/1.0` and `ge-0/0/2.0` and `ge-0/0/3.0`, each CIR = 90 / 3 = 30 Mbps, and each PIR = 150 / 3 = 50 Mbps.
- For `ge-0/0/1.1` and `ge-0/0/2.1` and `ge-0/0/3.1`, each CIR = 60 / 3 = 20 Mbps, and each PIR = 90 / 3 = 30 Mbps.

The scaled values are shown in Figure 74 on page 1051.

Figure 74: Scaled Mode for Aggregated Ethernet Interfaces



In scaled mode, when a new member link is added to the bundle, or an existing member link is either removed or fails, then the scaling factor (based on the number of active links) is recomputed and the new scheduler or traffic control profile parameters are reassigned. Only the PIR, CIR, and buffer parameters are recomputed: all other parameters are simply copied at each level.

NOTE: In `show class-of-service scheduler-map` commands, values derived in scaled mode instead of explicitly configured are flagged with `&**sf**`*n* suffix, where *n* indicates the value of the scaling factor.

The following sample shows the output for the scheduler map named `smap-all-abs` with and without a scaling factor:

```
user@host> show class-of-service scheduler-map
Scheduler map: smap-all-abs, Index: 65452

Scheduler: q0_sch_abs, Forwarding class: be, Index: 6775
Transmit rate: 40000000 bps, Rate Limit: none, Buffer size: remainder,
Priority: low
    Excess Priority: unspecified
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
      High            any             1    <default-drop-profile>

user@host> show class-of-service scheduler-map
Scheduler map: smap-all-abs, Index: 65452
```

```
Scheduler: q0_sch_abs&**sf**3, Forwarding class: be, Index: 2128
Transmit rate: 13333333 bps, Rate Limit: none, Buffer size: remainder,
Priority: low
    Excess Priority: unspecified
    Drop profiles:
      Loss priority   Protocol    Index    Name
      Low             any             1    <default-drop-profile>
      Medium low      any             1    <default-drop-profile>
      Medium high     any             1    <default-drop-profile>
      High            any             1    <default-drop-profile>
```

NOTE: There can be multiple scheduler maps created with different scaling factors, depending on when the child interfaces come up. For example, if there are only two active children on a parent interface, a new scheduler map with a scaling factor of 2 is created. The scheduler map name is `smap-all-abs&**sf**2`.

In replicated mode, in contrast to scaled mode, the configured scheduler parameters are simply replicated, not divided, among all member links of the `ae` bundle.

In replicated mode, the following scheduler parameter values are replicated among the member links and logical interfaces:

- When the parameters are configured using traffic control profiles, then the parameters replicated are the shaping rate, guaranteed rate, and delay buffer rate.

- When the parameters are configured using scheduler maps, then the parameters replicated are the transmit rate and buffer size.

If the scheduler parameters in the example configuration between routers R1 and R2 are applied with the `member-link-scheduler replicate` statement and option, the following parameters are applied:

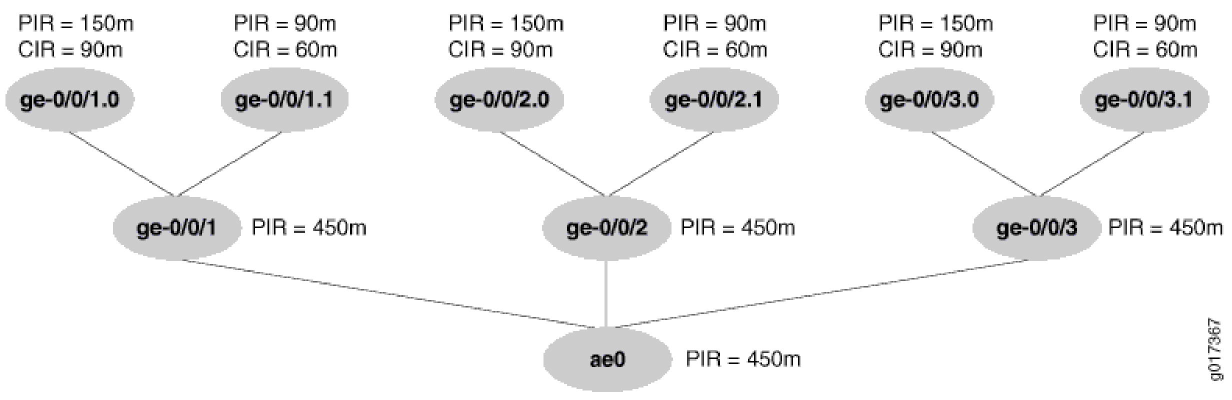
- The `ae0` PIR is copied among the member physical interfaces. Each receives 450 Mbps as a PIR.

- For each logical interface unit `.0`, the configured PIR and CIR for `ae0.0` is replicated (copied). Each logical interface unit `.0` receives a PIR of 150 Mbps and a CIR of 90 Mbps.

- For each logical interface unit `.1`, the configured PIR and CIR for `ae0.1` is replicated (copied). Each logical interface unit `.1` receives a PIR of 90 Mbps and a CIR of 60 Mbps.

The replicated values are shown in Figure 75 on page 1053.

Figure 75: Replicated Mode for Aggregated Ethernet Interfaces



In replicated mode, when a new member link is added to the bundle, or an existing member link is either removed or fails, the values are either copied or deleted from the required levels.

RELATED DOCUMENTATION

How Schedulers Define Output Queue Properties | 342

Default Schedulers Overview | 346

Configuring Schedulers | 348

Enabling VLAN Shaping and Scheduling on Aggregated Interfaces

Virtual LAN (VLAN) shaping (per-unit scheduling) is supported on aggregated Ethernet interfaces when link protection is enabled on the aggregated Ethernet interface. When VLAN shaping is configured on aggregate Ethernet interfaces with link protection enabled, the shaping is applied to the active child link.

To configure link protection on aggregated Ethernet interfaces, include the `link-protection` statement at the `[edit interfaces ae`*`x`*` aggregated-ether-options]` hierarchy level.

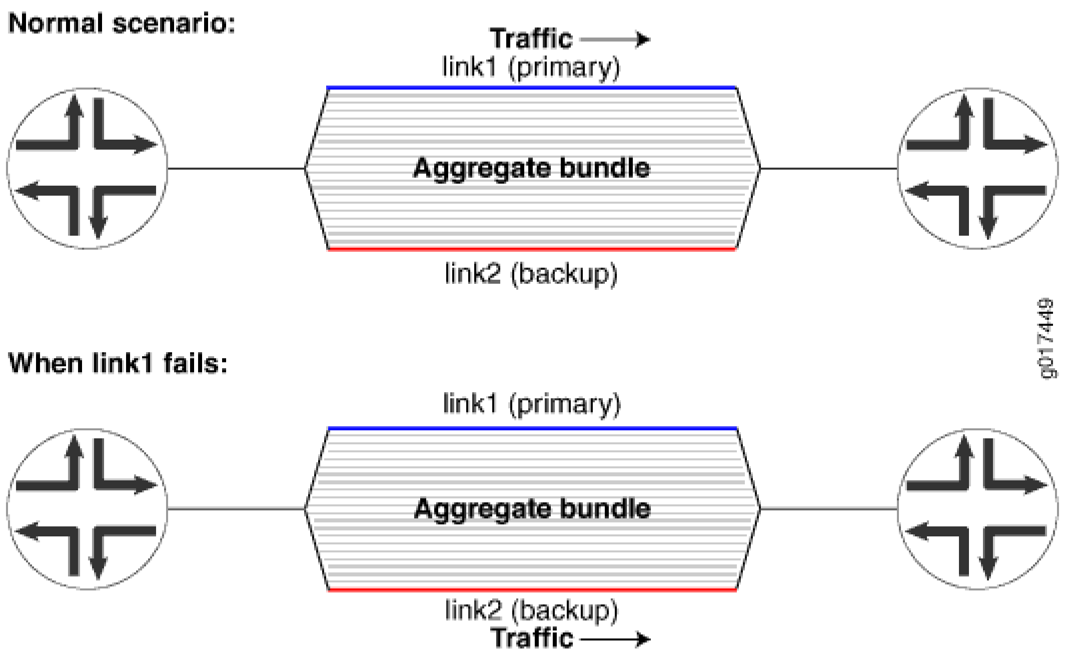
Traffic passes only through the designated primary link. This includes transit traffic and locally generated traffic on the router. When the primary link fails, traffic is routed through the backup link. You also can reverse traffic, from the designated backup link to the designated primary link. To revert back to sending traffic to the primary designated link when traffic is passing through the designated backup link, use the `revert` command. For example, `request interfaces revert ae0`.

To configure a primary and a backup link, include the `primary` and `backup` statements at the `[edit interfaces ge-`*`fpc/pic/port`*` gigether-options 802.3ad ae`*`x`*`]` hierarchy level or the `[edit interfaces xe-`*`fpc/pic/port`*` fastether-options 802.3ad ae`*`x`*`]` hierarchy level.

To disable link protection, delete the `link-protection` statement at `[edit interfaces aex aggregated-ether-options link-protection]` hierarchy level. To display the active, primary, and backup link for an aggregated Ethernet interface, use the operational mode command `show interfaces redundancy aex` .

Figure 76 on page 1054 shows how the flow of traffic changes from primary to backup when the primary link in an aggregate bundle fails.

Figure 76: Aggregated Ethernet Primary and Backup Links



This example configures two Gigabit Ethernet interfaces (`primary` and `backup`) as an aggregated Ethernet bundle (`ae0`) and enables link protection so that a shaping rate can be applied to VLANs on the aggregated Ethernet interface.

```
[edit class-of-service]
interface ae0 {
    shaping-rate 300m;
}
[edit interfaces]
ge-1/0/0 {
    gigether-options {
        802.3ad ae0 primary;
    }
}
ge-1/0/1 {
    gigether-options {
        802.3ad ae0 backup;
    }
}
ae0 {
    aggregated-ether-options {
        lacp {
            periodic slow;
        }
        link-protection {
            enable;
        }
    }
}
```

RELATED DOCUMENTATION

Applying Scheduler Maps and Shaping Rate to Logical Interfaces and VLANs | 394

Example: Applying Scheduling and Shaping to VLANs | 400

Class of Service on demux Interfaces

SUMMARY

This topic describes class of service (CoS) functionality and configuration on demux interfaces. demux interfaces support basic CoS functionality including behavior aggregate (BA) classifiers, rewrite rules, fixed classification, and port-level scheduling with targeted distribution.

IN THIS SECTION

Demultiplexing (demux) interfaces are logical interfaces that share a common, underlying interface. You can create logical subscriber interfaces using static or dynamic demultiplexing interfaces. In addition, you can use IP demultiplexing interfaces or VLAN demultiplexing interfaces when creating logical subscriber interfaces.

Overview of CoS on demux Interfaces

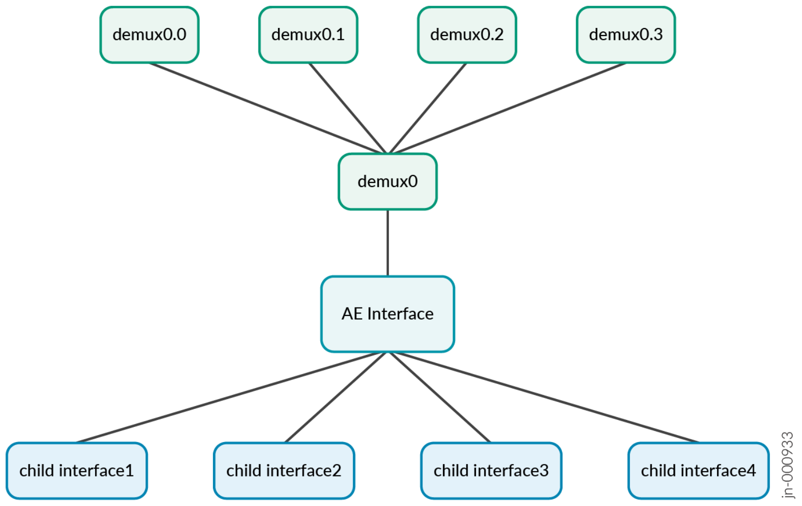
demux interfaces support basic CoS functionality, including:

- Classifiers including INET precedence, DSCP, IEEE 802.1p, and IEEE 802.1ad
- Rewrite rules including INET precedence, DSCP, IEEE 802.1p, and IEEE 802.1ad
- Fixed classification

NOTE: demux interfaces do not support MPLS EXP classifiers or rewrite rules.

Figure 77 on page 1056 shows the demux interface (`demux0`) with an underlying aggregated Ethernet (AE) interface. You can assign classifiers and rewrite rules to the logical interfaces (`demux0.x`) of the demux interface. You can then configure schedulers, scheduler maps, traffic control profiles, and so on, for the underlying AE interface.

Figure 77: demux interface with an underlying aggregated Ethernet interface



Configuring CoS on demux Interfaces

This section provides a simple example of how to configure a classifier and a rewrite rule for demux logical interfaces as well as a traffic control profile for the underlying AE interface.

1. Configure a classifier and apply it to a demux logical interface.

```
[edit class-of-service]
user@host# set classifiers ieee-802.1ad test forwarding-class expedited-forwarding loss-
priority low code-points 0000
user@host# set interfaces demux0 unit 0 classifiers ieee-802.1ad test
```

2. Configure a rewrite rule and apply it to a demux logical interface.

```
[edit class-of-service]
user@host# set rewrite-rules ieee-802.1ad test forwarding-class expedited-forwarding loss-
priority low code-point 1011
user@host# set interfaces demux0 unit 1 rewrite-rules ieee-802.1ad test
```

3. Configure a traffic control profile and apply it to the underlying AE interface.

```
[edit class-of-service]
user@host# set schedulers s1 shaping-rate 500m
user@host# set scheduler-maps testmap forwarding-class expedited-forwarding scheduler s1
user@host# set traffic-control-profiles tcp1 scheduler-map testmap
user@host# set interfaces ae0 output-traffic-control-profile tcp1
user@host# set interfaces ae0 member-link-scheduler replicate
```

4. Commit and show the configuration.

```
[edit class-of-service]
user@host# commit
user@host# show
classifiers {
    ieee-802.1ad test {
        forwarding-class expedited-forwarding {
            loss-priority low code-points 0000;
        }
    }
}
traffic-control-profiles {
    tcp1 {
        scheduler-map testmap;
    }
}
```

```
interfaces {
    ae0 {
        output-traffic-control-profile tcp1;
        member-link-scheduler replicate;
    }
    demux0 {
        unit 0 {
            classifiers {
                ieee-802.1ad test;
            }
        }
        unit 1 {
            rewrite-rules {
                ieee-802.1ad test;
            }
        }
    }
}
rewrite-rules {
    ieee-802.1ad test {
        forwarding-class expedited-forwarding {
            loss-priority low code-point 1011;
        }
    }
}
scheduler-maps {
    testmap {
        forwarding-class expedited-forwarding scheduler s1;
    }
}
schedulers {
    s1 {
        shaping-rate 500m;
    }
}
```

In the very simple example above, IEEE802.1ad traffic arriving on `demux0.0` with a code point of `0000` is assigned a forwarding class of `expedited-forwarding` and a loss priority of `low`. Through the traffic control profile, this traffic has up to 500 Mb of bandwidth as it traverses the router. Upon exiting the router through `demux0.1`, the traffic's code point is rewritten to `1011`.

Applying Layer 2 Policers to Gigabit Ethernet Interfaces

IN THIS SECTION

To rate-limit traffic by applying a policer to a Gigabit Ethernet interface (or a 10-Gigabit Ethernet interface [`xe-`*`fpc/pic/port`*]), include the `layer2-policer` statement with the direction, type, and name of the policer:

```
[edit interfaces ge-fpc/pic/port unit 0]
layer2-policer {
    input-policer policer-name;
    input-three-color policer-name;
    output-policer policer-name;
    output-three-color policer-name;
}
```

The direction (input or output) and type (policer or three-color) are combined into one statement and the policer named must be properly configured.

One input or output policer of either type can be configured on the interface.

Examples: Applying Layer 2 Policers to a Gigabit Ethernet Interface

Apply color-blind and color-aware two-rate TCM policers as input and output policers to a Gigabit Ethernet interface:

```
ge-1/0/0 {
    unit 0
    layer2-policer {
        input-three-color trTCM1-cb;    # Apply the trTCM1-color-blind policer.
        output-three-color trTCM1-ca;    # Apply the trTCM1-color-aware policer.
    }
}
```

Apply two-level and color-blind single-rate TCM policers as input and output policers to a Gigabit Ethernet interface:

```
ge-1/0/0 {
    unit 1
    layer2-policer {
        input-policer two-color-policer;    # Apply a two-color policer.
        output-three-color srTCM2-cb;    # Apply the srTCM1-color-blind policer.
    }
}
```

Apply a color-aware single-rate TCM policer as output policer on a Gigabit Ethernet interface:

```
ge-1/0/0 {
    unit 2
    layer2-policer {
        output-three-color srTCM3-ca {    # Apply the srTCM3-color-aware policer.
    }
}
```

RELATED DOCUMENTATION

Controlling Network Access Using Traffic Policing Overview

Overview of Tricolor Marking (TCM) Architecture | **245**

5

PART

Configuration Statements and Operational Commands

Junos CLI Reference Overview

We've consolidated all Junos CLI commands and configuration statements in one place. Read this guide to learn about the syntax and options that make up the statements and commands. Also understand the contexts in which you'll use these CLI elements in your network configurations and operations.

- Junos CLI Reference

Click the links to access Junos OS and Junos OS Evolved configuration statement and command summary topics.

- Configuration Statements
- Operational Commands