



Junos[®] OS

Gladiator Review



Modified: 2019-05-07



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

Juniper Networks, the Juniper Networks logo, Juniper, and Junos are registered trademarks of Juniper Networks, Inc. in the United States and other countries. All other trademarks, service marks, registered marks, or registered service marks are the property of their respective owners.

Juniper Networks assumes no responsibility for any inaccuracies in this document. Juniper Networks reserves the right to change, modify, transfer, or otherwise revise this publication without notice.

Junos® OS Gladiator Review

Copyright © 2019 Juniper Networks, Inc. All rights reserved.

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at <https://support.juniper.net/support/eula/>. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

	About the Documentation	xv
	Documentation and Release Notes	xv
	Using the Examples in This Manual	xv
	Merging a Full Example	xvi
	Merging a Snippet	xvi
	Documentation Conventions	xvii
	Documentation Feedback	xx
	Requesting Technical Support	xx
	Self-Help Online Tools and Resources	xxi
	Creating a Service Request with JTAC	xxi
Part 1	Overview	
Chapter 1	Understanding How Class of Service Manages Congestion and Defines Traffic Forwarding Behavior	3
	Understanding How Class of Service Manages Congestion and Controls Service	
	Levels in the Network	4
	CoS Applications	5
	CoS Standards	6
	How CoS Applies to Packet Flow Across a Network	6
	The Junos OS CoS Components Used to Manage Congestion and Control Service	
	Levels	7
	Mapping CoS Component Inputs to Outputs	11
	Default Junos OS CoS Settings	14
	Packet Flow Through the Junos OS CoS Process Overview	17
	Packet Flow Within Routers Overview	19
	Configuring Basic Packet Flow Through the Junos OS CoS Process	19
	Define Classifiers	20
	Apply Classifiers to Incoming Packets on Interfaces	21
	Define Policers to Limit Traffic and Control Congestion	22
	Define Drop Profiles	23
	Assign Each Forwarding Class to a Queue	23
	Define Schedulers	23
	Define Scheduler Maps	24
	Define CoS Header Rewrite Rules	24
	Apply Scheduler Maps and Rewrite Rules to Egress Interfaces	25
	Example: Classifying All Traffic from a Remote Device by Configuring Fixed	
	Interface-Based Classification	25
	Interface Types That Do Not Support Junos OS CoS	32

Part 2	Configuring Class of Service	
Chapter 2	Assigning Service Levels to Trusted Packets Using Behavior Aggregate Classifiers	37
	Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic . . .	38
	Default Aliases for CoS Value Bit Patterns Overview	42
	Defining Aliases for CoS Value Bit Patterns	46
	Applying Behavior Aggregate Classifiers to Logical Interfaces	49
	Configuring Behavior Aggregate Classifiers	53
	Default DSCP and DSCP IPv6 Classifiers	55
	Applying DSCP Classifiers to MPLS Traffic	56
	Applying a DSCP Classifier to MPLS Packets on a Core-facing Interface	57
	Applying a DSCP Classifier to MPLS Traffic for L3VPN/VPLS	59
	Example: Configuring and Applying a Default DSCP Behavior Aggregate Classifier	61
	Understanding DSCP Classification for VPLS	69
	Example: Configuring DSCP Classification for VPLS	70
	Example: Configuring Behavior Aggregate Classifiers	73
	Default MPLS EXP Classifier	83
	Applying MPLS EXP Classifiers to Routing Instances	84
	Configuring and Applying Custom MPLS EXP Classifiers to Routing Instances	85
	Applying Global Classifiers and Wildcard Routing Instances	86
	Applying Global MPLS EXP Classifiers to Routing Instances	87
	Applying Classifiers by Using Wildcard Routing Instances	89
	Verifying the Classifiers Associated with Routing Instances	90
	Applying MPLS EXP Classifiers for Explicit-Null Labels	91
	Default IEEE 802.1p Classifier	92
	Default IEEE 802.1ad Classifier	93
	Default IP Precedence Classifier	94
Chapter 3	Assigning Service Levels to Untrusted Packets by Using Multifield Classifiers	97
	Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields	97
	Configuring Multifield Classifiers	98
	Using Multifield Classifiers to Set Packet Loss Priority	101
	Example: Configuring and Applying a Firewall Filter for a Multifield Classifier . . .	103
	Example: Classifying Packets Based on Their Destination Address	109
	Example: Configuring and Verifying a Complex Multifield Filter	112
	Configuring a Complex Multifield Filter	112
	Verifying a Complex Multifield Filter	114
Chapter 4	Controlling Access to the Network Using Traffic Policing	117
	Controlling Network Access Using Traffic Policing Overview	117
	Congestion Management for IP Traffic Flows	117
	Traffic Limits	118
	Traffic Color Marking	119
	Forwarding Classes and PLP Levels	121

Policer Application to Traffic	121
Enabling Tricolor Marking and Limitations of Three-Color Policers	123
Overview of Tricolor Marking Architecture	125
Configuring Single-Rate Tricolor Marking	126
Configuring Color-Blind Mode for Single-Rate Tricolor Marking	126
Configuring Color-Aware Mode for Single-Rate Tricolor Marking	127
Effect on Low PLP of Single-Rate Policer	128
Effect on Medium-Low PLP of Single-Rate Policer	128
Effect on Medium-High PLP of Single-Rate Policer	129
Effect on High PLP of Single-Rate Policer	129
Example: Limiting Inbound Traffic at Your Network Border by Configuring an Ingress Single-Rate Two-Color Policer	129
Example: Performing CoS at an Egress Network Boundary by Configuring an Egress Single-Rate Two-Color Policer	138
Configuring Two-Rate Tricolor Marking	148
Configuring Color-Blind Mode for Two-Rate Tricolor Marking	148
Configuring Color-Aware Mode for Two-Rate Tricolor Marking	149
Effect on Low PLP of Two-Rate Policer	149
Effect on Medium-Low PLP of Two-Rate Policer	150
Effect on Medium-High PLP of Two-Rate Policer	150
Effect on High PLP of Two-Rate Policer	150
Example: Configuring and Verifying Two-Rate Tricolor Marking	151
Configuring and Applying Tricolor Marking Policers	159
Defining a Tricolor Marking Policer	159
Applying Tricolor Marking Policers to Firewall Filters	161
Applying Firewall Filter Tricolor Marking Policers to Interfaces	162
Example: Configuring and Applying a Single-Rate Tricolor Marking Policer	163
Example: Limiting Inbound Traffic Within Your Network by Configuring an Ingress Single-Rate Two-Color Policer and Configuring Multifield Classifiers	164
Example: Limiting Outbound Traffic Within Your Network by Configuring an Egress Single-Rate Two-Color Policer and Configuring Multifield Classifiers	178
Configuring Policers Based on Logical Interface Bandwidth	194
Effect of Two-Color Policers on Shaping Rate Changes	197
Chapter 5	
Defining Forwarding Behavior Based on Forwarding Classes	199
Understanding How Forwarding Classes Assign Classes to Output Queues	199
Output Queue Assignments Based on Forwarding Class	200
Devices That Support Up to Four Forwarding Classes	200
Devices That Support Up to 16 Forwarding Classes	201
Default and Configurable Packet Loss Priority Values	201
Configuration Statements Used to Configure and Apply Forwarding Classes	201
Default Forwarding Classes	202
Configuring a Custom Forwarding Class for Each Queue	206
Configuring Up to 16 Custom Forwarding Classes	208
Enabling Eight Queues on Interfaces	211
Assigning Multiple Forwarding Classes and Default Forwarding Classes	212

	Examples: Configuring Up to 16 Forwarding Classes	213
	Classifying Packets by Egress Interface	215
	Forwarding Policy Options Overview	217
	Configuring CoS-Based Forwarding	219
	Example: Configuring CoS-Based Forwarding	222
	Example: Configuring CoS-Based Forwarding for Different Traffic Types	224
	Example: Configuring CoS-Based Forwarding for IPv6	225
	Applying Forwarding Classes to Interfaces	226
	Understanding Queuing and Marking of Host Outbound Traffic	227
	Host Outbound Traffic Overview	227
	Routing Engine Sourced Traffic	227
	Distributed Protocol Handler Traffic	227
	Default Queuing and Marking of Host Outbound Traffic	228
	Configured Queuing and Marking of Host Outbound Traffic	228
	Configured Queuing and Marking of Outbound Routing Engine Traffic Only	228
	Forwarding Classes and Fabric Priority Queues	229
	Default Fabric Priority Queuing	229
	Overriding Default Fabric Priority Queuing	229
	Default Routing Engine Protocol Queue Assignments	229
	Assigning Forwarding Class and DSCP Value for Routing Engine-Generated Traffic	232
	Example: Writing Different DSCP and EXP Values in MPLS-Tagged IP Packets	233
	Changing the Default Queuing and Marking of Host Outbound Traffic	236
	Example: Configuring Different Queuing and Marking Defaults for Outbound Routing Engine and Distributed Protocol Handler Traffic	236
	Overriding the Input Classification	245
Chapter 6	Defining Output Queue Properties Using Schedulers	247
	How Schedulers Define Output Queue Properties	248
	Queue Scheduling Components	249
	Default Schedulers Overview	251
	Configuring Schedulers	252
	Configuring Scheduler Maps	252
	Applying Scheduler Maps Overview	253
	Applying Scheduler Maps to Physical Interfaces	254
	Understanding Interface Sets	254
	Configuring Interface Sets	255
	Interface Set Caveats	257
	Configuring Internal Scheduler Nodes	259
	Example: Configuring and Applying Scheduler Maps	260
Chapter 7	Controlling Bandwidth Using Scheduler Rates	263
	Oversubscribing Interface Bandwidth	263
	Verifying Configuration of Bandwidth Oversubscription	269
	Examples: Oversubscribing Interface Bandwidth	269
	Configuring Scheduler Transmission Rate	272
	Example: Configuring Scheduler Transmission Rate	274
	Allocation of Leftover Bandwidth	274

	Providing a Guaranteed Minimum Rate	275
	Verifying Configuration of Guaranteed Minimum Rate	279
	Example: Providing a Guaranteed Minimum Rate	279
	PIR-Only and CIR Mode	280
	Excess Rate and Excess Priority Configuration Examples	280
	Controlling Remaining Traffic	285
	Bandwidth Sharing on Nonqueueing Packet Forwarding Engines Overview	288
	Configuring Rate Limits on Nonqueueing Packet Forwarding Engines	289
	Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs	290
	Example: Applying Scheduler Maps and Shaping Rate to DLCIs	299
	Example: Applying Scheduling and Shaping to VLANs	303
	Example: Limiting Egress Traffic on an Interface Using Port Shaping for CoS	311
Chapter 8	Setting Transmission Order Using Scheduler Priorities and Hierarchical Scheduling	319
	Priority Scheduling Overview	319
	Strict-High Priority Configuration Overview	320
	Platform Support for Priority Scheduling	321
	Configuring Schedulers for Priority Scheduling	322
	Associating Schedulers with Fabric Priorities	323
	Example: Associating a Scheduler with a Fabric Priority	324
	Hierarchical Class of Service Overview	325
	Hierarchical Class of Service Network Scenarios	328
	Services to Subscribers	328
	Services to Businesses	329
	Wireless Backhaul	329
	Understanding Hierarchical Scheduling	329
	Hierarchical Scheduling Terminology	330
	Scheduler Node-Level Designations in Hierarchical Scheduling	330
	Hierarchical Scheduling at Non-Leaf Nodes	331
	Priority Propagation in Hierarchical Scheduling	332
	Configuring Hierarchical Schedulers for CoS	334
	Hierarchical Schedulers and Traffic Control Profiles	335
	Example: Building a Four-Level Hierarchy of Schedulers	337
	Configuring the Interface Sets	338
	Configuring the Interfaces	338
	Configuring the Traffic Control Profiles	339
	Configuring the Schedulers	340
	Configuring the Drop Profiles	340
	Configuring the Scheduler Maps	341
	Applying the Traffic Control Profiles	341
Chapter 9	Controlling Congestion Using Scheduler RED Drop Profiles and Buffers	343
	Managing Congestion Using RED Drop Profiles and Packet Loss Priorities	343
	Defining Packet Drop Behavior by Configuring RED Drop Profiles	347
	Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers	351

	Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows	353
	Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size	355
	Configuring Large Delay Buffers for Slower Interfaces	357
	Configuring the Maximum Delay Buffer for NxDSO Interfaces	361
	Example: Configuring Large Delay Buffers for Slower Interfaces	364
	Example: Configuring the Delay Buffer Value for a Scheduler	365
	Example: Configuring the Physical Interface Shaping Rate	367
	Complete Configuration	367
	Enabling and Disabling the Memory Allocation Dynamic per Queue	369
	Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy	371
	Example: Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy	373
Chapter 10	Altering Outgoing Packet Headers Using Rewrite Rules to Ensure Forwarding Behavior	377
	Rewriting Packet Headers to Ensure Forwarding Behavior	378
	Applying Default Rewrite Rules	379
	Configuring Rewrite Rules	381
	Configuring Rewrite Rules Based on PLP	383
	Applying IEEE 802.1p Rewrite Rules to Dual VLAN Tags	383
	Example: Applying an IEEE 802.1p Rewrite Rule to Dual VLAN Tags	385
	Applying IEEE 802.1ad Rewrite Rules to Dual VLAN Tags	385
	Example: Applying an IEEE 802.1ad Rewrite Rule to Dual VLAN Tags	386
	Rewriting IEEE 802.1p Packet Headers with an MPLS EXP Value	386
	Setting IPv6 DSCP and MPLS EXP Values Independently	388
	Configuring DSCP Values for IPv6 Packets Entering the MPLS Tunnel	388
	Setting Ingress DSCP Bits for Multicast Traffic over Layer 3 VPNs	390
	Applying Rewrite Rules to Output Logical Interfaces	391
	Rewriting MPLS and IPv4 Packet Headers	393
	Example: Rewriting MPLS and IPv4 Packet Headers	395
	Example: Simultaneous DSCP and EXP Rewrite	397
	Defining a Custom Frame Relay Loss Priority Map	398
	Example: Per-Node Rewriting of EXP Bits	399
	Example: Rewriting CoS Information at the Network Border to Enforce CoS Strategies	400
	Example: Remarking Diffserv Code Points to MPLS EXPs to Carry CoS Profiles Across a Service Provider's L3VPN MPLS Network	410
	Example: Remarking Diffserv Code Points to 802.1P PCPs to Carry CoS Profiles Across a Service Provider's VPLS Network	436
Chapter 11	Altering Class of Service Values in Packets Exiting the Network Using IPv6 DiffServ	461
	Resources for CoS with DiffServ for IPv6	461
	System Requirements for CoS with DiffServ for IPv6	462
	Terms and Acronyms for CoS with DiffServ for IPv6	462
	Default DSCP Mappings	463
	Default Forwarding Classes	464

Juniper Networks Default Forwarding Classes	467
Roadmap for Configuring CoS with IPv6 DiffServ	469
Configuring a Firewall Filter for an MF Classifier on Customer Interfaces	470
Applying the Firewall Filter to Customer Interfaces	471
Assigning Forwarding Classes to Output Queues	471
Configuring Rewrite Rules	472
DSCP IPv6 Rewrites and Forwarding Class Maps	473
Applying Rewrite Rules to an Interface	473
Configuring RED Drop Profiles	474
Configuring BA Classifiers	474
Applying a BA Classifier to an Interface	475
Configuring a Scheduler	476
Configuring Scheduler Maps	476
Applying a Scheduler Map to an Interface	477
Example: Configuring DiffServ for IPv6	477
Configuration	477
Verification	488

List of Figures

Part 1	Overview
Chapter 1	Understanding How Class of Service Manages Congestion and Defines Traffic Forwarding Behavior 3
	Figure 1: Packet Flow Across the Network 7
	Figure 2: Packet Flow Through CoS-Configurable Components 8
	Figure 3: Packet Flow Through CoS-Configurable Components 11
	Figure 4: CoS Classifier, Queues, and Scheduler 18
	Figure 5: Packet Flow Through CoS- Configurable Components 18
	Figure 6: CoS Classifier, Queues, and Scheduler 19
	Figure 7: Packet Flow Through CoS- Configurable Components 20
	Figure 8: Fixed-Interface Classifier Scenario 27
Part 2	Configuring Class of Service
Chapter 2	Assigning Service Levels to Trusted Packets Using Behavior Aggregate Classifiers 37
	Figure 9: How a Classifier Works 39
	Figure 10: Behavior Aggregate Classifier with Two Queues 64
	Figure 11: Behavior Aggregate Classifier Scenario 65
	Figure 12: Behavior Aggregate Classifier Scenario 75
Chapter 3	Assigning Service Levels to Untrusted Packets by Using Multifield Classifiers 97
	Figure 13: How a Classifier Works 98
	Figure 14: Multifield Classifier Based on TCP Source Ports 104
	Figure 15: Multifield Classifier Scenario 104
Chapter 4	Controlling Access to the Network Using Traffic Policing 117
	Figure 16: Network Traffic and Burst Rates 119
	Figure 17: Flow of Tricolor Marking Policer Operation 125
	Figure 18: Single-Rate Two-Color Policer Scenario 132
	Figure 19: Traffic Limiting in a Single-Rate Two-Color Policer Scenario 132
	Figure 20: Single-Rate Two-Color Policer Scenario 141
	Figure 21: Traffic Limiting in a Single-Rate Two-Color Policer Scenario 141
	Figure 22: Tricolor Marking Sample Topology 151
	Figure 23: Single-Rate Two-Color Policer Scenario 167
	Figure 24: Traffic Limiting in a Single-Rate Two-Color Policer Scenario 167
	Figure 25: Multifield Classifier Based on TCP Source Ports 168
	Figure 26: Single-Rate Two-Color Policer Scenario 181
	Figure 27: Traffic Limiting in a Single-Rate Two-Color Policer Scenario 181
	Figure 28: Multifield Classifier Based on TCP Source Ports 182

Chapter 5	Defining Forwarding Behavior Based on Forwarding Classes	199
	Figure 29: Customer-Facing and Core-Facing Forwarding Classes	209
	Figure 30: Sample CoS-Based Forwarding	222
Chapter 7	Controlling Bandwidth Using Scheduler Rates	263
	Figure 31: Handling Remaining Traffic	286
	Figure 32: Another Example of Handling Remaining Traffic	287
	Figure 33: Port Shaping Scenario	312
Chapter 8	Setting Transmission Order Using Scheduler Priorities and Hierarchical Scheduling	319
	Figure 34: Hierarchical Scheduling Architecture	326
	Figure 35: Hierarchical Schedulers and Priorities	334
	Figure 36: Building a Scheduler Hierarchy	337
Chapter 9	Controlling Congestion Using Scheduler RED Drop Profiles and Buffers	343
	Figure 37: Discrete and Interpolated Drop Profiles	344
	Figure 38: Discrete and Interpolated Drop Profiles	348
Chapter 10	Altering Outgoing Packet Headers Using Rewrite Rules to Ensure Forwarding Behavior	377
	Figure 39: Packet Flow Across the Network	378
	Figure 40: Rewriting CoS Information at the Network Border to Enforce CoS Strategies Scenario	401
	Figure 41: MPLS Packet Structure	412
	Figure 42: Rewriting CoS Information at the Network Border to Transit an MPLS Network Scenario	413
	Figure 43: Typical VPLS Topology	437
	Figure 44: VPLS with CoS Scenario	439
Chapter 11	Altering Class of Service Values in Packets Exiting the Network Using IPv6 DiffServ	461
	Figure 45: Basic IPv6 DiffServ Topology	478
	Figure 46: IPv6 DiffServ Configuration	478

List of Tables

	About the Documentation	xv
	Table 1: Notice Icons	xviii
	Table 2: Text and Syntax Conventions	xviii
Part 1	Overview	
Chapter 1	Understanding How Class of Service Manages Congestion and Defines Traffic Forwarding Behavior	3
	Table 3: CoS Mappings—Inputs and Outputs	11
Part 2	Configuring Class of Service	
Chapter 2	Assigning Service Levels to Trusted Packets Using Behavior Aggregate Classifiers	37
	Table 4: Default CoS Value Aliases	43
	Table 5: Logical Interface Classifier Combinations	49
	Table 6: Default DSCP and DSCP IPv6 Classifiers	56
	Table 7: Default VPLS Classifiers	70
	Table 8: Sample ba-classifier Loss Priority Assignments	74
	Table 9: Default MPLS EXP Classification	84
	Table 10: Default MPLS EXP Classifier	84
	Table 11: Default IEEE 802.1p Classifier	93
	Table 12: Default IEEE 802.1ad Classifier	93
	Table 13: Default IP Precedence (ipprec-compatibility) Classifier	94
	Table 14: Default IP Precedence (ipprec-default) Classifier	95
Chapter 4	Controlling Access to the Network Using Traffic Policing	117
	Table 15: Policer Actions	120
	Table 16: Devices Versus TCM	124
	Table 17: Color-Blind Mode TCM Color-to-PLP Mapping	127
	Table 18: Color-Aware Mode TCM PLP Mapping	127
	Table 19: Color-Blind Mode TCM Color-to-PLP Mapping	148
	Table 20: Color-Aware Mode TCM Mapping	149
	Table 21: Tricolor Marking Policer Statements	159
Chapter 5	Defining Forwarding Behavior Based on Forwarding Classes	199
	Table 22: Default Forwarding Classes	202
	Table 23: Sample Forwarding Class-to-Queue Mapping	209
	Table 24: Default Queue Assignments for Packets Generated by the Routing Engine	229
Chapter 6	Defining Output Queue Properties Using Schedulers	247

	Table 25: Output Queue Scheduler Components	249
	Table 26: Sample diffserv-cos-map Scheduler Mapping	260
Chapter 7	Controlling Bandwidth Using Scheduler Rates	263
	Table 27: Bandwidth and Delay Buffer Allocations by Configuration Scenario . .	267
	Table 28: Bandwidth and Delay Buffer Allocations by Configuration Scenario . .	278
	Table 29: Current Behavior with Multiple Priority Levels	281
	Table 30: Current Behavior with Same Priority Levels	281
	Table 31: Current Behavior with Strict-High Priority	281
	Table 32: Strict-High Priority with Higher Load	282
	Table 33: Sharing with Multiple Priority Levels	282
	Table 34: Sharing with the Same Priority Levels	283
	Table 35: Sharing with at Least One Strict-High Priority	283
	Table 36: Sharing with at Least One Strict-High Priority and Higher Load	283
	Table 37: Sharing with at Least One Strict-High Priority and Rate Limit	284
	Table 38: Fine-Grained Queuing and Scheduling Support by Interface or PIC Type	293
	Table 39: Fine-Grained Queuing and Scheduling Support by MIC or MPC Type	295
Chapter 8	Setting Transmission Order Using Scheduler Priorities and Hierarchical Scheduling	319
	Table 40: Scheduling Priority Mappings by FPC Type	322
	Table 41: Resources for Learning More About HCoS	327
	Table 42: Node Levels Designations in Hierarchical Scheduling	331
	Table 43: Queue Priority	332
	Table 44: Internal Node Queue Priority for CIR Mode	333
	Table 45: Internal Node Queue Priority for PIR-Only Mode	333
Chapter 9	Controlling Congestion Using Scheduler RED Drop Profiles and Buffers	343
	Table 46: Buffer Size Temporal Value Ranges by Routing Device Type	356
	Table 47: Recommended Delay Buffer Sizes	357
	Table 48: Maximum Delay Buffer with q-pic-large-buffer Enabled by Interface	358
	Table 49: Delay-Buffer Calculations	359
	Table 50: NxDSO Transmission Rates and Delay Buffers	362
Chapter 10	Altering Outgoing Packet Headers Using Rewrite Rules to Ensure Forwarding Behavior	377
	Table 51: Default Packet Header Rewrite Mappings	380
	Table 52: Default MPLS EXP Rewrite Rules	394
Chapter 11	Altering Class of Service Values in Packets Exiting the Network Using IPv6 DiffServ	461
	Table 53: Default DSCP Mappings	463
	Table 54: Default Behavior Aggregate Classification	465
	Table 55: Forwarding Classes and Queues Classification	466
	Table 56: Resources Assigned to Queues	467
	Table 57: Default Forwarding Classes	468

About the Documentation

- Documentation and Release Notes on page xv
- Using the Examples in This Manual on page xv
- Documentation Conventions on page xvii
- Documentation Feedback on page xx
- Requesting Technical Support on page xx

Documentation and Release Notes

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

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

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

Using the Examples in This Manual

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

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

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

Merging a Full Example

To merge a full example, follow these steps:

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

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

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

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

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

Merging a Snippet

To merge a snippet, follow these steps:

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

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

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


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

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

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

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

For more information about the **load** command, see [CLI Explorer](#).

Documentation Conventions

[Table 1 on page xviii](#) defines notice icons used in this guide.

Table 1: Notice Icons

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

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

Table 2: Text and Syntax Conventions

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

Table 2: Text and Syntax Conventions (continued)

Convention	Description	Examples
<i>Italic text like this</i>	<ul style="list-style-type: none">Introduces or emphasizes important new terms.Identifies guide names.Identifies RFC and Internet draft titles.	<ul style="list-style-type: none">A policy <i>term</i> is a named structure that defines match conditions and actions.<i>Junos OS CLI User Guide</i>RFC 1997, <i>BGP Communities Attribute</i>
<i>Italic text like this</i>	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: <div><pre>[edit] root@# set system domain-name domain-name</pre></div>
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	<ul style="list-style-type: none">To configure a stub area, include the stub statement at the [edit protocols ospf area area-id] hierarchy level.The console port is labeled CONSOLE.
< > (angle brackets)	Encloses optional keywords or variables.	stub <default-metric <i>metric</i> >;
(pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	broadcast multicast (string1 string2 string3)
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	rsvp { # Required for dynamic MPLS only
[] (square brackets)	Encloses a variable for which you can substitute one or more values.	community name members [community-ids]
Indentation and braces ({ })	Identifies a level in the configuration hierarchy.	<div><pre>[edit] routing-options { static { route default { nexthop address; retain; } } }</pre></div>
;(semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
GUI Conventions		
Bold text like this	Represents graphical user interface (GUI) items you click or select.	<ul style="list-style-type: none">In the Logical Interfaces box, select All Interfaces.To cancel the configuration, click Cancel.

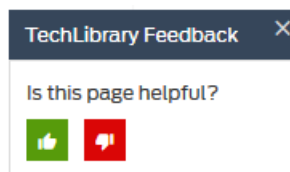
Table 2: Text and Syntax Conventions (continued)

Convention	Description	Examples
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select Protocols>Ospf .

Documentation Feedback

We encourage you to provide feedback so that we can improve our documentation. You can use either of the following methods:

- Online feedback system—Click TechLibrary Feedback, on the lower right of any page on the [Juniper Networks TechLibrary](#) site, and do one of the following:



- Click the thumbs-up icon if the information on the page was helpful to you.
- Click the thumbs-down icon if the information on the page was not helpful to you or if you have suggestions for improvement, and use the pop-up form to provide feedback.
- E-mail—Send your comments to techpubs-comments@juniper.net. Include the document or topic name, URL or page number, and software version (if applicable).

Requesting Technical Support

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

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

Self-Help Online Tools and Resources

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

- Find CSC offerings: <https://www.juniper.net/customers/support/>
- Search for known bugs: <https://prsearch.juniper.net/>
- Find product documentation: <https://www.juniper.net/documentation/>
- Find solutions and answer questions using our Knowledge Base: <https://kb.juniper.net/>
- Download the latest versions of software and review release notes: <https://www.juniper.net/customers/csc/software/>
- Search technical bulletins for relevant hardware and software notifications: <https://kb.juniper.net/InfoCenter/>
- Join and participate in the Juniper Networks Community Forum: <https://www.juniper.net/company/communities/>
- Create a service request online: <https://myjuniper.juniper.net>

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

Creating a Service Request with JTAC

You can create a service request with JTAC on the Web or by telephone.

- Visit <https://myjuniper.juniper.net>.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see <https://support.juniper.net/support/requesting-support/>.

PART 1

Overview

- [Understanding How Class of Service Manages Congestion and Defines Traffic Forwarding Behavior on page 3](#)

CHAPTER 1

Understanding How Class of Service Manages Congestion and Defines Traffic Forwarding Behavior

- [Understanding How Class of Service Manages Congestion and Controls Service Levels in the Network on page 4](#)
- [How CoS Applies to Packet Flow Across a Network on page 6](#)
- [The Junos OS CoS Components Used to Manage Congestion and Control Service Levels on page 7](#)
- [Mapping CoS Component Inputs to Outputs on page 11](#)
- [Default Junos OS CoS Settings on page 14](#)
- [Packet Flow Through the Junos OS CoS Process Overview on page 17](#)
- [Configuring Basic Packet Flow Through the Junos OS CoS Process on page 19](#)
- [Example: Classifying All Traffic from a Remote Device by Configuring Fixed Interface-Based Classification on page 25](#)
- [Interface Types That Do Not Support Junos OS CoS on page 32](#)

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

Usually, IP routers 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 system (Junos OS) 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 OS 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 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.

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 OS 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 routing 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 OS 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 OS class of service (CoS) capabilities are defined in the following RFCs:

- RFC 2474, *Definition of the Differentiated Services Field in the IPv4 and IPv6 Headers*
- RFC 2597, *Assured Forwarding PHB Group*
- RFC 2598, *An Expedited Forwarding PHB*
- RFC 2698, *A Two Rate Three Color Marker*

**Related
Documentation**

- [The Junos OS CoS Components Used to Manage Congestion and Control Service Levels on page 7](#)

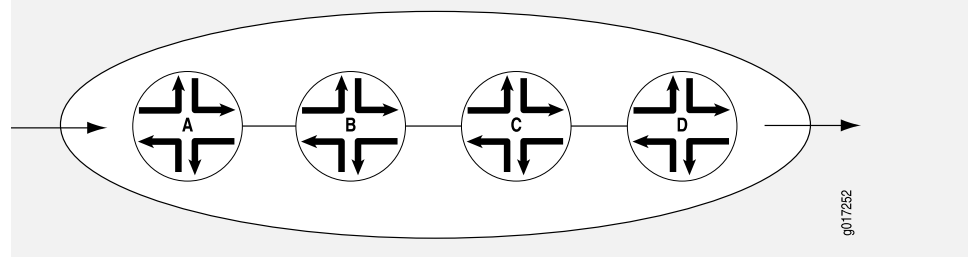
How CoS Applies to Packet Flow Across a Network

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 routing device in the network. Generally, each routing 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 routing device. In addition, the routing 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 7](#), 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**

- [The Junos OS CoS Components Used to Manage Congestion and Control Service Levels on page 7](#)

The Junos OS CoS Components Used to Manage Congestion and Control Service Levels

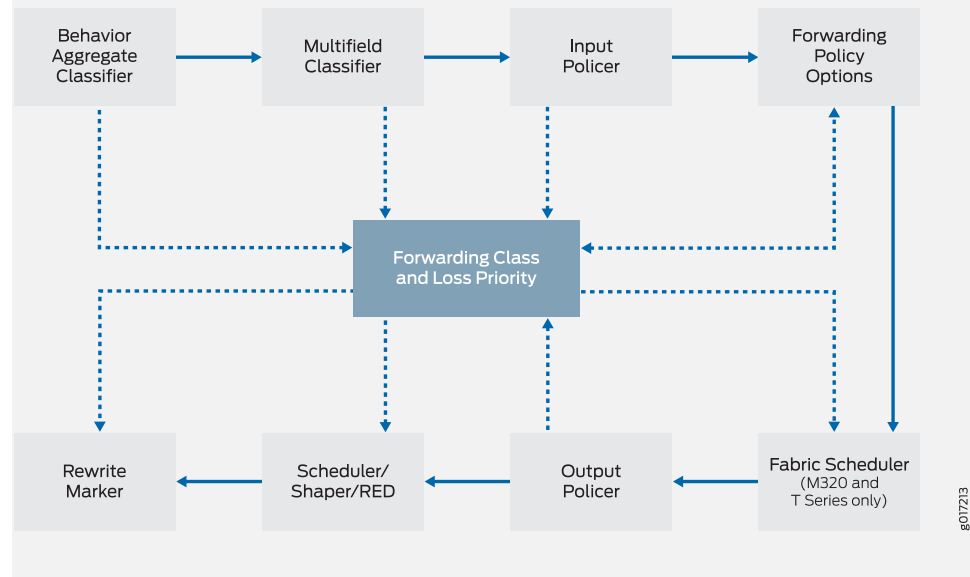
Any CoS implementation must work consistently end to end through the network. A standards-based, vendor-neutral CoS implementation satisfies this requirement best. Junos OS CoS features interoperate with other vendors' CoS implementations because they are based on IETF Differentiated Services (DiffServ) standards. Junos OS 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 8](#) shows the components of the Junos OS 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 OS 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 OS, 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 38](#) 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 97](#) 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. For most Juniper Networks M Series Multiservice Edge Routers, four forwarding classes are supported. You can configure up to one each of the four types of forwarding classes. For M120 and M320 Multiservice Edge Routers, Juniper Networks MX Series 5G Universal Routing Platforms, Juniper Networks T Series Core Routers, and EX Series switches, 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 199](#) 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 353](#) 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 217](#) 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 routing device is determined by the availability of the outgoing physical media as well as the amount of traffic using the interface.
 - Schedulers—An individual routing device interface has multiple queues assigned to store packets. The routing 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. The Junos OS 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 248](#) for more information on schedulers.

- Fabric schedulers—For M120, M320, and T Series routers only, fabric schedulers allow you to identify a packet as high or low priority based on its forwarding class, and to associate schedulers with the fabric priorities.
- 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”](#) on page 117 for more information on policers.

- Rewrite rules—A *rewrite rule* sets the appropriate CoS bits in the outgoing packet. This allows the next downstream routing 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 378 for more information on rewrite rules.

**Related
Documentation**

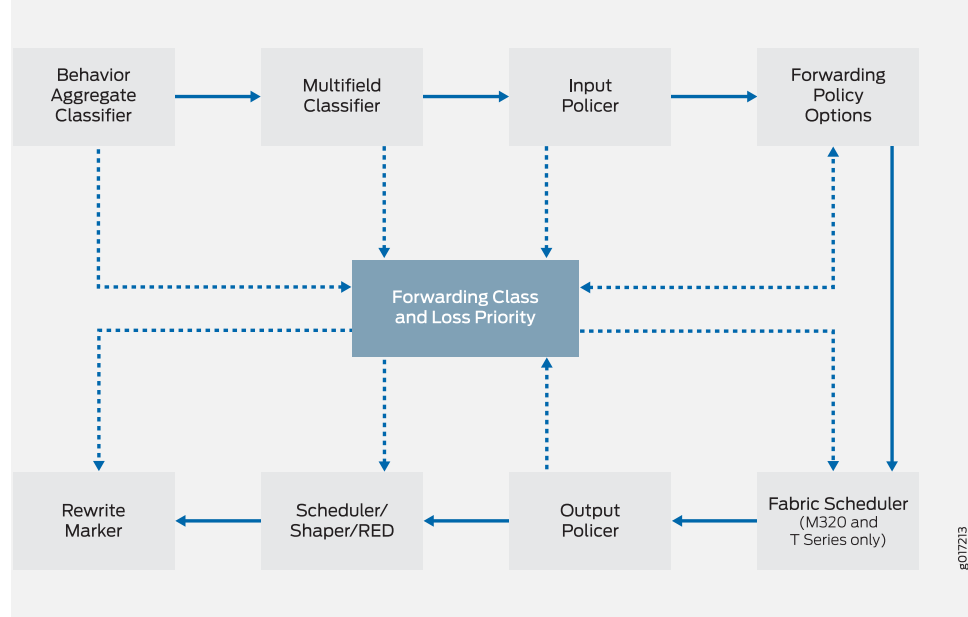
- [Understanding How Class of Service Manages Congestion and Controls Service Levels in the Network](#) on page 4

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 2 on page 8 shows the components of the Junos OS 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 3 on page 11.

Table 3: 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 3: 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 53](#) 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 351](#) 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 252](#) 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 381](#) 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

- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)
- [Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers on page 351](#)
- [Configuring Scheduler Maps on page 252](#)
- [Applying Default Rewrite Rules on page 379](#)

Default Junos OS 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 Frame Relay Loss Priority Map

```

Loss-priority-map: frame-relay-de-default, Code point type: frame-relay-de, Index:
13
  Code point      Loss priority
  0               low
  1               high

```

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 on page 202](#)
- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)
- [Managing Congestion Using RED Drop Profiles and Packet Loss Priorities on page 343](#)
- [Default Schedulers Overview on page 251](#)
- [Forwarding Classes and Fabric Priority Queues on page 229](#)

Packet Flow Through the Junos OS CoS Process Overview

Perhaps the best way to understand Junos OS 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.



NOTE: You can only rewrite the values of these bits on ingress on the Juniper Networks M40e, M120, M320 Multiservice Edge Routers, and T Series Core Routers with IQE PICs. For more information about rewriting the values of these bits on ingress, see *Configuring ToS Translation Tables*.

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 18 show the components of the Junos OS CoS features, illustrating the sequence in which they interact.

Figure 4: CoS Classifier, Queues, and Scheduler

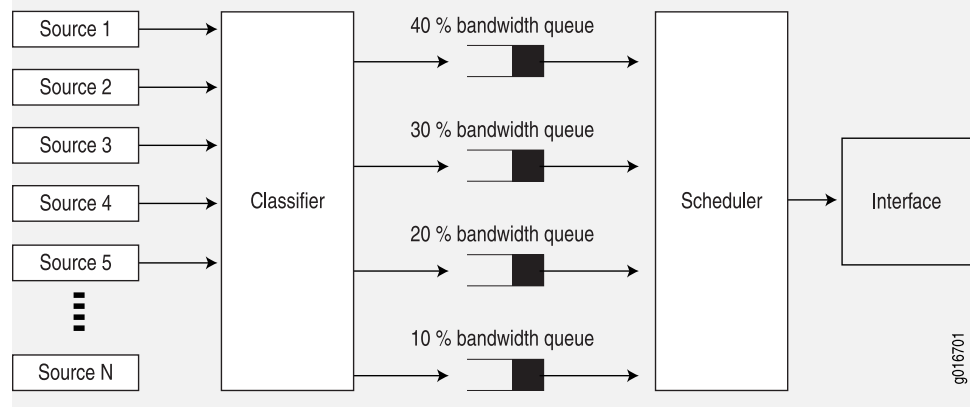
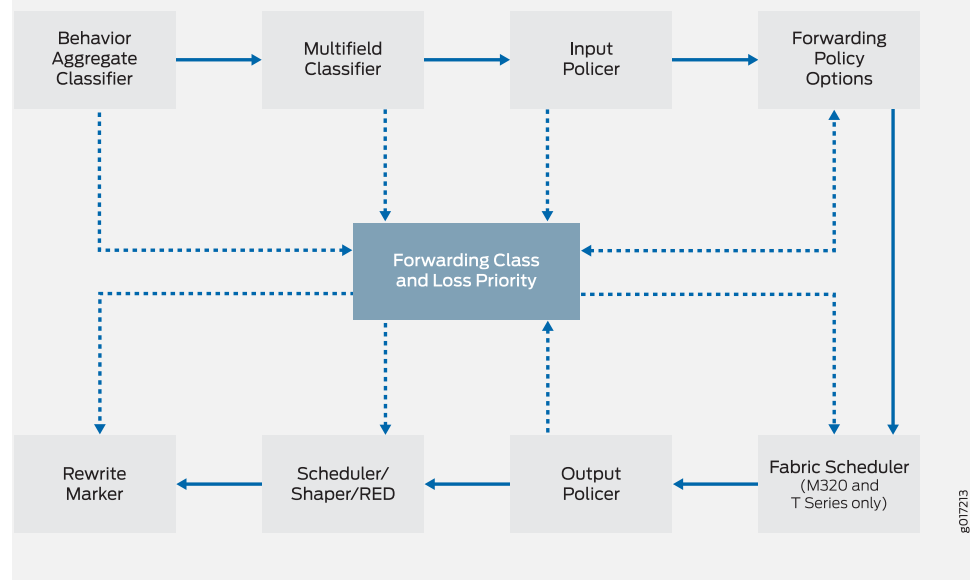


Figure 5: Packet Flow Through CoS- Configurable Components



Each outer box in Figure 5 on page 18 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 routers different in detail, the overall flow of a packet within the router remains consistent.

When a packet enters a Juniper Networks router, 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 OS CoS Process on page 19](#)
- *Packet Flow on Juniper Networks M Series Multiservice Edge Routers*
- *Packet Flow on MX Series 3D Universal Edge Routers*
- *Packet Flow on Juniper Networks T Series Core Routers*

Configuring Basic Packet Flow Through the Junos OS CoS Process

Figure 6 on page 19 and Figure 7 on page 20 show the components of the Junos OS CoS features, illustrating the sequence in which they interact.

Figure 6: CoS Classifier, Queues, and Scheduler

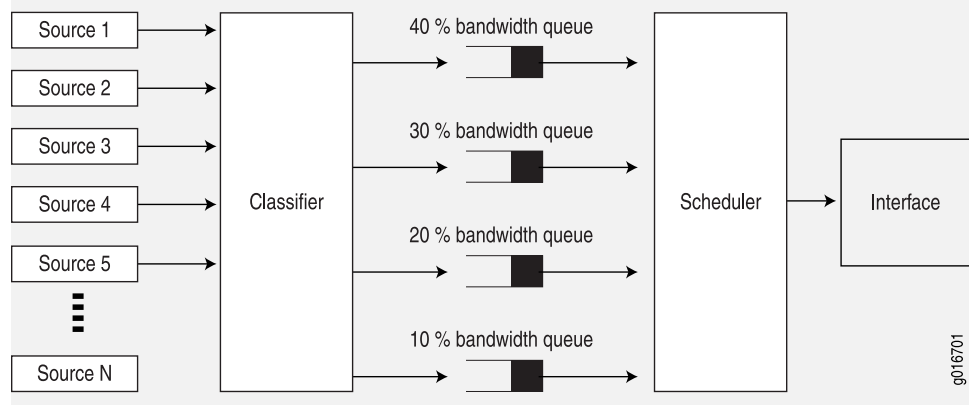
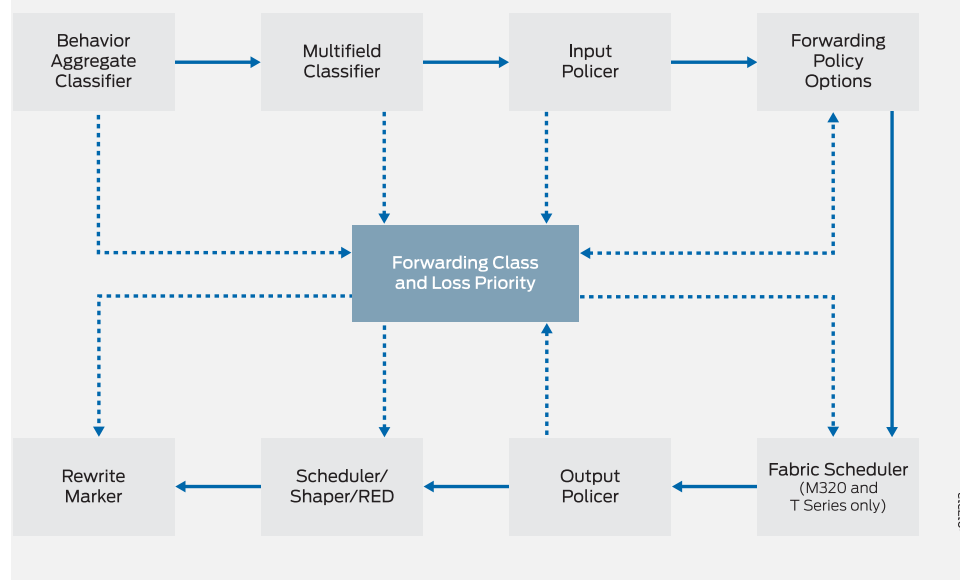


Figure 7: Packet Flow Through CoS- Configurable Components



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

- [Define Classifiers on page 20](#)
- [Apply Classifiers to Incoming Packets on Interfaces on page 21](#)
- [Define Policers to Limit Traffic and Control Congestion on page 22](#)
- [Define Drop Profiles on page 23](#)
- [Assign Each Forwarding Class to a Queue on page 23](#)
- [Define Schedulers on page 23](#)
- [Define Scheduler Maps on page 24](#)
- [Define CoS Header Rewrite Rules on page 24](#)
- [Apply Scheduler Maps and Rewrite Rules to Egress Interfaces on page 25](#)

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 low code-points 000;
      loss-priority high code-points 001;
    }
    forwarding-class video-queue {
      loss-priority low code-points 010;
    }
  }
}
```

```

        loss-priority high code-points 011;
    }
    forwarding-class voice-queue {
        loss-priority low code-points 100;
        loss-priority high code-points 101;
    }
    forwarding-class nc-queue {
        loss-priority low code-points 110;
        loss-priority high code-points 111;
    }
}

```

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**
- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)
 - [Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields on page 97](#)

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 on page 117](#)

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;  
  }  
}
```

See Also • [Managing Congestion Using RED Drop Profiles and Packet Loss Priorities on page 343](#)

Assign Each Forwarding Class to a Queue

To provide differentiated services to each forwarding class, assign each forwarding class to its 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 on page 199](#)

Define Schedulers

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

```
[edit class-of-service]  
schedulers { #  
  data-scheduler {
```

```
    transmit-rate percent 50;
    buffer-size percent 50;
    priority low;
    drop-profile-map loss-priority high protocol any drop-profile be-red;
  }
  video-scheduler {
    transmit-rate percent 25;
    buffer-size percent 25;
    priority strict-high;
  }
  voice-scheduler {
    transmit-rate percent 20;
    buffer-size percent 20;
    priority high;
  }
  nc-scheduler {
    transmit-rate percent 5;
    buffer-size percent 5;
    priority high;
  }
}
```

See Also • [How Schedulers Define Output Queue Properties on page 248](#)

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 video-queue scheduler video-scheduler;
    forwarding-class voice-queue scheduler voice-scheduler;
    forwarding-class nc-queue scheduler nc-scheduler;
  }
}
```

See Also • [Configuring Scheduler Maps on page 252](#)

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 low code-point 000;
        loss-priority high code-point 001;
    }
    forwarding-class voice-queue {
        loss-priority low code-point 010;
        loss-priority high code-point 011;
    }
    forwarding-class video-queue {
        loss-priority low code-point 100;
        loss-priority high code-point 101;
    }
    forwarding-class nc-queue {
        loss-priority low code-point 110;
        loss-priority high code-point 111;
    }
}
}
```

See Also • [Rewriting Packet Headers to Ensure Forwarding Behavior on page 378](#)

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 on page 253](#)
• [Applying Rewrite Rules to Output Logical Interfaces on page 391](#)

Related Documentation • [Packet Flow Through the Junos OS CoS Process Overview on page 17](#)

Example: Classifying All Traffic from a Remote Device by Configuring Fixed Interface-Based Classification

This example shows the configuration of fixed classification based on the incoming interface. Fixed classification can be based on the physical interface (such as an ATM or

Gigabit Ethernet interface) or a logical interface (such as an Ethernet VLAN, a Frame Relay DLCI, or an MPLS tunnel).

- [Requirements on page 26](#)
- [Overview on page 26](#)
- [Configuration on page 27](#)
- [Verification on page 30](#)

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 SRX Series devices running Junos OS Release 12.1. The SRX devices are configured to run as routers.



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

Overview

A fixed interface classifier is the simplest way to classify all packets from a specific interface to a forwarding class. This is typically used on edge routers to classify all traffic from a remote router or server 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 class of service.

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 classes of service. 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 OS, 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;
```

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

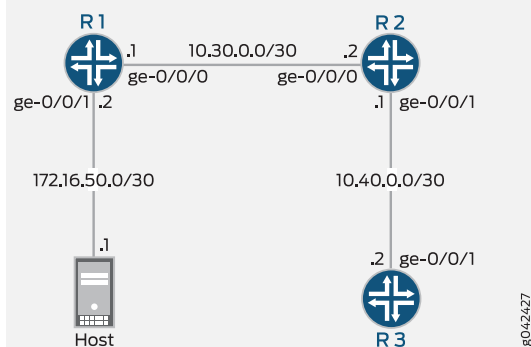


Video: [Class of Service Basics, Part 2: Classification Learning Byte](#)

Topology

Figure 8 on page 27 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 27 shows the configuration for all of the Juniper Networks devices in Figure 8 on page 27. The section “Step-by-Step Procedure” on page 28 describes the steps on Device R2.

Configuration

CLI Quick Configuration

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

Device R1 set interfaces 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 behavior aggregate 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-or-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

Confirm that the configuration is working properly.

- [Verifying a Fixed-Interface Classifier on page 30](#)

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

- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)
- [Default Aliases for CoS Value Bit Patterns Overview on page 42](#)
- [Managing Congestion Using RED Drop Profiles and Packet Loss Priorities on page 343](#)

Interface Types That Do Not Support Junos OS CoS

For original Channelized OC12 PICs, limited CoS functionality is supported. For more information, contact Juniper Networks customer support.

The standard Junos OS CoS hierarchy is not supported on ATM interfaces. ATM has traffic-shaping capabilities that would override CoS, because ATM traffic shaping is performed at the ATM layer and CoS is performed at the IP layer. For more information about ATM traffic shaping and ATM CoS components, see the *Junos OS Network Interfaces Library for Routing Devices*.



NOTE: Transmission scheduling is not supported on 8-port, 12-port, and 48-port Fast Ethernet PICs.

You can configure CoS on all interfaces, except the following:

- **cau4**—Channelized STM1 IQ interface (configured on the Channelized STM1 IQ PIC).
- **coc1**—Channelized OC1 IQ interface (configured on the Channelized OC12 IQ PIC).
- **coc12**—Channelized OC12 IQ interface (configured on the Channelized OC12 IQ PIC).
- **cstm-1**—Channelized STM1 IQ interface (configured on the Channelized STM1 IQ PIC).
- **ct1**—Channelized T1 IQ interface (configured on the Channelized DS3 IQ PIC or Channelized OC12 IQ PIC).
- **ct3**—Channelized T3 IQ interface (configured on the Channelized DS3 IQ PIC or Channelized OC12 IQ PIC).
- **ce1**—Channelized E1 IQ interface (configured on the Channelized E1 IQ PIC or Channelized STM1 IQ PIC).
- **dsc**—Discard interface.
- **fxp**—Management and internal Ethernet interfaces.
- **lo**—Loopback interface. This interface is internally generated.
- **pe**—Encapsulates packets destined for the rendezvous point router. This interface is present on the first-hop router.
- **pd**—De-encapsulates packets at the rendezvous point. This interface is present on the rendezvous point.
- **vt**—Virtual loopback tunnel interface.



NOTE: For channelized interfaces, you can configure CoS on channels, but not at the controller level.

Related Documentation

- *CoS on ATM Interfaces Overview*

PART 2

Configuring Class of Service

- [Assigning Service Levels to Trusted Packets Using Behavior Aggregate Classifiers on page 37](#)
- [Assigning Service Levels to Untrusted Packets by Using Multifield Classifiers on page 97](#)
- [Controlling Access to the Network Using Traffic Policing on page 117](#)
- [Defining Forwarding Behavior Based on Forwarding Classes on page 199](#)
- [Defining Output Queue Properties Using Schedulers on page 247](#)
- [Controlling Bandwidth Using Scheduler Rates on page 263](#)
- [Setting Transmission Order Using Scheduler Priorities and Hierarchical Scheduling on page 319](#)
- [Controlling Congestion Using Scheduler RED Drop Profiles and Buffers on page 343](#)
- [Altering Outgoing Packet Headers Using Rewrite Rules to Ensure Forwarding Behavior on page 377](#)
- [Altering Class of Service Values in Packets Exiting the Network Using IPv6 DiffServ on page 461](#)

CHAPTER 2

Assigning Service Levels to Trusted Packets Using Behavior Aggregate Classifiers

- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)
- [Default Aliases for CoS Value Bit Patterns Overview on page 42](#)
- [Defining Aliases for CoS Value Bit Patterns on page 46](#)
- [Applying Behavior Aggregate Classifiers to Logical Interfaces on page 49](#)
- [Configuring Behavior Aggregate Classifiers on page 53](#)
- [Default DSCP and DSCP IPv6 Classifiers on page 55](#)
- [Applying DSCP Classifiers to MPLS Traffic on page 56](#)
- [Example: Configuring and Applying a Default DSCP Behavior Aggregate Classifier on page 61](#)
- [Understanding DSCP Classification for VPLS on page 69](#)
- [Example: Configuring DSCP Classification for VPLS on page 70](#)
- [Example: Configuring Behavior Aggregate Classifiers on page 73](#)
- [Default MPLS EXP Classifier on page 83](#)
- [Applying MPLS EXP Classifiers to Routing Instances on page 84](#)
- [Applying MPLS EXP Classifiers for Explicit-Null Labels on page 91](#)
- [Default IEEE 802.1p Classifier on page 92](#)
- [Default IEEE 802.1ad Classifier on page 93](#)
- [Default IP Precedence Classifier on page 94](#)

Understanding 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.



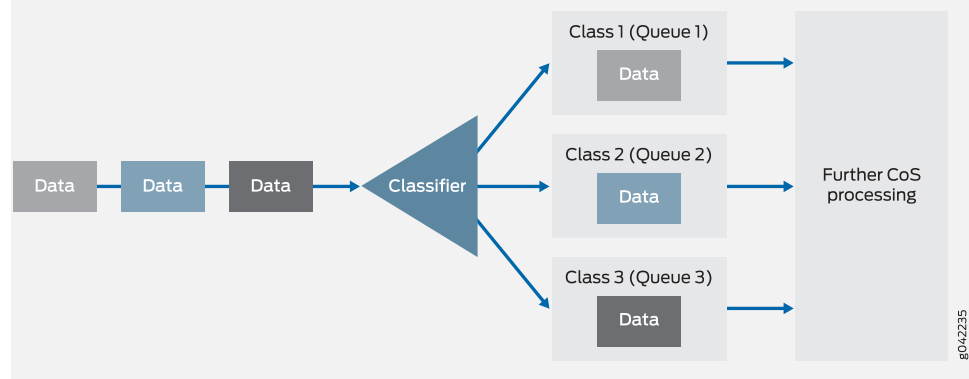
NOTE: Support was added for filtering on Differentiated Services Code Point (DSCP) and forwarding class for Routing Engine sourced packets, including IS-IS packets encapsulated in generic routing encapsulation (GRE). Subsequently, when upgrading from a previous version of Junos OS where you have both a class of service (CoS) and firewall filter, and both include DSCP or forwarding class filter actions, the criteria in the firewall filter automatically takes precedence over the CoS settings. The same is true when creating new configurations; that is, where the same settings exist, the firewall filter takes precedence over the CoS, regardless of which was created first.

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 97](#)) 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 378](#)), then use the more efficient BA classification for transversing 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 9 on page 39](#) provides a high-level illustration of how a classifier works.

Figure 9: 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 (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)

Unlike multifield classifiers (which are discussed in [“Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields” on page 97](#)), BA classifiers are based on fixed-length fields, which makes them computationally more efficient than multifield classifiers. For this reason, core devices are normally configured to perform BA classification, because of the higher traffic volumes they 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 ingress 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 378](#).



NOTE: 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. Classifiers for IP (DSCP or IP precedence) and MPLS (EXP) can coexist on a logical interface if the hardware requirements are met.

For Juniper Networks M Series Multiservice Edge Routers, four classes can forward traffic independently. For M320 Multiservice Edge Routers, T Series Core Routers, MX Series 5G Universal Routing Platforms, and PTX Series Packet Transport Routers, eight classes can forward traffic independently. 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 if they are both either protocol specific or protocol independent, the BA classifier followed by the multifield classifier, any BA classification result is overridden by a multifield classifier if they conflict.

In the case that a protocol-specific BA classifier and a protocol-independent firewall filter are both configured together, the protocol-independent filter is processed before the protocol-specific BA classifier, regardless of 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 97](#). 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 226](#).

If you do nothing to configure or assign classifiers, Junos OS automatically assigns an implicit default IP precedence classifier to all logical interfaces that 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 OS CoS policy reserves 5 percent of available bandwidth for **network-control** traffic and 95 percent for **best-effort** traffic. Junos OS provides a range of default BA classifiers that you can apply to logical interfaces and that map various CoS values to **assured-forwarding** and **expedited-forwarding** forwarding classes as well as to the **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 OS CoS policy, 95 percent of the bandwidth for queue 0 and 5 percent for queue 3 on routing devices (see [“Default Schedulers Overview” on page 251](#)), is in effect regardless of any custom BA classifier or forwarding class definitions, until you configure a custom scheduler (see [“Configuring Schedulers” on page 252](#)).

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 (see [“Default MPLS EXP Classifier” on page 83](#)) 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 [“Rewriting MPLS and IPv4 Packet Headers” on page 393](#)).

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 92](#).



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.

Release History Table

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

Related Documentation

- [Default IP Precedence Classifier on page 94](#)
- [Default DSCP and DSCP IPv6 Classifiers on page 55](#)
- [Default MPLS EXP Classifier on page 83](#)
- [Default IEEE 802.1p Classifier on page 92](#)
- [Default IEEE 802.1ad Classifier on page 93](#)
- [Configuring Behavior Aggregate Classifiers on page 53](#)
- [Applying Behavior Aggregate Classifiers to Logical Interfaces on page 49](#)
- [Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields on page 97](#)
- [Rewriting Packet Headers to Ensure Forwarding Behavior on page 378](#)

Default Aliases for CoS Value Bit Patterns Overview

Behavior aggregate (BA) classifiers use class-of-service (CoS) values—such as Differentiated Services code points (DSCPs), DSCP IPv6, IP precedence, IEEE 802.1, and MPLS experimental (EXP) bits—to associate incoming packets with a particular CoS servicing 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** (expedited forwarding).



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

When you define classifiers, you can refer to the markers by alias names. You can configure user-defined classifiers in terms of alias names. If the value of an alias changes, it alters the behavior of any classifier that references it.

[Table 4 on page 43](#) shows the default mappings between the CoS values and standard aliases.

Table 4: 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
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

Table 4: Default CoS Value Aliases (continued)

Default CoS Value Alias	CoS Value
cs3	011000
cs4	100000
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
ef	010
ef1	011
af11	100
af12	101
nc1/cs6	110
nc2/cs7	111
IEEE 802.1ad CoS Aliases and CoS Values	
be	0000

Table 4: Default CoS Value Aliases (continued)

Default CoS Value Alias	CoS Value
be-dei	0001
be1	0010
be1-dei	0011
ef	0100
ef-dei	0101
ef1	0110
ef1-dei	0111
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
nc1/cs6	110
nc2/cs7	111

**Related
Documentation**

- [Defining Aliases for CoS Value Bit Patterns on page 46](#)
- [Default IP Precedence Classifier on page 94](#)
- [Default DSCP and DSCP IPv6 Classifiers on page 55](#)
- [Default MPLS EXP Classifier on page 83](#)
- [Default IEEE 802.1p Classifier on page 92](#)
- [Default IEEE 802.1ad Classifier on page 93](#)
- *code-point-aliases*

Defining 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-802.1ad**—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.
 - Nothing in the default mapping table is changed by this statement.
- **my2 101110:**
 - 101110 is now mapped to **my2** as well as **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 are now mapped to the default forwarding class.
- **cs7 110000:**
 - **cs7** is now mapped to 110000, as well as **nc1** and **cs6**.
 - The old value of **cs7**, 111000, is still mapped to **nc2**.

**Related
Documentation**

- [Default Aliases for CoS Value Bit Patterns Overview on page 42](#)
- [Applying Behavior Aggregate Classifiers to Logical Interfaces on page 49](#)

Applying Behavior Aggregate Classifiers to Logical Interfaces

This topic describes how to apply behavior aggregate (BA) classifiers to logical interfaces.

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

For most PICs, if you apply an IEEE 802.1 classifier to a logical interface, you cannot apply non-IEEE classifiers to other logical interfaces on the same physical interface. This restriction does not apply to Gigabit Ethernet IQ2 PICs.

There are some restrictions on applying multiple BA classifiers to a single logical interface. [Table 5 on page 49](#) shows the supported combinations. In this table, the OSE PICs refer to the 10-port 10-Gigabit OSE PICs.

Table 5: Logical Interface Classifier Combinations

Classifier Combinations	Gigabit Ethernet IQ2 PICs	OSE PICs	Other PICs on M320, MX Series, T Series routers and on EX Series Switches	Other M Series with Regular FPCs	Other M Series with Enhanced FPCs
dscp and inet-precedence	No	No	No	No	No
dscp-ipv6 and (dscp inet-precedence)	Yes	Yes	Yes	No	No
exp and ieee 802.1	Yes	Yes	No	No	No
ieee 802.1 and (dscp dscp-ipv6 exp inet-precedence)	Yes	Yes	No	No	Yes
exp and (dscp dscp-ipv6 inet-precedence)	Yes	Yes	Yes	No	Yes

For Gigabit Ethernet IQ2 and 10-port 10-Gigabit Oversubscribed Ethernet (OSE) interfaces, 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.



NOTE: For an interface on an M Series FPC, you can apply only the default exp classifier. For an enhanced FPC, you can create a new exp classifier and apply it to an interface.

On MX960, MX480, MX240, MX80, M120, and M320 routers and EX Series switches with Enhanced Type III FPCs only, you can configure user-defined DSCP-based BA classification for MPLS interfaces (this feature is not available for IQE PICs or on MX Series routers and EX Series switches when ingress queuing is used) or VPLS or Layer 3 VPN routing instances (LSI interfaces). The DSCP-based classification for MPLS packets for Layer 2 VPNs is not supported.



NOTE: If you do not apply a DSCP classifier, the default EXP classifier is applied to MPLS traffic. At times 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, on MX Series devices only, 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 a Layer 3 VPN using a label-switched interface (LSI) routing instance.
 - Supported on the M120, M320, MX960, MX480, MX240, and MX80 routers.
 - DSCP classifier applied under **[edit class-of-service routing-instances]** on the egress provider edge (PE) router.
- In VPLS using an LSI routing instance.
 - Supported on the M120, M320, MX960, MX480, MX240, and MX80 routers.
 - DSCP classifier applied under **[edit class-of-service routing-instances]** on the egress PE router.
- In a Layer 3 VPN using a virtual tunnel (VT) routing instance.
 - Supported on the M120, M320, MX960, MX480, MX240, and MX80 routers.
 - 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.
 - Supported on the M120, M320, MX960, MX480, MX240, and MX80 routers (not supported on IQE and MX when ingress queuing is enabled).
 - DSCP classifier applied under **[edit class-of-service interfaces]** on the ingress core-facing interface on the provider (P) or egress PE router.

MPLS forwarding when the label stacking is greater than 2 is not supported.

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 84](#).
- 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 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]
user@host# 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-classifier {
    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

- [Applying DSCP Classifiers to MPLS Traffic on page 56](#)

Configuring Behavior Aggregate Classifiers

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

The BA classifiers map sets the forwarding class and packet loss priority (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 11](#).

The classifiers work as follows:

- **dscp**—Handles incoming IPv4 packets.
- **dscp-ipv6**—Handles incoming IPv6 packets.
- **exp**—Handles MPLS packets using Layer 2 headers.
- **ieee-802.1**—Handles Layer 2 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 M Series, MX Series, and T Series routers, and EX Series switches that do not have tricolor marking enabled, the loss priority can be configured only by setting the PLP within a multifold 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 353](#).

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 new 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, four classifier PLP designations are supported: **low**, **medium-low**, **medium-high**, and **high**. For example, in the following configuration, the **assured-forwarding** forwarding class and **medium-low** PLP are assigned 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 199.

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

- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)
- [Applying Behavior Aggregate Classifiers to Logical Interfaces on page 49](#)
- [Enabling Tricolor Marking and Limitations of Three-Color Policers on page 123](#)

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.



NOTE: If you deactivate or delete the `dscp-ipv6` statement from the configuration, the default IPv6 classifier is not activated on the M5, M10, M7i, M10i, M20, M40, M40e, and M160 routing platforms. As a workaround, explicitly specify the default option to the `dscp-ipv6` statement.

Table 6 on page 56 shows the forwarding class and packet loss priority (PLP) that are assigned to each well-known DSCP when you apply the explicit default DSCP or DSCP IPv6 classifier.

Table 6: 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

- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)
- [Default Aliases for CoS Value Bit Patterns Overview on page 42](#)
- [Changing the Default Queuing and Marking of Host Outbound Traffic on page 236](#)
- *classifiers (Logical Interface)*

Applying DSCP Classifiers to MPLS Traffic

On MX960, MX480, MX240, MX80, M120, and M320 routers with Enhanced Type III FPCs and EX Series switches only, you can configure user-defined DSCP-based 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 IQE PICs or 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.

- [Applying a DSCP Classifier to MPLS Packets on a Core-facing Interface on page 57](#)
- [Applying a DSCP Classifier to MPLS Traffic for L3VPN/VPLS on page 59](#)

Applying 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 *CLI User Guide*.

The following example:

- Configures core-facing interface ge-5/3/1.0 for protocol families IPv4, IPv6, and International Organization for Standardization Open Systems Interconnection (ISO OSI)
- Configures the DSCP classifier **dscp11**.
- 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:

- 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
```

- 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]
```

- 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
```

Applying 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 *CLI User Guide*.

The following example:

- a. Configures routing instances of type either vrf or vpls.
- b. Configures the DSCP classifier.
- c. 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 either 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

- [Applying Behavior Aggregate Classifiers to Logical Interfaces on page 49](#)

Example: Configuring and Applying a Default DSCP Behavior Aggregate Classifier

A Junos OS classifier identifies and separates traffic flows and provides the means to prioritize traffic later in the class-of-service (CoS) process.

A behavior aggregate (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 a default classifier is not applied to an interface, it does not take effect.

Junos OS 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 (BA) DiffServ code point (DSCP) classifier and verify its functionality.

- [Requirements on page 61](#)
- [Overview on page 62](#)
- [Configuration on page 65](#)
- [Verification on page 67](#)

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. This approach is shown as well.

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

Overview

The basis of Junos OS CoS is traffic differentiation. Assigning traffic to different classes of service provides the necessary differentiation. From the point of view of a router, the class of service 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.

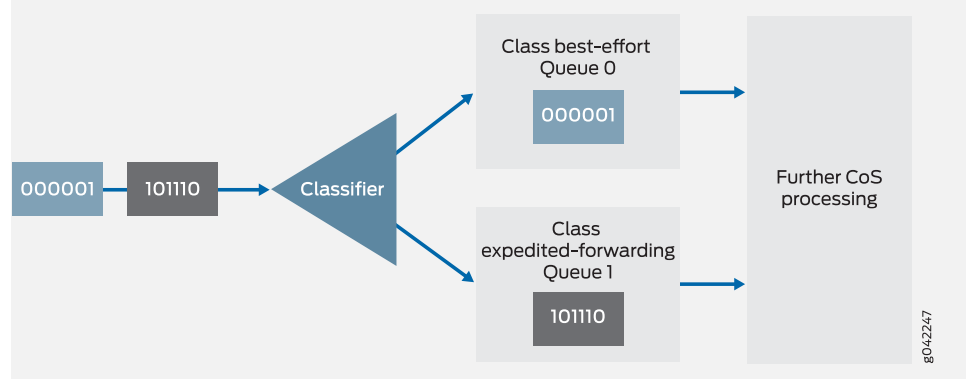
```
user@host> show class-of-service drop-profile
```

```
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 10 on page 64](#). The figure shows two IPv4 packets entering an interface and being classified according to the DSCP code points in the packet headers.

Figure 10: Behavior Aggregate Classifier with Two Queues



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

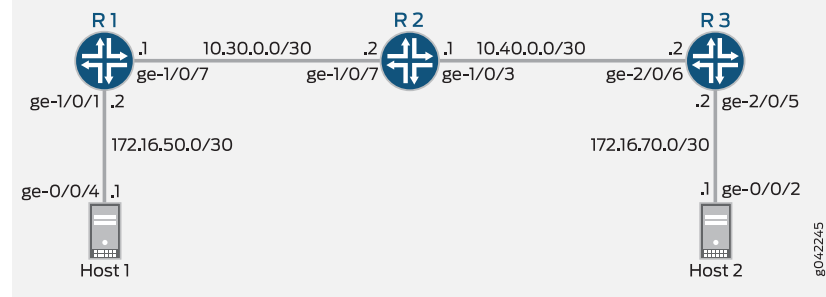


Video: [Class of Service Basics, Part 2: Classification Learning Byte](#)

Topology

[Figure 11 on page 65](#) shows the sample network.

Figure 11: 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 65 shows the configuration for all of the Juniper Networks devices in Figure 11 on page 65. The section “Step-by-Step Procedure” on page 66 describes the steps on Device R2.

Configuration

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

Device R1

```
set interfaces 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-1/0/9 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
```

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

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.

```
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
```

```
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

- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)
- [Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows on page 353](#)

Understanding DSCP Classification for VPLS

You can perform Differentiated Services Code Point (DSCP) classification for IPv4 packets on Ethernet interfaces that are part of a virtual private LAN service (VPLS) routing instance on the ingress provider edge (PE) router. This is supported on the M320 router with Enhanced type III FPC and the M120 router. On the ATM II IQ PIC, the **ether-vpls-over-atm-llc** encapsulation statement is required. On the Intelligent Queuing 2 (IQ2) or Intelligent Queuing 2 Enhanced (IQ2E) PICs, the **vlan-vpls** encapsulation statement is required. DSCP for IPv6 and Internet precedence for IPv6 are not supported.

In order 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 make sure of 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 with the **no-tunnel-services** option configured has a default MPLS EXP classifier applied to the label-switched interface for all VPLS packets coming from the remote VPLS PE. This default classifier is modifiable only on MX Series routers. On T Series, when **no-tunnel-services** option is configured, the custom classifier for VPLS instances is not supported.



NOTE: With **no-tunnel-services** configured, a custom classifier for VPLS routing instances on T Series and LMNR based FPC for M320 is not supported. When a wild card configuration or explicit routing instances are configured for VPLS on CoS CLI, the custom classifier binding results in default classifier binding on Packet Forwarding Engine (PFE).

For example, on routing devices with eight queues (Juniper Networks M120 and M320 Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, and T Series Core Routers), the default classification applied to **no-tunnel-services** VPLS packets are shown in [Table 7 on page 70](#).

Table 7: Default VPLS Classifiers

MPLS Label EXP Bits	Forwarding Class/Queue
000	0
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 206](#).

On MX Series routers, VPLS filters and policers act on a Layer 2 frame that includes the media access control (MAC) header (after any VLAN rewrite or other rules are applied), 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 will not tally these packets.

Related Documentation

- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)

Example: Configuring DSCP Classification for VPLS

This example shows how to configure a DSCP classifier for a virtual private LAN service (VPLS).

- [Requirements on page 71](#)
- [Overview on page 71](#)
- [Configuration on page 71](#)

Requirements

This example uses the following hardware and software components:

- An M Series Multiservice Edge Router (M120 and M320 only), MX Series 5G Universal Routing Platform, or T Series Core Router (TX Matrix and TX Matrix Plus only) with an ATM interface.
- Junos OS Release 10.4 or later.

Overview

In this example, you configure a DSCP classifier **dscp_vpls** on ATM interface **at-4/1/1** with **ether-vpls-over-atm-llc** encapsulation. The classifier **dscp_vpls** is applied to the interface and the interface is listed in the VPLS routing instance **vpls1** on the ingress PE router.

Configuration

CLI Quick Configuration

To quickly configure the DSCP classifier for a virtual private LAN service (VPLS), copy the following commands to a text file, remove any line breaks, and then paste the commands into the CLI.

```
user@host# set interfaces at-4/1/1 mtu 9192
user@host# set interfaces at-4/1/1 atm-options vpi 10
user@host# set interfaces at-4/1/1 unit 0 encapsulation ether-vpls-over-atm-llc
user@host# set interfaces at-4/1/1 unit 0 vci 10.128
user@host# set interfaces at-4/1/1 unit 0 family vpls
user@host# set class-of-service classifiers dscp dscp_vpls forwarding-class
  expedited-forwarding loss-priority low code-points 000010
user@host# set interfaces at-4/1/1 unit 0 classifiers dscp dscp_vpls
user@host# set routing-instances vpls1 instance-type vpls
user@host# set routing-instances vpls1 interface at-4/1/1.0
user@host# set routing-instances vpls1 route-distinguisher 10.255.245.51:1
user@host# set routing-instances vpls1 vrf-target target:1234:1
user@host# set routing-instances vpls1 protocols vpls site-range 10
user@host# set routing-instances vpls1 protocols vpls no-tunnel-services
user@host# set routing-instances vpls1 protocols vpls site vpls-1-site-1 site-identifier 1
```

Configuring the DSCP Classifier for a Virtual Private LAN Service (VPLS)

Step-by-Step Procedure

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

To configure the DSCP classifier for a virtual private LAN service (VPLS):

1. Configure the ATM interface **at-4/1/1.0** and the encapsulation as **ether-vpls-over-atm-llc**.

```
[edit interfaces]
```

```

user@host# set at-4/1/1 mtu 9192
user@host# set at-4/1/1 atm-options vpi 10
user@host# set at-4/1/1 unit 0 encapsulation ether-vpls-over-atm-llc
user@host# set at-4/1/1 unit 0 vci 10.128
user@host# set at-4/1/1 unit 0 family vpls

```

2. Configure the DSCP classifier **dscp_vpls**.

```

[edit class-of-service]
user@host# set classifiers dscp dscp_vpls forwarding-class expedited-forwarding
loss-priority low code-points 000010

```

3. Apply the classifier **dscp_vpls** to the ATM interface **at-4/1/1.0**.

```

[edit interfaces]
user@host# set at-4/1/1 unit 0 classifiers dscp dscp_vpls

```

4. Include the ATM interface virtual circuit **at-4/1/1.0** as part of the routing instance **vpls1** configuration.

```

user@host# set routing-instances vpls1 instance-type vpls
user@host# set routing-instances vpls1 interface at-4/1/1.0
user@host# set routing-instances vpls1 route-distinguisher 10.255.245.51:1
user@host# set routing-instances vpls1 vrf-target target:1234:1
user@host# set routing-instances vpls1 protocols vpls site-range 10
user@host# set routing-instances vpls1 protocols vpls no-tunnel-services
user@host# set routing-instances vpls1 protocols vpls site vpls-1 site-1 site-identifier
1

```

Results

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

```

user@host# show interfaces at-4/1/1

```

```

mtu 9192;
atm-options {
    vpi 10;
}
unit 0 {

```

```

classifiers {
  dscp dscp_vpls;
}

encapsulation ether-vpls-over-atm-llc;
vci 10.128;
family vpls;
}

```

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

```

classifiers {
  dscp dscp_vpls {
    forwarding-class expedited-forwarding {
      loss-priority low code-points 000010;
    }
  }
}

```

```
user@host# show routing-instances
```

```

vpls1 {
  instance-type vpls;
  interface at-4/1/1.0;
  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;
      }
    }
  }
}

```

Related Documentation

- [Understanding DSCP Classification for VPLS on page 69](#)

Example: Configuring Behavior Aggregate Classifiers

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

- [Requirements on page 74](#)
- [Overview on page 74](#)

- [Configuration on page 75](#)
- [Verification on page 78](#)

Requirements

Before you begin, determine the forwarding class and PLP that are assigned by default to each well-known DSCP that you want to configure for the behavior aggregate classifier. See *Default Behavior Aggregate Classification*.

Overview

You configure behavior aggregate classifiers to classify packets that contain valid DSCPs to appropriate queues. Once configured, you must apply the behavior aggregate 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 behavior aggregate 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, and a network control forwarding class as nc-class. Finally, you apply the behavior aggregate classifier to an interface called ge-0/0/0.

[Table 8 on page 74](#) shows how the behavior aggregate classifier assigns loss priorities, to incoming packets in the four forwarding classes.

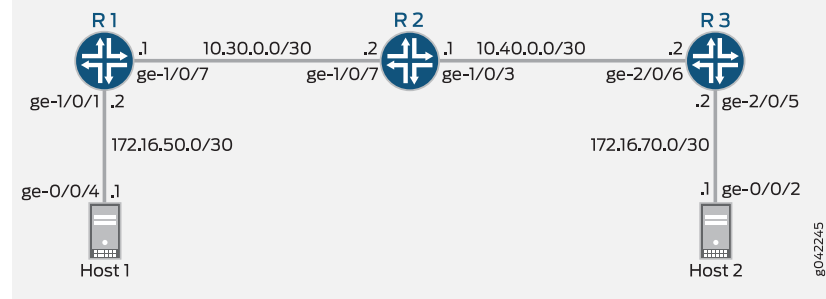
Table 8: Sample ba-classifier Loss Priority Assignments

mf-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

Topology

[Figure 12 on page 75](#) shows the sample network.

Figure 12: Behavior Aggregate Classifier Scenario



“CLI Quick Configuration” on page 75 shows the configuration for all of the Juniper Networks devices in Figure 12 on page 75.

The section “Step-by-Step Procedure” on page 75 describes the steps on Device R2.

Configuration

CLI Quick Configuration

To quickly configure this example, copy the following commands, paste them into a text file, remove any line breaks, change any details necessary to match your network configuration, 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;
    }
  }
}
forwarding-classes {
  class BE-data queue-num 0;
  class Premium-data queue-num 1;
  class Voice queue-num 2;
  class NC queue-num 3;
}
interfaces {
  ge-0/0/0 {
    unit 0 {
      classifiers {
        dscp ba-classifier;
      }
    }
  }
  ge-1/0/9 {
    unit 0 {
      classifiers {
        dscp v4-ba-classifier;
      }
    }
  }
  ge-1/0/9 {
    unit 0 {
      classifiers {
        dscp v4-ba-classifier;
      }
    }
  }
  ge-1/0/9 {
    unit 0 {
      classifiers {
        dscp v4-ba-classifier;
      }
    }
  }
}

```

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

Verification

Confirm that the configuration is working properly.

- [Verifying the Code-Point Aliases on page 78](#)
- [Verifying the DSCP Classifier on page 79](#)
- [Verifying the Forwarding Classes and Output Queues on page 80](#)
- [Verifying That the Classifier Is Applied to the Interfaces on page 81](#)
- [Verifying Behavior Aggregate Classifiers on page 81](#)

Verifying the Code-Point Aliases

Purpose Make sure that the code-point aliases are configured as expected.

Action On Device R2, run the `show class-of-service code-point-aliases dscp` command.

```
user@R2> show class-of-service code-point-aliases dscp
```

```
Code point type: dscp
Alias      Bit pattern
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        000000
be1      000001
cs1       001000
cs2       010000
cs3       011000
cs4       100000
cs5       101000
cs6       110000
cs7       111000
ef        101110
ef1      101111
nc1       110000
nc2       111000
```

Meaning The code-point aliases are configured as expected. Notice that the custom aliases that you configure are added to the default code-point aliases.

Verifying the DSCP Classifier

Purpose Make sure that the DSCP classifier is configured as expected.

Action On Device R2, run the **show class-of-service classifiers name v4-ba-classifier** command.

```
user@R2> show class-of-service classifiers name v4-ba-classifier
```

```
Classifier: v4-ba-classifier, Code point type: dscp, Index: 10755
Code point      Forwarding class      Loss priority
000000          BE-data              high
000001          BE-data              low
000010          BE-data              low
000011          BE-data              low
000100          BE-data              low
000101          BE-data              low
000110          BE-data              low
000111          BE-data              low
001000          BE-data              low
001001          BE-data              low
001010          Voice                low
001011          BE-data              low
001100          Voice                high
001101          BE-data              low
001110          Voice                high
001111          BE-data              low
010000          BE-data              low
010001          BE-data              low
010010          BE-data              low
010011          BE-data              low
010100          BE-data              low
010101          BE-data              low
010110          BE-data              low
010111          BE-data              low
011000          BE-data              low
011001          BE-data              low
011010          BE-data              low
011011          BE-data              low
011100          BE-data              low
011101          BE-data              low
011110          BE-data              low
011111          BE-data              low
100000          BE-data              low
100001          BE-data              low
100010          BE-data              low
100011          BE-data              low
100100          BE-data              low
100101          BE-data              low
100110          BE-data              low
100111          BE-data              low
101000          BE-data              low
101001          BE-data              low
101010          BE-data              low
101011          BE-data              low
101100          BE-data              low
101101          BE-data              low
101110          Premium-data         high
```

101111	Premium-data	low
110000	NC	low
110001	BE-data	low
110010	BE-data	low
110011	BE-data	low
110100	BE-data	low
110101	BE-data	low
110110	BE-data	low
110111	BE-data	low
111000	NC	low
111001	BE-data	low
111010	BE-data	low
111011	BE-data	low
111100	BE-data	low
111101	BE-data	low
111110	BE-data	low
111111	BE-data	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@R2> show class-of-service classifier name v4-ba-classifier
```

```
Classifier: v4-ba-classifier, Code point type: dscp, Index: 10755
Code point      Forwarding class      Loss priority
000000          BE-data                high
000001          BE-data                low
101110          Premium-data          high
101111          Premium-data          low
```

Verifying the Forwarding Classes and Output Queues

Purpose Make sure that the forwarding classes are configured as expected.

Action On Device R2, run the **show class-of-service forwarding-class** command.

```
user@R2> show class-of-service forwarding-class
```

Forwarding class	ID	Queue	Restricted queue	Fabric
priority Policing priority SPU priority				
BE-data normal low	0	0	0	low
Premium-data normal low	1	1	1	low
Voice normal low	2	2	2	low
NC normal low	3	3	3	low

Meaning The forwarding classes are configured as expected.

Verifying That the Classifier Is Applied to the Interfaces

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

Action On Device R2, run the **show class-of-service interface** command.

```
user@R2> show class-of-service interface ge-1/0/3
```

```
Physical interface: ge-1/0/3, Index: 144
Queues supported: 8, Queues in use: 4
  Scheduler map: <default>, Index: 2
  Congestion-notification: Disabled
```

```
Logical interface: ge-1/0/3.0, Index: 333
Object      Name      Type      Index
Classifier  v4-ba-classifier  dscp      10755
```

```
user@R2> show class-of-service interface ge-1/0/9
```

```
Physical interface: ge-1/0/9, Index: 150
Queues supported: 8, Queues in use: 4
  Scheduler map: <default>, Index: 2
  Congestion-notification: Disabled
```

```
Logical interface: ge-1/0/9.0, Index: 332
Object      Name      Type      Index
Classifier  v4-ba-classifier  dscp      10755
```

Meaning The interfaces are configured as expected.

Verifying Behavior Aggregate Classifiers

Purpose Verify that the behavior aggregate classifiers were configured properly on the device.

Action From configuration mode, enter the **show class-of-service** command.

When you are using **hping** to set the DSCP code points in the IPv4 packet header, the type-of-service (ToS) hex value (in this case, BC) 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 101111, use 10111100. 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 188 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 = 0.404/0.483/1.395/0.207 ms
```

hping Sent from Host 1

```
root@host1> hping 172.16.70.1 --tos BC -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
```

On Device R2, Verify that Queue 2 is Incrementing.

Code point 101111 is associated with Premium-data, which uses queue 1.

```
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                50               50                0
2                0                0                0
3                42               42                0
Queue number:    Mapped forwarding classes
0               BE-data
1               Premium-data
2               Voice
3               NC
...

```

Meaning The output shows that queue 1 has incremented by 50 packets after sending 50 packets through the router.

Related Documentation

- *Interfaces Feature Guide for Security Devices*
- *Classification Overview*
- *Sample Behavior Aggregate Classification*
- *Understanding Packet Loss Priorities*

Default MPLS EXP Classifier

Multiprotocol Label Switching (MPLS) class of service (CoS) works in conjunction with the routing device's general CoS functionality.

When IP traffic enters a label-switched path (LSP) tunnel, the ingress routing device marks all packets with a class-of-service (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 from the egress routing device. The routing devices within the LSP utilize the CoS value set at the ingress routing device. The CoS value is encoded by means of the CoS bits (also known as the EXP or experimental bits).

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

For all PICs except PICs mounted on Juniper Networks M Series Multiservice Edge Router standard (nonenhanced) FPCs, if you enable the MPLS protocol family on a logical interface, the default MPLS EXP classifier is automatically applied to that logical interface.

[Table 9 on page 84](#) lists the default MPLS classifier mapping of EXP bits to forwarding classes and loss priorities..

Table 9: 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

Related Documentation

- [Configuring Class of Service for MPLS LSPs](#)
- [Default Aliases for CoS Value Bit Patterns Overview on page 42](#)
- [code-point-aliases](#)

Applying MPLS EXP Classifiers to Routing Instances

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, the default MPLS EXP classifier is applied to the routing instance. For detailed information about VRF table labels, see the *Junos OS VPNs Library for Routing Devices*.

The default MPLS EXP classification table contents are shown in [Table 10 on page 84](#).

Table 10: 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

Table 10: Default MPLS EXP Classifier (continued)

MPLS EXP Bits	Forwarding Class	Loss Priority
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, which would override the original classifier. Starting with Junos OS Release 16.1, on MX Series devices only, 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;
  }
}
```

This topic describes:

- [Configuring and Applying Custom MPLS EXP Classifiers to Routing Instances on page 85](#)
- [Applying Global Classifiers and Wildcard Routing Instances on page 86](#)
- [Applying Global MPLS EXP Classifiers to Routing Instances on page 87](#)
- [Applying Classifiers by Using Wildcard Routing Instances on page 89](#)
- [Verifying the Classifiers Associated with Routing Instances on page 90](#)

Configuring and Applying Custom MPLS EXP Classifiers to Routing Instances



NOTE: The following caveats apply to custom MPLS EXP classifiers for routing instances:

- An enhanced FPC is required.
- Logical systems are not supported.

For PICs that are installed on enhanced FPCs, 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 *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
```

Applying 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 associated with specific classifiers, the global configuration is ignored.

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 there is a specific configuration, it is given precedence over the wildcard configuration.



NOTE: The wildcard * and the all keyword are supported 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 OS 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, the Junos OS treats the *vpn** routing instance name as a wildcard, and 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 should not be intended as a stand-alone substitute for the all keyword.

Applying 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. In this example, there are 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 *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]
```

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

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

Applying Classifiers by Using Wildcard Routing Instances

Configure a wildcard routing instance and override the wildcard with a specific routing instance. In this example, there are 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 *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;
  }
}
```

Verifying the Classifiers Associated with Routing Instances

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 OS 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

Release History Table

Release	Description
16.1	Starting with Junos OS Release 16.1, on MX Series devices only, you can maintain the original MPLS EXP classifier.

- Related Documentation**
- [Configuring Behavior Aggregate Classifiers on page 53](#)
 - [Default MPLS EXP Classifier on page 83](#)
 - [Applying MPLS EXP Classifiers for Explicit-Null Labels on page 91](#)

Applying 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 ensures that any packets traversing an MPLS network include a label. For more information about explicit-null labels and ultimate-hop popping, see the *MPLS Applications Feature Guide*.

On M320 and T Series routers, 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. To combine an EXP classifier with a distinct IPv6 classifier, the PIC must be mounted on an Enhanced FPC.



NOTE: For M Series routers, MPLS explicit-null labels with MPLS EXP classification are supported if you set the same classifier for EXP and IPv4 traffic, or EXP and IPv6 traffic.

For more information about how IPv4 and IPv6 packet classification is handled, see [“Applying Behavior Aggregate Classifiers to Logical Interfaces” on page 49](#).

To 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
```

To 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
```

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

```
[edit class-of-service classifiers interfaces interface-name ]
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. Starting with Junos OS 18.1R1, PTX Series routers with third-generation FPCs (FPC3) support a new CLI option, **[explicit-null-cos *inet|inet6*]** at the **[edit forwarding-options]** hierarchy level, that makes the packet classification based on the MPLS EXP value rather than on the payload, thus preserving the MPLS classification of the packet.

Release History Table

Release	Description
18.1	Starting with Junos OS 18.1R1, PTX Series routers with third-generation FPCs (FPC3) support a new CLI option, [explicit-null-cos <i>inet inet6</i>] at the [edit forwarding-options] hierarchy level, that 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

- [Configuring Behavior Aggregate Classifiers on page 53](#)
- [Default MPLS EXP Classifier on page 83](#)
- [Applying MPLS EXP Classifiers to Routing Instances on page 84](#)

Default IEEE 802.1p Classifier

Table 11 on page 93 shows the forwarding class and PLP that are assigned to each IEEE 802.1p CoS value when you apply the explicit default IEEE 802.1p classifier. To do this, include the **default** statement at the **[edit class-of-service interfaces *interface-name* unit *logical-unit-number* classifiers ieee-802.1]** hierarchy level:



NOTE: Only the IEEE 802.1p classifier is supported in Layer 2 interfaces. You must explicitly apply this classifier as shown.

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

Table 11: 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

- [Applying Behavior Aggregate Classifiers to Logical Interfaces on page 49](#)
- [Default IEEE 802.1ad Classifier on page 93](#)

Default IEEE 802.1ad Classifier

Table 12 on page 93 shows the forwarding class and packet loss priority (PLP) that are assigned 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 loss priority is determined by the DEI bit. 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 12: Default IEEE 802.1ad Classifier

IEEE 802.1ad CoS Value	Forwarding Class	PLP
0000	best-effort	low

Table 12: Default IEEE 802.1ad Classifier (continued)

IEEE 802.1ad CoS Value	Forwarding Class	PLP
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
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](#)

Default IP Precedence Classifier

By default, all logical interfaces are automatically assigned an implicit IP precedence classifier called **ipprec-compatibility**. The **ipprec-compatibility** IP precedence classifier maps IP precedence bits to forwarding classes and packet loss priorities (PLPs), as shown in [Table 13 on page 94](#).

Table 13: Default IP Precedence (ipprec-compatibility) Classifier

IP Precedence Bits	Forwarding Class	Loss Priority
000	best-effort	low

Table 13: Default IP Precedence (ipprec-compatibility) Classifier (continued)

IP Precedence Bits	Forwarding Class	Loss Priority
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 do this, 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 14 on page 95 shows the forwarding class and PLP that are assigned to the IP precedence bits when you apply the default IP precedence classifier.

Table 14: 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**
- [Applying Behavior Aggregate Classifiers to Logical Interfaces on page 49](#)

CHAPTER 3

Assigning Service Levels to Untrusted Packets by Using Multifield Classifiers

- [Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields on page 97](#)
- [Configuring Multifield Classifiers on page 98](#)
- [Using Multifield Classifiers to Set Packet Loss Priority on page 101](#)
- [Example: Configuring and Applying a Firewall Filter for a Multifield Classifier on page 103](#)
- [Example: Classifying Packets Based on Their Destination Address on page 109](#)
- [Example: Configuring and Verifying a Complex Multifield Filter on page 112](#)

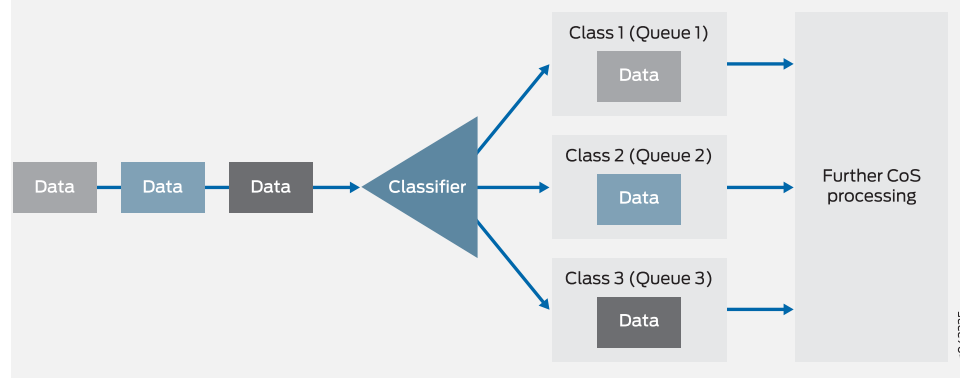
Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields

Behavior aggregate (BA) classification (see [“Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic” on page 38](#)), where packets are classified based on their QoS markings, 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 them 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 DiffServ code point (DSCP) or IP precedence support in end-user applications.

In an edge router, a multifield classifier provides the filtering functionality that scans through a variety of packet 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 13 on page 98](#) provides a high-level illustration of how a classifier works.

Figure 13: How a Classifier Works



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

- Classify packets to a forwarding class and loss priority. The forwarding class determines the output queue. The loss priority is used by schedulers in conjunction with the random early discard (RED) algorithm to control packet discard during periods of congestion.
- Police traffic to a specific bandwidth and burst size. Packets exceeding the policer limits can be discarded, or can be assigned to a different forwarding class, to a different loss priority, or to both.



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

Related Documentation

- [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)
- [Configuring Multifield Classifiers on page 98](#)

Configuring Multifield Classifiers

This topic describes how you configure multifield classifiers.

Multifield classifiers classify packets to a forwarding class and loss priority based on the filter match criteria. Multifield classification is usually done at the edge of the network for packets that do not have valid or trusted behavior aggregate code points.

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



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

To activate (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.



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

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

You configure multifield classifiers by:

1. **Defining the filter**—Configure *either* a firewall filter or a simple filter. Simple filters filter IPv4 traffic (family inet) only. 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
```

Related Documentation

- [Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields on page 97](#)
- [Configuring a Simple Filter](#)
- [Guidelines for Applying Standard Firewall Filters](#)
- [Using Multifield Classifiers to Set Packet Loss Priority on page 101](#)

Using Multifield Classifiers to Set Packet Loss Priority

This topic describes how to use and configure multifield classifiers to set the loss priority 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. When tricolor marking (TCM) is enabled, Juniper Networks M320 Multiservice Edge Routers and T Series Core Routers support four multifield classifier packet loss priority (PLP) designations: **low**, **medium-low**, **medium-high**, and **high**.

To configure the PLP for a multifield classifier, include the **loss-priority** statement in a policer or firewall filter that you configure 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

loss priority (PLP). A multifold 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 199](#).

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

- [Configuring Multifield Classifiers on page 98](#)
- [Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields on page 97](#)

Example: Configuring and Applying a Firewall Filter for a Multifield Classifier

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

- [Requirements on page 103](#)
- [Overview on page 103](#)
- [Configuration on page 105](#)
- [Verification on page 108](#)

Requirements

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

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

Overview

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

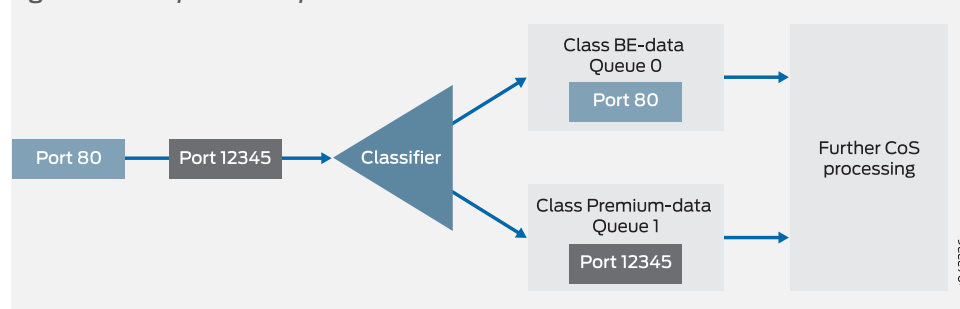
TCP packets with source port 12345 are classified into the Premium-data forwarding class and queue number 1.

Multifield classifiers are typically used at the network edge as packets enter an autonomous system (AS).

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

The classifier operation is shown in [Figure 14 on page 104](#).

Figure 14: Multifield Classifier Based on TCP Source Ports

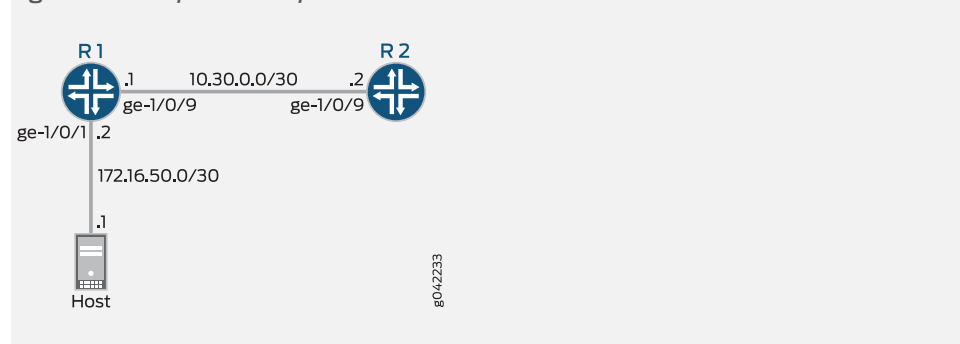


You apply the multifield classifier's firewall filter as an input filter on each customer-facing or host-facing interface that needs the filter. The incoming interface is `ge-1/0/1` on Device R1. The classification and queue assignment is verified on the outgoing interface. The outgoing interface is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

Topology

[Figure 15 on page 104](#) shows the sample network.

Figure 15: Multifield Classifier Scenario



[“CLI Quick Configuration” on page 105](#) shows the configuration for all of the Juniper Networks devices in [Figure 15 on page 104](#).

The section [“Step-by-Step Procedure” on page 105](#) describes the steps on Device R1.

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



Video: [Class of Service Basics, Part 2: Classification Learning Byte](#)

Configuration

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

Device R1

```
set interfaces ge-1/0/1 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 *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

Confirm that the configuration is working properly.

- [Checking the CoS Settings on page 108](#)
- [Sending TCP Traffic into the Network and Monitoring the Queue Placement on page 108](#)

Checking the CoS Settings

Purpose Confirm that the forwarding classes are configured correctly.

Action From Device R1, run the **show class-of-service forwarding-classes** command.

```
user@R1> show class-of-service forwarding-class
```

Forwarding class	ID	Queue	Restricted queue	Fabric
priority Policing priority SPU priority				
BE-data normal low 0	0	0	0	low
Premium-data normal low 1	1	1	1	low
Voice normal low 2	2	2	2	low
NC normal low 3	3	3	3	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: Classifying Packets Based on Their Destination Address

This example shows how to classify packets based on their destination address by using a multifield classifier.

- [Requirements on page 109](#)
- [Overview on page 110](#)
- [Configuration on page 110](#)

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 **so-1/2/2.0** and verify your configuration is attached to the correct interface, issue the **show interfaces filters** command..

Configuration

CLI Quick Configuration To quickly configure the multifield classifier (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 so-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 so-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
```

```
so-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

- [Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields on page 97](#)
- [Configuring Multifield Classifiers on page 98](#)

Example: Configuring and Verifying a Complex Multifield Filter

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

- [Configuring a Complex Multifield Filter on page 112](#)
- [Verifying a Complex Multifield Filter on page 114](#)

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*** 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 the Junos OS. (Ingress queuing is supported on Gigabit Ethernet IQ2 PICs and Enhanced IQ2 PICs only.)

To verify the operation of the multifield filter:

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

CHAPTER 4

Controlling Access to the Network Using Traffic Policing

- [Controlling Network Access Using Traffic Policing Overview on page 117](#)
- [Enabling Tricolor Marking and Limitations of Three-Color Policers on page 123](#)
- [Overview of Tricolor Marking Architecture on page 125](#)
- [Configuring Single-Rate Tricolor Marking on page 126](#)
- [Example: Limiting Inbound Traffic at Your Network Border by Configuring an Ingress Single-Rate Two-Color Policer on page 129](#)
- [Example: Performing CoS at an Egress Network Boundary by Configuring an Egress Single-Rate Two-Color Policer on page 138](#)
- [Configuring Two-Rate Tricolor Marking on page 148](#)
- [Example: Configuring and Verifying Two-Rate Tricolor Marking on page 151](#)
- [Configuring and Applying Tricolor Marking Policers on page 159](#)
- [Example: Limiting Inbound Traffic Within Your Network by Configuring an Ingress Single-Rate Two-Color Policer and Configuring Multifield Classifiers on page 164](#)
- [Example: Limiting Outbound Traffic Within Your Network by Configuring an Egress Single-Rate Two-Color Policer and Configuring Multifield Classifiers on page 178](#)
- [Configuring Policers Based on Logical Interface Bandwidth on page 194](#)
- [Effect of Two-Color Policers on Shaping Rate Changes on page 197](#)

Controlling Network Access Using Traffic Policing Overview

- [Congestion Management for IP Traffic Flows on page 117](#)
- [Traffic Limits on page 118](#)
- [Traffic Color Marking on page 119](#)
- [Forwarding Classes and PLP Levels on page 121](#)
- [Policer Application to Traffic on page 121](#)

Congestion Management for IP Traffic Flows

Traffic policing, also known as *rate limiting*, is an essential component of network access security that is designed to thwart denial-of-service (DoS) attacks. Traffic policing enables

you to control the maximum rate of IP traffic sent or received on an interface and also to partition network traffic into multiple priority levels, also known as *classes of service*. A policer defines a set of traffic rate limits and sets consequences for traffic that does not conform to the configured limits. Packets in a traffic flow that do not conform to traffic limits are either discarded or marked with a different forwarding class or packet loss priority (PLP) level.

With the exception of policers configured to rate-limit aggregate traffic (all protocol families and logical interfaces configured on a physical interface), you can apply a policer to all IP packets in a Layer 2 or Layer 3 traffic flow at a logical interface.

With the exception of policers configured to rate-limit based on physical interface media rate, you can apply a policer to specific IP packets in a Layer 3 traffic flow at a logical interface by using a stateless firewall filter.

You can apply a policer to inbound or outbound interface traffic. Policers applied to inbound traffic help to conserve resources by dropping traffic that does not need to be routed through a network. Dropping inbound traffic also helps to thwart denial-of-service (DoS) attacks. Policers applied to outbound traffic control the bandwidth used.



NOTE: Traffic policers are instantiated on a per-PIC basis. Traffic policing does not work when the traffic for one local policy decision function (L-PDF) subscriber is distributed over multiple Multiservices PICs in an AMS group.

Traffic Limits

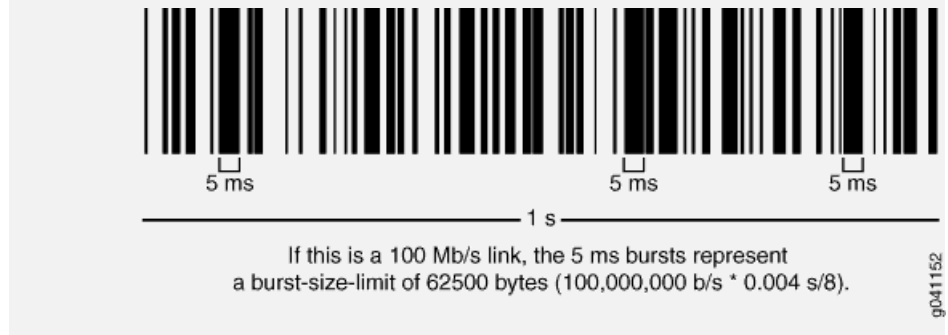
Junos OS policers use a *token bucket algorithm* to enforce a limit on an average transmit or receive rate of traffic at an interface while allowing bursts of traffic up to a maximum value based on the configured bandwidth limit and configured burst size. The token bucket algorithm offers more flexibility than a *leaky bucket algorithm* in that you can allow a specified traffic burst before starting to discard packets or apply a penalty such as packet output-queuing priority or packet-drop priority.

In the token-bucket model, the bucket represents the rate-limiting function of the policer. Tokens are added to the bucket at a fixed rate, but once the specified depth of the bucket is reached, tokens allocated after cannot be stored and used. Each token represents a “credit” for some number of bits, and tokens in the bucket are “cashed in” for the ability to transmit or receive traffic at the interface. When sufficient tokens are present in the bucket, a traffic flow continues unrestricted. Otherwise, packets might be dropped or else re-marked with a lower forwarding class, a higher packet loss priority (PLP) level, or both.

- The rate at which tokens are added to the bucket represents the highest average transmit or receive rate in bits per second allowed for a given service level. You specify this highest average traffic rate as the *bandwidth limit* of the policer. If the traffic arrival rate (or fixed bits-per-second) is so high that at some point insufficient tokens are present in the bucket, then the traffic flow is no longer conforming to the traffic limit. During periods of relatively low traffic (traffic that arrives at or departs from the interface at average rates below the token arrival rate), unused tokens accumulate in the bucket.

- The depth of the bucket in bytes controls the amount of back-to-back bursting allowed. You specify this factor as the *burst-size limit* of the policer. This second limit affects the average transmit or receive rate by limiting the number of bytes permitted in a transmission burst for a given interval of time. Bursts exceeding the current burst-size limit are dropped until there are sufficient tokens available to permit the burst to proceed.

Figure 16: Network Traffic and Burst Rates



As shown in the figure above, a UPC bar code is a good facsimile of what traffic looks like on the line; an interface is either transmitting (bursting at full rate) or it is not. The black lines represent periods of data transmission and the white space represents periods of silence when the token bucket can replenish.

Depending on the type of policer used, packets in a policed traffic flow that surpasses the defined limits might be implicitly set to a higher PLP level, assigned to a configured forwarding class or set to a configured PLP level (or both), or simply discarded. If packets encounter downstream congestion, packets with a **low** PLP level are less likely to be discarded than those with a **medium-low**, **medium-high**, or **high** PLP level.

Traffic Color Marking

Based on the particular set of traffic limits configured, a policer identifies a traffic flow as belonging to one of either two or three categories that are similar to the colors of a traffic light used to control automobile traffic.

- *Single-rate two-color*—A two-color marking policer (or “policer” when used without qualification) meters the traffic stream and classifies packets into two categories of packet loss priority (PLP) according to a configured bandwidth and burst-size limit. You can mark packets that exceed the bandwidth and burst-size limit in some way, or simply discard them.

A policer is most useful for metering traffic at the port (physical interface) level.

- *Single-rate three-color*—This type of policer is defined in RFC 2697, *A Single Rate Three Color Marker*, as part of an assured forwarding (AF) per-hop-behavior (PHB) classification system for a Differentiated Services (DiffServ) environment. This type of policer meters traffic based on the configured committed information rate (CIR), committed burst size (CBS), and the excess burst size (EBS). Traffic is marked as belonging to one of three categories (green, yellow, or red) based on whether the

packets arriving are below the CBS (green), exceed the CBS (yellow) but not the EBS, or exceed the EBS (red).

A single-rate three-color policer is most useful when a service is structured according to packet length and not peak arrival rate.

- *Two-rate three-color*—This type of policer is defined in RFC 2698, *A Two Rate Three Color Marker*, as part of an assured forwarding (AF) per-hop-behavior (PHB) classification system for a Differentiated Services (DiffServ) environment. This type of policer meters traffic based on the configured CIR and peak information rate (PIR), along with their associated burst sizes, the CBS and *peak burst size* (PBS). Traffic is marked as belonging to one of three categories (green, yellow, or red) based on whether the packets arriving are below the CIR (green), exceed the CIR (yellow) but not the PIR, or exceed the PIR (red).

A two-rate three-color policer is most useful when a service is structured according to arrival rates and not necessarily packet length.

Policer actions are implicit or explicit and vary by policer type. The term *Implicit* means that Junos assigns the loss-priority automatically. [Table 15 on page 120](#) 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
	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

Table 15: Policer Actions (continued)

Policer	Marking	Implicit Action	Configurable Action
Two-rate three-color	Green (Conforming)	Assign low loss priority	None
	Yellow (Above the CIR and CBS)	Assign medium-high loss priority	None
	Red (Above the PIR and PBS)	Assign high loss priority	Discard

Forwarding Classes and PLP Levels

A packet's forwarding class assignment and PLP level are used by the Junos OS class of service (CoS) features. The Junos OS CoS features include a set of mechanisms that you can use to provide differentiated services when best-effort traffic delivery is insufficient. For router (and switch) interfaces that carry IPv4, IPv6, and MPLS traffic, you can configure CoS features to take in a single flow of traffic entering at the edge of your network and provide different levels of service across the network—internal forwarding and scheduling (queuing) for output—based on the forwarding class assignments and PLP levels of the individual packets.



NOTE: Forwarding-class or loss-priority assignments performed by a policer or a stateless firewall filter override any such assignments performed on the ingress by the CoS default IP precedence classification at all logical interfaces or by any configured behavior aggregate (BA) classifier that is explicitly mapped to a logical interface.

Based on CoS configurations, packets of a given forwarding class are transmitted through a specific output queue, and each output queue is associated with a transmission service level defined in a *scheduler*.

Based on other CoS configurations, when packets in an output queue encounter congestion, packets with higher loss-priority values are more likely to be dropped by the random early detection (RED) algorithm. Packet loss priority values affect the scheduling of a packet without affecting the packet's relative ordering within the traffic flow.

Policer Application to Traffic

After you have defined and named a policer, it is stored as a template. You can later use the same policer name to provide the same policer configuration each time you want to use it. This eliminates the need to define the same policer values more than once.

You can apply a policer to a traffic flow in either of two ways:

- You can configure a standard stateless firewall filter that specifies the **policer** *policer-name* nonterminating action or the **three-color-policer (single-rate | two-rate)** *policer-name* nonterminating action. When you apply the standard filter to

the input or output at a logical interface, the policer is applied to all packets of the filter-specific protocol family that match the conditions specified in the filter configuration.

With this method of applying a policer, you can define specific classes of traffic on an interface and apply traffic rate-limiting to each class.

- You can apply a policer directly to an interface so that traffic rate-limiting applies to all traffic on that interface, regardless of protocol family or any match conditions.

You can configure policers at the queue, logical interface, or Layer 2 (MAC) level. Only a single policer is applied to a packet at the egress queue, and the search for policers occurs in this order:

- Queue level
- Logical interface level
- Layer 2 (MAC) level

**Related
Documentation**

- *Stateless Firewall Filter Overview.*
- *Traffic Policer Types*
- *Order of Policer and Firewall Filter Operations*
- [Packet Flow Through the Junos OS CoS Process Overview on page 17](#)

Enabling Tricolor Marking and Limitations of Three-Color Policers

This topic describes how to enable TCM on Juniper Networks devices, as well as limitations you need to be aware of when you are using TCM.

Table 16 on page 124 lists the default state for TCM on Juniper Networks devices:

Table 16: Devices Versus TCM

TCM Enabled by Default	TCM Disabled by Default
M120 routers	M320 routers with Enhanced II FPCs
MX Series routers	T Series routers with Enhanced II FPCs
T4000 routers	T640 routers with Enhanced Scaling FPC4s
EX Series switches	T1600 routers with Enhanced Scaling FPC4s



NOTE: If you do not enable TCM on platforms that require it, you cannot configure medium-low or medium-high packet loss priority (PLP) for classifiers, rewrite rules, drop profiles, or firewall filters.



NOTE: On MX Series and M120 routers, you can apply three-color policers to aggregated interfaces.



NOTE: On T Series routers, three-color policers and hierarchical policers are supported on aggregated interfaces if all child links are hosted on Enhanced Scaling FPCs.

TCM has some limitations that must be kept in mind during configuration and operation.

- When you enable TCM on a 10-port Gigabit Ethernet PIC or a 10-Gigabit Ethernet PIC, for queues 6 and 7 only, the output of the **show interfaces queue *interface-name*** command does not display the number of queued bytes and packets, or the number of bytes and packets dropped due to RED. If you do not configure tricolor marking on the interface, these statistics are available for all queues.
- When you enable TCM, Transmission Control Protocol (TCP)-based configurations for drop profiles are rejected. In other words, you cannot include the **protocol** statement at the **[edit class-of-service schedulers *scheduler-name* drop-profile-map]** hierarchy level. The result is that drop profiles are applied to packets with the specified PLP and any protocol type.
- On Gigabit Ethernet IQ PICs, for IEEE 802.1 rewrite rules, only two loss priorities are supported. Exiting packets with medium-high loss priority are treated as high, and packets with medium-low loss priority are treated as low. In other words rewrite rules corresponding to high and low apply instead of those corresponding to medium-high

and medium-low. 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.

- When some PICs with Frame Relay encapsulation mark a packet with high loss priority, the packet is treated as having medium-high loss priority on M320 Multiservice Edge Routers and T Series Core Routers with Enhanced II FPCs and T640 Core Routers with Enhanced Scaling FPC4.
- 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.

To enable TCM:

- Enable two-rate tricolor marking.

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

Related Documentation

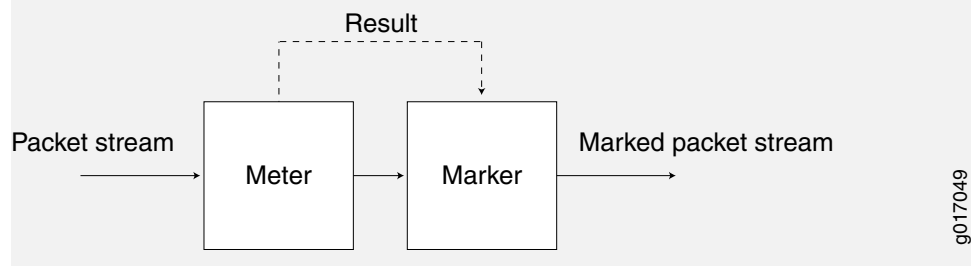
- [Overview of Tricolor Marking Architecture on page 125](#)
- [Configuring and Applying Tricolor Marking Policers on page 159](#)

Overview of Tricolor Marking Architecture

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 17 on page 125](#).

Figure 17: 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 packet loss priority (PLP) field, which has been set by an upstream device as PLP high, medium-high, medium-low, or low; in other words, the PLP field has already been set by a behavior aggregate (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 148](#).

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 and the EBS specifies the maximum burst size in bytes for packets that are 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 53](#) and [“Using Multifield Classifiers to Set Packet Loss Priority” on page 101](#).

- Related Documentation**
- [Enabling Tricolor Marking and Limitations of Three-Color Policers on page 123](#)
 - [Configuring and Applying Tricolor Marking Policers on page 159](#)

Configuring Single-Rate Tricolor Marking

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 Tricolor Marking on page 126](#)
- [Configuring Color-Aware Mode for Single-Rate Tricolor Marking on page 127](#)

Configuring Color-Blind Mode for Single-Rate Tricolor Marking

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 127](#).

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 Tricolor Marking

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 127](#).

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

Table 18: Color-Aware Mode TCM PLP Mapping (continued)

Incoming PLP	Packet Metered Against	Possible Cases	Outgoing PLP
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 according to the type-of-service (ToS) bits in the IP header, 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 according to the ToS bits in the IP header, 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 according to the ToS bits in the IP header, 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 and Applying Tricolor Marking Policers on page 159](#)
 - [Configuring Two-Rate Tricolor Marking on page 148](#)

Example: Limiting Inbound Traffic at Your Network Border by Configuring an Ingress Single-Rate Two-Color Policer

This example shows you how to configure an ingress single-rate two-color policer to filter incoming traffic. The policer enforces the class-of-service (CoS) strategy for in-contract and out-of-contract traffic. You can apply a single-rate two-color policer to incoming packets, outgoing packets, or both. This example applies the policer as an input (ingress) policer. The goal of this topic is to provide you with an introduction to policing by using 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 on page 130](#)
- [Overview on page 130](#)
- [Configuration on page 132](#)
- [Verification on page 137](#)

Requirements

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

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

Overview

Single-rate two-color policing enforces a configured rate of traffic flow for a particular service level by applying implicit or configured actions to traffic that does not conform to the limits. When you apply a single-rate two-color policer to the input or output traffic at an interface, the policer meters the traffic flow to the rate limit defined by the following components:

- **Bandwidth limit**—The average number of bits per second permitted for packets received or transmitted at the interface. You can specify the bandwidth limit as an absolute number of bits per second or as a percentage value from 1 through 100. If a percentage value is specified, the effective bandwidth limit is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.
- **Burst-size limit**—The maximum size permitted for bursts of data. Burst sizes are measured in bytes. We recommend two formulas for calculating burst size:

Burst size = bandwidth x allowable time for burst traffic / 8

Or

Burst size = interface mtu x 10

For information about configuring the burst size, see *Determining Proper Burst Size for Traffic Policers*.



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 18 on page 132](#).

Figure 18: Single-Rate Two-Color Policer Scenario

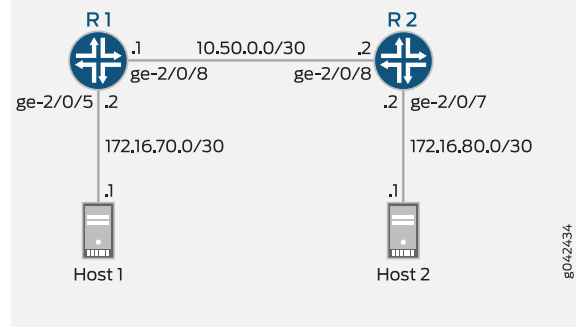
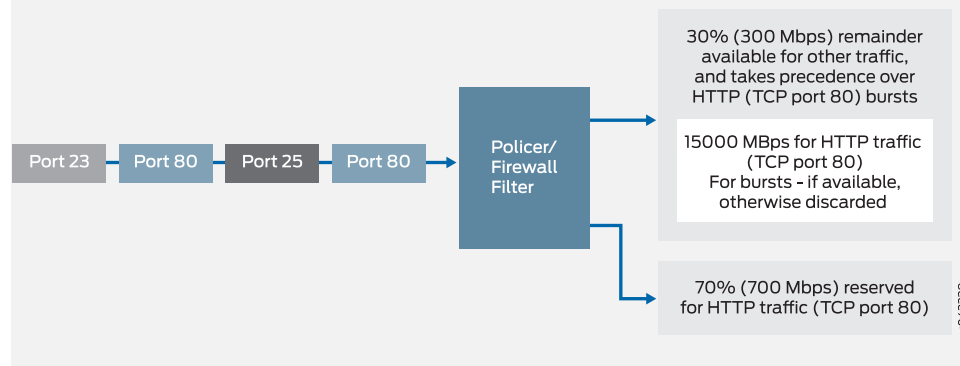


Figure 19 on page 132 shows the policing behavior.

Figure 19: Traffic Limiting in a Single-Rate Two-Color Policer Scenario



Configuration

CLI Quick Configuration

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

Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description 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 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 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
```

**Step-by-Step
Procedure**

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

To configure Device R1:

1. Configure the device interfaces.

```
[edit interfaces]
user@R1# set ge-2/0/5 description to-Host
user@R1# set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set ge-2/0/8 description to-R2
user@R1# set ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set lo0 unit 0 description loopback-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 loopback-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 loopback-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 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 Device R2, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- [Clearing the Counters on page 137](#)
- [Sending TCP Traffic into the Network and Monitoring the Discards on page 137](#)

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.

Related Documentation

- *Junos OS Routing Protocols and Policies Configuration Guide for Security Devices*

Example: Performing CoS at an Egress Network Boundary by Configuring an Egress Single-Rate Two-Color Policer

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 class-of-service (CoS) strategy for in-contract and out-of-contract traffic. You can apply a single-rate two-color policer to incoming packets, outgoing packets, or both. This example applies the policer

as an 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 on page 139](#)
- [Overview on page 139](#)
- [Configuration on page 141](#)
- [Verification on page 146](#)

Requirements

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

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

Overview

Single-rate two-color policing enforces a configured rate of traffic flow for a particular service level by applying implicit or configured actions to traffic that does not conform to the limits. When you apply a single-rate two-color policer to the input or output traffic at an interface, the policer meters the traffic flow to the rate limit defined by the following components:

- **Bandwidth limit**—The average number of bits per second permitted for packets received or transmitted at the interface. You can specify the bandwidth limit as an absolute number of bits per second or as a percentage value from 1 through 100. If a percentage value is specified, the effective bandwidth limit is calculated as a percentage of either the physical interface media rate or the logical interface configured shaping rate.
- **Burst-size limit**—The maximum size permitted for bursts of data. Burst sizes are measured in bytes. We recommend two formulas for calculating burst size:

Burst size = bandwidth x allowable time for burst traffic / 8

Or

Burst size = interface mtu x 10

For information about configuring the burst size, see *Determining Proper Burst Size for Traffic Policers*.



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.

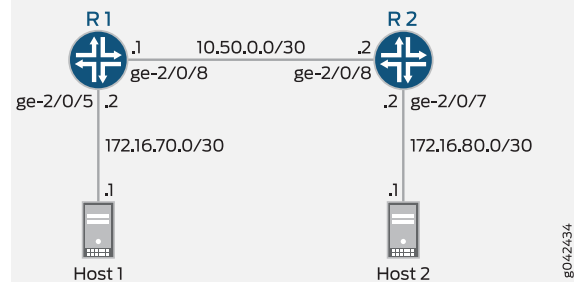


NOTE: You need to leave some additional bandwidth available that is not rate-limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

Topology

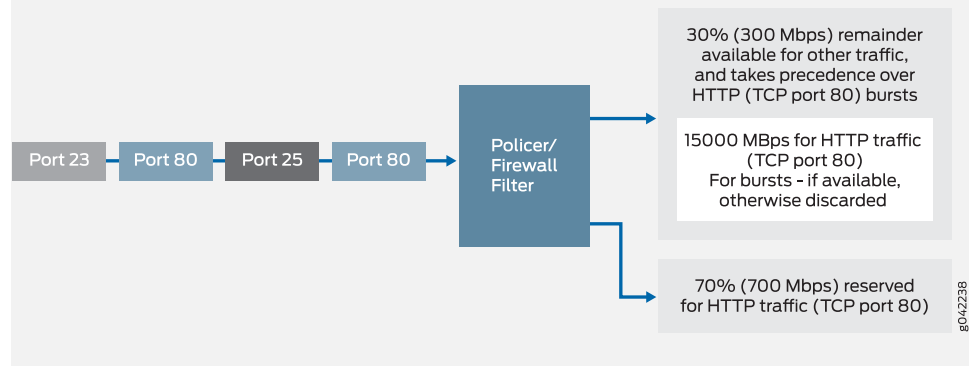
This example uses the topology in [Figure 18 on page 132](#).

Figure 20: Single-Rate Two-Color Policer Scenario



[Figure 19 on page 132](#) shows the policing behavior.

Figure 21: Traffic Limiting in a Single-Rate Two-Color Policer Scenario



Configuration

CLI Quick Configuration

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

Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/8 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 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 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

```

**Step-by-Step
Procedure**

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

To configure Device R1:

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 loopback-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 loopback-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 loopback-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 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 Device R2, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- [Clearing the Counters on page 146](#)
- [Sending TCP Traffic into the Network and Monitoring the Discards on page 146](#)

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 Feature Guide*
- *Example: Configuring a Two-Rate Three-Color Policer*

Configuring Two-Rate Tricolor Marking

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 Tricolor Marking on page 148](#)
- [Configuring Color-Aware Mode for Two-Rate Tricolor Marking on page 149](#)

Configuring Color-Blind Mode for Two-Rate Tricolor Marking

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 148](#).

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* 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 Two-Rate Tricolor Marking

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 149](#).

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
		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 according to the ToS bits in the IP header, 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 according to the ToS bits in the IP header, 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 according to the ToS bits in the IP header, 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

- [Configuring and Applying Tricolor Marking Policers on page 159](#)
- [Configuring Single-Rate Tricolor Marking on page 126](#)

Example: Configuring and Verifying Two-Rate Tricolor Marking

This topic provides several examples of how you can configure and verify two-rate tricolor marking policers and includes the following sections:

- [Requirements on page 151](#)
- [Overview on page 151](#)
- [Configuration on page 151](#)
- [Verification on page 158](#)

Requirements

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

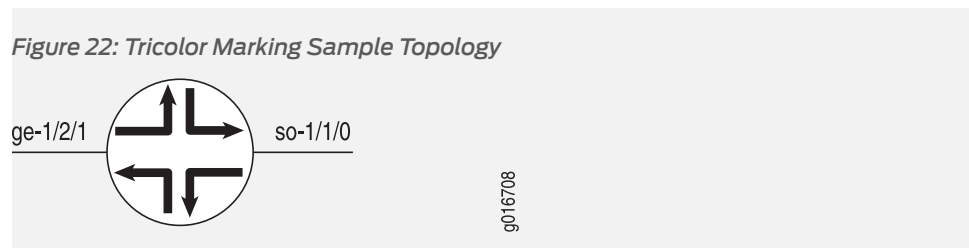
Overview

This example configures a two-rate tricolor marking policer on an input Gigabit Ethernet interface and shows commands to verify its operation.

Traffic enters the Gigabit Ethernet interface and exits a SONET/SDH OC12 interface. Oversubscription occurs when you send line-rate traffic from the Gigabit Ethernet interface out the OC12 interface.

Topology

Figure 22 on page 151 shows the sample topology.



Configuration

To configure two-rate tricolor marking policers, perform these tasks:

- [Example: Applying a Policer to an Input Interface on page 153](#)
- [Example: Applying Profiles to an Output Interface on page 154](#)
- [Example: Marking Packets with Medium-Low Loss Priority on page 156](#)
- [Results on page 157](#)

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 ge-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 so-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 ge-1/2/1 unit 0 family inet filter input 4PLP
set interfaces ge-1/2/1 unit 0 family inet policer input 4PLP
set interfaces ge-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 Gigabit Ethernet 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* in the *CLI User Guide*.

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 ge-1/2/1 unit 0 family inet
user@host# set filter input trtcm-filter
```

4. Confirm the configuration.

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

```
interfaces {
ge-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 OC12 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 so-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* in the *CLI User Guide*.

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 ge-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

Confirm that the configuration is working properly.

- [Verifying Two-Rate Tricolor Marking Operation on page 158](#)

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 Tricolor Marking on page 148](#)
 - [Guidelines for Configuring Firewall Filters](#)

Configuring and Applying Tricolor Marking Policers

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 and includes the following topics:

- [Defining a Tricolor Marking Policer on page 159](#)
- [Applying Tricolor Marking Policers to Firewall Filters on page 161](#)
- [Applying Firewall Filter Tricolor Marking Policers to Interfaces on page 162](#)
- [Example: Configuring and Applying a Single-Rate Tricolor Marking Policer on page 163](#)

Defining a Tricolor Marking Policer

To configure a TCM policer, first enable tricolor marking if not already enabled by default (see [“Enabling Tricolor Marking and Limitations of Three-Color Policers” on page 123](#)):

You can configure a tricolor policer to discard high loss priority traffic on a logical interface in the ingress or egress direction. `statement`.

In all cases, the range of allowable bits-per-second or byte values is 1500 to 100,000,000,000. 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

[Table 21 on page 159](#) describes all the configurable TCM statements.

Table 21: Tricolor Marking 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.	—

Table 21: Tricolor Marking Policer Statements (continued)

Statement	Meaning	Configurable Values
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
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
```

Applying Tricolor Marking Policers to Firewall Filters

To rate-limit traffic by applying a tricolor marking 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;
    }
  }
}
```

Applying Firewall Filter Tricolor Marking Policers to Interfaces

To apply a tricolor marking 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: Configuring and Applying a Single-Rate Tricolor Marking 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 *CLI User Guide*.

This example describes how to configure and apply a color-blind, single-rate, tricolor policer.

1. Configure the single-rate, color-blind, three-color 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 srtcm1-cb
```

3. Apply the **fil** firewall filter to the logical interface:

```
[edit]
user@host# edit interfaces so-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

```

```

so-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 on page 117](#)
 - [Overview of Tricolor Marking Architecture on page 125](#)

Example: Limiting Inbound Traffic Within Your Network by Configuring an Ingress Single-Rate Two-Color Policer and Configuring Multifield Classifiers

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 class-of-service (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 on page 165](#)
- [Overview on page 165](#)
- [Configuration on page 168](#)
- [Verification on page 174](#)

Requirements

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

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

Overview

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.

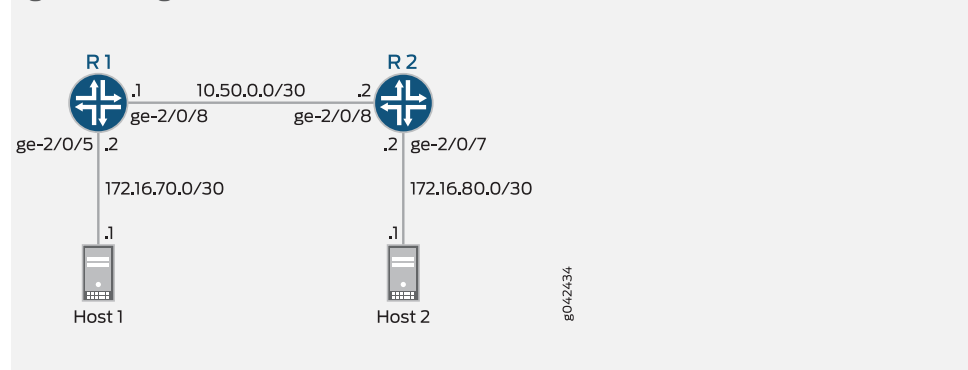


NOTE: You need to leave some additional bandwidth available that is not rate-limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

Topology

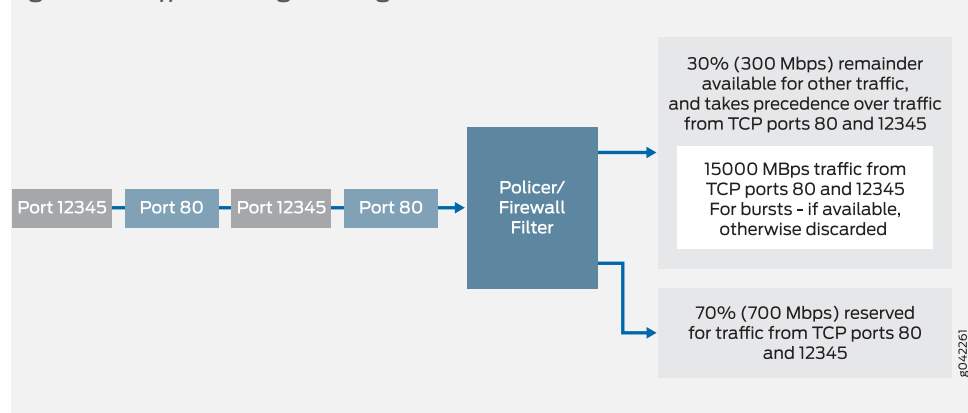
This example uses the topology in [Figure 18 on page 132](#).

Figure 23: Single-Rate Two-Color Policer Scenario



[Figure 19 on page 132](#) shows the policing behavior.

Figure 24: Traffic Limiting in a Single-Rate Two-Color Policer Scenario



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,

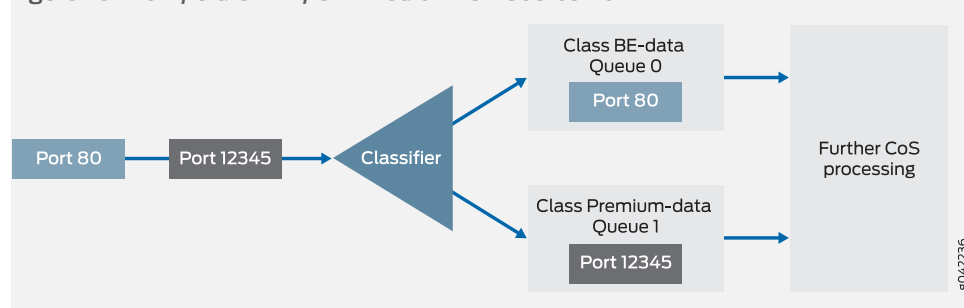
using firewall filter match conditions. The configuration in this example specifies that TCP packets with a source port 80 are classified into the BE-data forwarding class and queue number 0, and TCP packets with a source port 12345 are classified into the Premium-data forwarding class and queue number 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 14 on page 104](#).

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

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 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-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

```

**Step-by-Step
Procedure**

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

To configure Device R1:

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 loopback-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 loopback-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 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;
    }
  }
}
```



```
}
}
```

```
user@R2# show protocols ospf
area 0.0.0.0 {
  interface ge-2/0/7.0 {
    passive;
  }
  interface lo0.0 {
    passive;
  }
  interface ge-2/0/8.0;
}
```

If you are done configuring Device R2, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- [Checking the CoS Settings on page 174](#)
- [Clearing the Counters on page 175](#)
- [Sending Traffic into the Network from TCP HTTP Port 80 and Monitoring the Results on page 175](#)
- [Sending Traffic into the Network from TCP Port 12345 and Monitoring the Results on page 176](#)

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 normal low	0	0	0	low
Premium-data normal low	1	1	1	low
Voice normal low	2	2	2	low
NC normal low	3	3	3	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 be 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.

- 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 be monitored at the policer and custom queue level.

- Action**
- Clear the counters again as shown in section [“Clearing the Counters” on page 175](#).
 - 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
```

- 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.

- 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 Feature Guide*
- *Example: Configuring a Two-Rate Three-Color Policer*

Example: Limiting Outbound Traffic Within Your Network by Configuring an Egress Single-Rate Two-Color Policer and Configuring Multifield Classifiers

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 class-of-service (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 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 on page 178](#)
- [Overview on page 178](#)
- [Configuration on page 182](#)
- [Verification on page 190](#)

Requirements

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

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

Overview

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, the host connected to Device 1 is a traffic generator emulating a webserver. Devices R1, R2, and R3 are owned by a service provider. The webserver is accessed by users behind Device R2. Both hosts are owned by the same customers and their traffic needs to be managed. The host connected to Device 1 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 connected to Device R1) and the service provider that owns Devices R1, R2, and R3 for the web traffic that flows over the link that connects Devices R1 and R2.

The reason that this example is applying the policer as an egress policer between Devices R1 and R2 is because this is the point where the traffic from both customers 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 Devices R1 and R2 by applying the policers as ingress policers on interfaces ge-0/0/0 on Device R3 and ge-2/0/5 on Device R1 would be very 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 the host and Device R3 and the host and Device R1 would result in allowing a maximum throughput of 1400 Mbps over the link between Devices R1 and R2. Therefore, the rate-limiting applied to the host connections between the hosts and Devices R3 and R1 would have to be reduced below 700 Mbps. The calculation of what to reduce the rate-limit number to would be problematic 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 Devices 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 has to go into applying CoS in a network to achieve the desired goals.

In accordance with the contractual bandwidth availability made between the owner of the webserver and the service provider that owns Devices R1, R2 and R3, the egress policer on Device R1 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.

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 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 18 on page 132](#).

Figure 26: Single-Rate Two-Color Policer Scenario

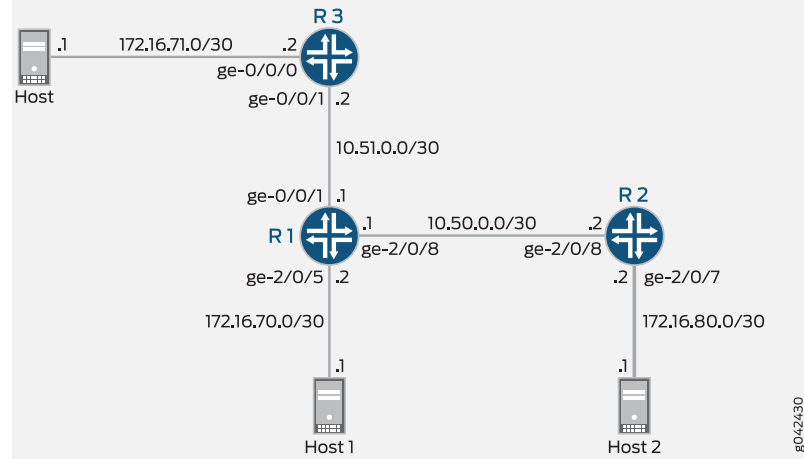
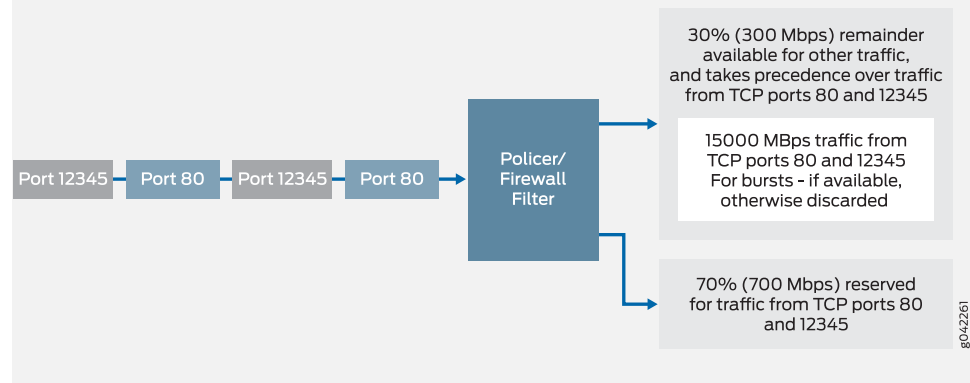


Figure 19 on page 132 shows the policing behavior.

Figure 27: Traffic Limiting in a Single-Rate Two-Color Policer Scenario



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

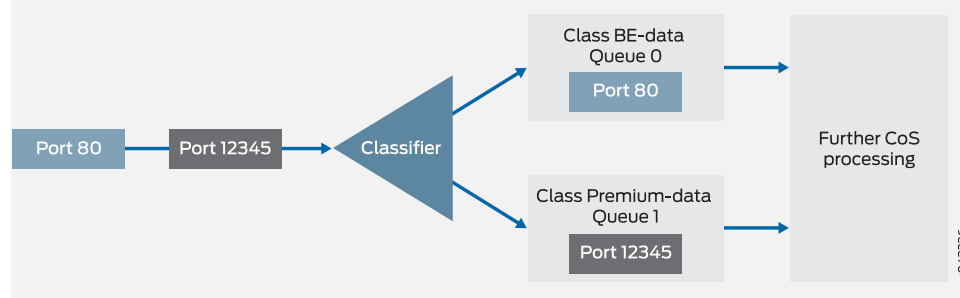
monitored by the policer first. If the traffic makes it through the policer, it is handed off to the outbound interface in the assigned queue for transmission.

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 Device R1. In specifying the custom forwarding classes, you also associate each class with a queue.

The classifier operation is shown in [Figure 14 on page 104](#).

Figure 28: 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 Device R1 by using the **extensive** option in the **show interfaces** command.

Configuration

CLI Quick Configuration

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

Device R1

```
set interfaces ge-0/0/1 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 *CLI User Guide*.

To configure Device 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 Device 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 Device 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 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 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 Device 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 Device R3, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- [Checking the CoS Settings on page 190](#)
- [Clearing the Counters on page 190](#)
- [Sending Traffic into the Network from TCP HTTP Port 80 and Monitoring the Results on page 191](#)
- [Sending Traffic into the Network from TCP Port 12345 and Monitoring the Results on page 192](#)

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 be 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 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                    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 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	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 be monitored at the policer and custom queue level.

- Action**
1. Clear the counters again as shown in section [“Clearing the Counters” on page 175](#).
 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 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	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.

- 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	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 Feature Guide*
- Example: Configuring a Two-Rate Three-Color Policer*

Configuring 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 class-of-service 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 on page 117](#)
- *logical-bandwidth-policer*
- *shaping-rate (Applying to an Interface)*

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

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

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

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

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

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**

- [Configuring Policers Based on Logical Interface Bandwidth on page 194](#)

CHAPTER 5

Defining Forwarding Behavior Based on Forwarding Classes

- [Understanding How Forwarding Classes Assign Classes to Output Queues on page 199](#)
- [Default Forwarding Classes on page 202](#)
- [Configuring a Custom Forwarding Class for Each Queue on page 206](#)
- [Configuring Up to 16 Custom Forwarding Classes on page 208](#)
- [Classifying Packets by Egress Interface on page 215](#)
- [Forwarding Policy Options Overview on page 217](#)
- [Configuring CoS-Based Forwarding on page 219](#)
- [Example: Configuring CoS-Based Forwarding on page 222](#)
- [Example: Configuring CoS-Based Forwarding for Different Traffic Types on page 224](#)
- [Example: Configuring CoS-Based Forwarding for IPv6 on page 225](#)
- [Applying Forwarding Classes to Interfaces on page 226](#)
- [Understanding Queuing and Marking of Host Outbound Traffic on page 227](#)
- [Forwarding Classes and Fabric Priority Queues on page 229](#)
- [Default Routing Engine Protocol Queue Assignments on page 229](#)
- [Assigning Forwarding Class and DSCP Value for Routing Engine-Generated Traffic on page 232](#)
- [Example: Writing Different DSCP and EXP Values in MPLS-Tagged IP Packets on page 233](#)
- [Changing the Default Queuing and Marking of Host Outbound Traffic on page 236](#)
- [Example: Configuring Different Queuing and Marking Defaults for Outbound Routing Engine and Distributed Protocol Handler Traffic on page 236](#)
- [Overriding the Input Classification on page 245](#)

Understanding How Forwarding Classes Assign Classes to Output Queues

This topic covers the following information:

- [Output Queue Assignments Based on Forwarding Class on page 200](#)
- [Devices That Support Up to Four Forwarding Classes on page 200](#)

- [Devices That Support Up to 16 Forwarding Classes on page 201](#)
- [Default and Configurable Packet Loss Priority Values on page 201](#)
- [Configuration Statements Used to Configure and Apply Forwarding Classes on page 201](#)

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 Four Forwarding Classes

Some of the Juniper Networks routing platforms support up to four forwarding classes for classifying customer traffic. On these platforms, you can configure one of each type of default forwarding class. The following Juniper Networks routing platforms support up to four forwarding classes:

- M7i Multiservice Edge Routers with Compact Forwarding Engine Boards (CFEBs)
- M10i Multiservice Edge Routers with CFEBs



NOTE: This list does not reference any Juniper Networks device that has reached its End of Life (EOL) period and its End of Support (EOS) milestone date.

Devices That Support Up to 16 Forwarding Classes

Other Juniper Networks routing platforms 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. On these platforms, the Junos OS 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. The following Juniper Networks routing and switching platforms support up to 16 forwarding classes and up to 8 output queues:

- EX Series switches
- M7i Multiservices Edge Routers with Enhanced Compact Forwarding Engine Boards (CFEB-Es)
- M10i Multiservices Edge Routers with CFEB-Es
- M120 Multiservices Edge Routers
- M320 Multiservices Edge Routers
- MX Series 5G Universal Routing Platforms
- T Series Core Routers
- PTX Packet Transport Routers

Default and Configurable Packet Loss Priority Values

By default, the loss priority is low. On most devices, you can configure high or low loss priority. On the following devices, you can configure high, low, medium-high, or medium-low loss priority:

- M320 routers and T Series routers with Enhanced III Flexible PIC Concentrators (FPCs)
- T640 routers with Enhanced Scaling FPC4s
- PTX Series Packet Transport Routers

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);
}
forwarding-classes-interface-specific forwarding-class-map-name {
  class class-name queue-num queue-number [ restricted-queue queue-number ];
}
interfaces {
  interface-name {
```

```

    unit logical-unit-number {
        forwarding-class class-name;
        forwarding-classes-interface-specific forwarding-class-map-name;
    }
}
restricted-queues {
    forwarding-class class-name queue queue-number;
}

```

Related Documentation

- [Default Forwarding Classes on page 202](#)
- [Configuring a Custom Forwarding Class for Each Queue on page 206](#)
- [Applying Forwarding Classes to Interfaces on page 226](#)
- [Configuring Up to 16 Custom Forwarding Classes on page 208](#)
- [Controlling Network Access Using Traffic Policing Overview on page 117](#)

Default Forwarding Classes

By default, four queues are assigned 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 22 on page 202](#).

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 is applicable to that queue.



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.

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 them to the queues.

Table 22: 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.

Table 22: Default Forwarding Classes (continued)

Queue	Forwarding Class Name	Comments
Queue 1	expedited-forwarding (ef)	<p>The software delivers assured bandwidth, low loss, low delay, and low delay variation (jitter) end-to-end for packets in this service class.</p> <p>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.</p>
Queue 2	assured-forwarding (af)	<p>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.</p> <p>The software accepts excess traffic, but applies a RED drop profile to determine if the excess packets are dropped and not forwarded.</p> <p>Depending on router type, up to four drop probabilities (low, medium-low, medium-high, and high) are defined for this service class.</p>
Queue 3	network-control (nc)	<p>The software delivers packets in this service class with a low priority. (These packets are not delay sensitive.)</p> <p>Typically, these packets represent routing protocol hello or keepalive messages. Because loss of these packets jeopardizes proper network operation, delay is preferable to discard.</p>

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.

This 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**.

On Juniper Networks M320 Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, and T Series Core Routers, 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 208](#).



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.
- For Asynchronous Transfer Mode (ATM) interfaces on Juniper Networks M Series Multiservice Edge Routers, when you use fixed classification with multiple logical interfaces classifying to separate queues, a logical interface without a classifier attached inherits the most recent classifier applied on a different logical interface. For example,

suppose you configure traffic through logical unit 0 to be classified into queue 1, and you configure traffic through logical unit 1 to be classified into queue 3. You want traffic through logical unit 2 to be classified into the default classifier, which is queue 0. In this case, traffic through logical unit 2 is classified into queue 3, because the configuration of logical unit 1 was committed last.

**Related
Documentation**

- [Understanding How Forwarding Classes Assign Classes to Output Queues on page 199](#)
- [Configuring a Custom Forwarding Class for Each Queue on page 206](#)
- [Changing the Default Queuing and Marking of Host Outbound Traffic on page 236](#)
- *CoS Features and Limitations on M Series and T Series Routers*
- *CoS Features and Limitations on PTX Series Routers*

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 208), 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 queue number and forwarding class name.

```
[edit class-of-service forwarding-classes]
user@host# set queue queue-num class-name
```

Related Documentation

- [Understanding How Forwarding Classes Assign Classes to Output Queues on page 199](#)
- [Default Forwarding Classes on page 202](#)
- [Changing the Default Queuing and Marking of Host Outbound Traffic on page 236](#)

Configuring Up to 16 Custom Forwarding Classes

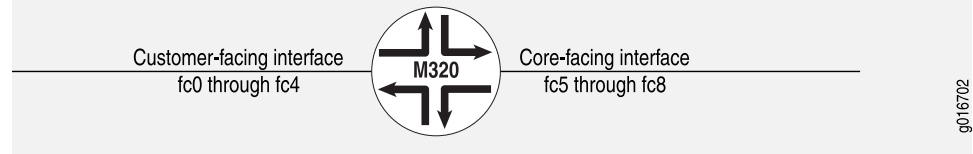
By default on all routers, four forwarding classes are mapped to four output queues, as shown in the topic “[Default Forwarding Classes](#)” on page 202. On M120 and M320 Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, PTX Series Packet Transport Routers, and T Series Core Routers, 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 29 on page 209](#).

Figure 29: Customer-Facing and Core-Facing Forwarding Classes



g016702

In this example, you need nine classifiers and, therefore, nine forwarding classes. The forwarding class-to-queue mapping is shown in [Table 23 on page 209](#).

Table 23: Sample Forwarding Class-to-Queue Mapping

Forwarding Class Names	Queue Number
fc0	0
fc5	
fc1	1
fc6	
fc2	2
fc7	
fc3	3
fc8	
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 to 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 226](#).

For information about configuring eight forwarding classes on ATM2 IQ interfaces, see [Enabling Eight Queues on ATM Interfaces](#).

- [Enabling Eight Queues on Interfaces on page 211](#)
- [Assigning Multiple Forwarding Classes and Default Forwarding Classes on page 212](#)
- [Examples: Configuring Up to 16 Forwarding Classes on page 213](#)

Enabling Eight Queues on Interfaces

By default, Intelligent Queuing (IQ), Intelligent Queuing 2 (IQ2), Intelligent Queuing Enhanced (IQE), and Intelligent Queuing 2 Enhanced (IQ2E) PICs on M320 and T Series 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 IQ PIC can be 4 or 8. If you include the **max-queues-per-interface** statement, all ports on the IQ 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.

This procedure describes how to configure the maximum number of queues the interface supports on a TX Matrix or TX Matrix Plus router.



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 IQ PIC can be 4 or 8. If you include the **max-queues-per-interface** statement, all ports on the IQ PIC use configured mode and all interfaces on the IQ PIC have the same maximum number of queues.

1. To configure a maximum of eight egress queues on these PICs:

```
[edit]
```

```
user@host# edit chassis lcc number 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.

For 4-port OC3c/STM1 Type I and Type II PICs on M320 and T Series routers, when you include the **max-queues-per-interface 8** statement, you can configure up to eight queues on ports 0 and 2. After you commit the configuration, the PIC goes offline and comes back online with only ports 0 and 2 operational. No interfaces can be configured on ports 1 and 3.

For Quad T3 and Quad E3 PICs, when you include the **max-queues-per-interface 8** statement, you can configure up to eight queues on ports 0 and 2. After you commit the configuration, the PIC goes offline and comes back online with only ports 0 and 2 operational. No interfaces can be configured on ports 1 and 3.

When you include the **max-queues-per-interface** statement and commit the configuration, all physical interfaces on the IQ 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 IQ 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 202](#).

Examples: Configuring Up to 16 Forwarding Classes

To configure 16 forwarding classes, map two forwarding classes to each queue. For example:

1. 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 Q0
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 on page 199](#)
 - [Default Forwarding Classes on page 202](#)

Classifying Packets by Egress Interface

For Juniper Networks M320 Multiservice Edge Routers and T Series Core Routers with the Intelligent Queuing (IQ), IQ2, Enhanced IQ (IQE), Multiservices link services intelligent queuing (LSQ) interfaces, or ATM2 PICs, 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 Differentiated Services code point (DSCP) rewrites on the interface. By default, the system does not perform any classification based on the egress interface.

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 in conjunction 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.

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-classes-interface-specific forwarding-class-map-name {
  class class-name queue-num queue-number [ restricted-queue queue-number ];
}

```

For T Series routers that are restricted to only four queues, you can control the queue assignment with the **restricted-queue** option, or you can allow the system to automatically determine the queue in a modular fashion. For example, a map assigning packets to queue 6 would map to queue 2 on a four-queue system.



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 intelligent queuing (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 206](#).

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
```

Forwarding Policy Options Overview

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;
    }
}
}
}

```



NOTE: Beginning with Junos OS Release 17.1R1, QFX10000 Series switches support CoS-based forwarding. **[set class-of-service forwarding-policy class]** is not supported on QFX10000 Series switches.

Beginning with Junos OS Release 17.2, MX routers with MPCs or MS-DPCs, VMX, PTX3000 routers, PTX5000 routers, and VPTX support configuring CoS-based forwarding (CBF) for up to 16 forwarding classes. All other platforms support CBF for up to 8 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. Enabling enhanced-ip is not necessary on PTX routers to support 16 forwarding classes for CBF.

Release History Table

Release	Description
17.2R1	Beginning with Junos OS Release 17.2, MX routers with MPCs or MS-DPCs, VMX, PTX3000 routers, PTX5000 routers, and VPTX support configuring CoS-based forwarding (CBF) for up to 16 forwarding classes.
17.1R1	Beginning with Junos OS Release 17.1R1, QFX10000 Series switches support CoS-based forwarding. [set class-of-service forwarding-policy class] is not supported on QFX10000 Series switches.

Related Documentation

- [Configuring CoS-Based Forwarding on page 219](#)
- [Example: Configuring CoS-Based Forwarding on page 222](#)

Configuring CoS-Based Forwarding

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 Feature Guide*.



NOTE: On M Series routers (except the M120 and M320 routers), forwarding-class-based matching and CBF do not work as expected if the forwarding class has been set with a multifield filter on an input interface.

Beginning with Junos OS Release 17.2, MX routers with MPCs or MS-DPCs, VMX, PTX3000 routers, and PTX5000 routers support configuring CoS-based forwarding (CBF) for up to 16 forwarding classes. All other platforms support CBF for up to 8 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.

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 222](#).

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 OS 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 QN are a set of forwarding classes that map to the specific next hop.

That is, when a packet is switched with Q1 through QN, 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 OS 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.

Release History Table

Release	Description
17.2R1	Beginning with Junos OS Release 17.2, MX routers with MPCs or MS-DPCs, VMX, PTX3000 routers, and PTX5000 routers support configuring CoS-based forwarding (CBF) for up to 16 forwarding classes.

Related Documentation

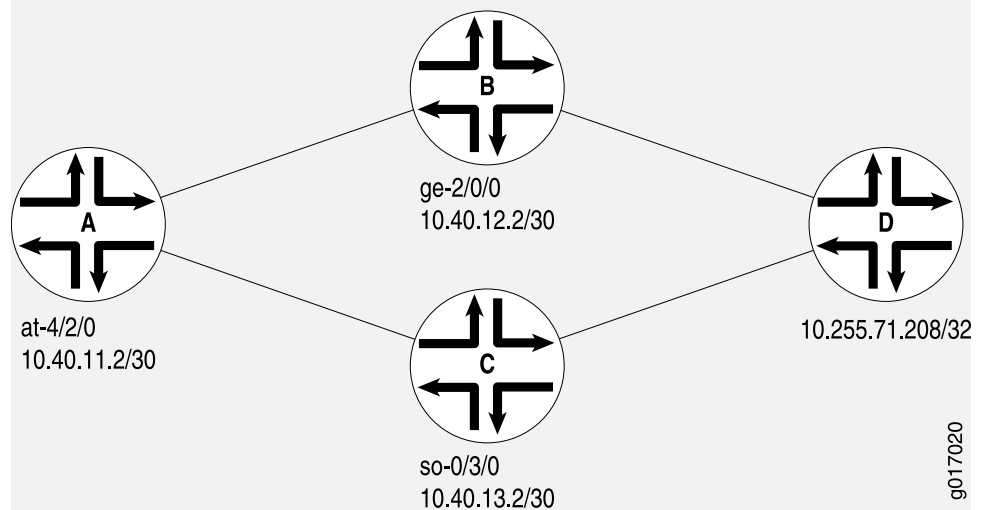
- [Load Balancing VPLS Non-Unicast Traffic Across Member Links of an Aggregate Interface](#)
- [Forwarding Policy Options Overview on page 217](#)

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 30 on page 222](#).

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 30: 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 on page 217](#)

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.



NOTE: On the T4000 router, BA classification and fixed classification are mutually exclusive. That is, only one of the classifications can be configured.

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

This topic covers the following information:

- [Host Outbound Traffic Overview on page 227](#)
- [Default Queuing and Marking of Host Outbound Traffic on page 228](#)
- [Configured Queuing and Marking of Host Outbound Traffic on page 228](#)
- [Configured Queuing and Marking of Outbound Routing Engine Traffic Only on page 228](#)

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.



NOTE: For interfaces on MX80 routers, LACP control traffic is sent through the Routing Engine rather than through 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 229](#).

By default, 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). 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 55](#).

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 hosted on MX Series routers, M120 routers, and Enhanced III FPCs in M320 routers.

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.

Related Documentation

- [Default Routing Engine Protocol Queue Assignments on page 229](#)
- [Default DSCP and DSCP IPv6 Classifiers on page 55](#)
- [Example: Configuring Different Queuing and Marking Defaults for Outbound Routing Engine and Distributed Protocol Handler Traffic on page 236](#)

Forwarding Classes and Fabric Priority Queues

This topic covers the following information:

- [Default Fabric Priority Queuing on page 229](#)
- [Overriding Default Fabric Priority Queuing on page 229](#)

Default Fabric Priority Queuing

On Juniper Networks M320 routers, MX Series routers, T Series routers and EX Series switches only, 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 on page 323](#)

Default Routing Engine Protocol Queue Assignments

[Table 24 on page 229](#) 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.
- 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 24: 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

Table 24: Default Queue Assignments for Packets Generated by the Routing Engine (continued)

Routing Engine Protocol	Default Queue Assignment
Address Resolution Protocol (ARP)	Queue 0
ATM Operation, Administration, and Maintenance (OAM)	Queue 3
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 Maintenance (OAM)	Queue 3
Frame Relay Local Management Interface (LMI)	Queue 3
Frame Relay Asynchronization permanent virtual circuit (PVC)/data link connection identifier (DLCI) status messages	Queue 3
FTP	Queue 0
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

Table 24: Default Queue Assignments for Packets Generated by the Routing Engine (continued)

Routing Engine Protocol	Default Queue Assignment
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
Multicast Source Discovery Protocol (MSDP)	Queue 0
MSDP TCP Retransmission	Queue 3
Multilink Frame Relay Link Integrity Protocol (LIP)	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
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
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

On Juniper Networks M320 Multiservice Edge Routers and T Series Core Routers, 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.

Changing the Default Queuing and Marking of Host Outbound Traffic

You can modify the default queue assignment (forwarding class) and DSCP bits used in the ToS field of *host outbound traffic* (packets generated by the Routing Engine).

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 queue selected is global to the routing device. That is, the traffic is placed in the selected queue on all egress interfaces. In the case of a restricted interface, the Routing Engine sourced traffic flows through the restricted queue.

The queue selected must be properly configured on all interfaces.

To change the default queue 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 queue 3 (network control) with a DSCP value of 101010:

```
[edit class-of-service]
host-outbound-traffic {
  forwarding-class network-control;
  dscp-code-point 101010;
}
```

Related Documentation

- [Understanding How Forwarding Classes Assign Classes to Output Queues on page 199](#)
- [Default Routing Engine Protocol Queue Assignments on page 229](#)
- [Default DSCP and DSCP IPv6 Classifiers on page 55](#)

Example: Configuring Different Queuing and Marking Defaults for Outbound Routing Engine and Distributed Protocol Handler Traffic

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 nondefault 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 keepalive packets generated by the router. This feature is supported for egress interfaces hosted on MX Series routers, M120 routers, and Enhanced III FPCs in M320 routers.

- [Requirements on page 237](#)
- [Overview on page 237](#)
- [Configuration on page 238](#)
- [Verification on page 242](#)

Requirements

This example uses the following hardware and software components:

- Two MX80 routers, R1 and R2, each with a 20-port Gigabit Ethernet MIC with SFP. The two routers are directly connected over an IPv4 network.
- Junos OS Release 13.2 or later.

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 MX80 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 nondefault queues and marked with different nondefault 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 MX80 router in this example has interfaces hosted on a 20-port Gigabit Ethernet MIC with SFP, 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

To configure different queuing and DSCP marking defaults for egress Routing Engine and distributed protocol handler traffic, perform these tasks:

- [Configuring R1 Packet Counting on page 238](#)
- [Configuring R2 Queuing and Re-Marking of Host Outbound Traffic on page 239](#)
- [Configuring R2 Queuing and Re-Marking of Routing Engine Sourced Traffic on page 240](#)

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

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 on page 242](#)
- [Verifying That Router R1 Is Sending BFD Traffic on page 243](#)
- [Verifying That Router R2 Is Receiving BFD Traffic on page 244](#)

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 on page 227](#)
 - [Changing the Default Queuing and Marking of Host Outbound Traffic on page 236](#)
 - *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 Using Schedulers

- [How Schedulers Define Output Queue Properties on page 248](#)
- [Default Schedulers Overview on page 251](#)
- [Configuring Schedulers on page 252](#)
- [Configuring Scheduler Maps on page 252](#)
- [Applying Scheduler Maps Overview on page 253](#)
- [Applying Scheduler Maps to Physical Interfaces on page 254](#)
- [Understanding Interface Sets on page 254](#)
- [Configuring Interface Sets on page 255](#)
- [Interface Set Caveats on page 257](#)
- [Configuring Internal Scheduler Nodes on page 259](#)
- [Example: Configuring and Applying Scheduler Maps on page 260](#)

How Schedulers Define Output Queue Properties

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 on page 249](#)

Queue Scheduling Components

[Table 25 on page 249](#) provides a quick reference to the scheduler components you can configure to determine the bandwidth properties of output queues (forwarding classes).

Table 25: Output Queue Scheduler Components

Output Queue Scheduler Component	Description
Buffer size	Sets the size of the queue buffer.

Table 25: Output Queue Scheduler Components (continued)

Output Queue Scheduler Component	Description
Drop profile map	<p>Maps a drop profile to a packet loss priority. Drop profile map components include:</p> <ul style="list-style-type: none"> 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.
Transmit rate	<p>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.</p> <p>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.</p> <p>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.</p>

Related Documentation

- [Understanding How Forwarding Classes Assign Classes to Output Queues on page 199](#)
- [Default Schedulers Overview on page 251](#)

- [Configuring Schedulers on page 252](#)
- [Priority Scheduling Overview on page 319](#)

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 best effort 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 expedited-forwarding (queue 1) and assured-forwarding (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 expedited-forwarding and assured-forwarding 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 274](#).

The following default scheduler is provided when you install the Junos OS. These settings are not visible in the output of the **show class-of-service** command; rather, they 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 a scheduler by including 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 355](#)
- [Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers on page 351](#)
- [Configuring Scheduler Transmission Rate on page 272](#)
- [Configuring Schedulers for Priority Scheduling on page 322](#)

Configuring Scheduler Maps

After defining a scheduler, you can associate it with a specified forwarding class by including it in a *scheduler map*. To do this, 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, **t3-0/0/0**, **t3-0/0/0:0**, 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 have applied scheduling to one or more of the associated logical interfaces.

Logical interfaces (for example, **t3-0/0/0 unit 0** and **ge-0/0/0 unit 0**) support scheduling on data link connection identifiers (DLCIs) or 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 **t1-0/0/0:0.1**, where the logical unit number is 1.

Although channelized interfaces are generally thought of as logical or virtual, the Junos OS sees T3, T1, and NxDS0 interfaces within a channelized IQ PIC as physical interfaces. For example, both **t3-0/0/0** and **t3-0/0/0:1** are treated as physical interfaces by the Junos OS. In contrast, **t3-0/0/0.2** and **t3-0/0/0:1.2** are considered logical interfaces because they have the **.2** at the end of the interface names.

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 To apply a scheduler map to network traffic, you associate the map with an interface. See the following topics:

- [Applying Scheduler Maps to Physical Interfaces on page 254](#)
- [Applying Scheduler Maps and Shaping Rate to Physical Interfaces on IQ PICs](#)
- [Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs on page 290](#)
- [Oversubscribing Interface Bandwidth on page 263](#)
- [Providing a Guaranteed Minimum Rate on page 275](#)
- [Applying Scheduler Maps to Chassis-Level Queues](#)
- [Forwarding Classes and Fabric Priority Queues on page 229](#)
- [Associating Schedulers with Fabric Priorities on page 323](#)

Applying Scheduler Maps to Physical Interfaces

After you have defined a scheduler map, as described in [“Configuring Scheduler Maps” on page 252](#), 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 290](#).



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.

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 unit1-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 on page 257](#)

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.
  }
}
```

- 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 on page 255](#)

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

This example shows how to configure and apply a scheduler map to a device's interface.

- [Requirements on page 260](#)
- [Overview on page 260](#)
- [Configuration on page 260](#)
- [Verification on page 262](#)

Requirements

Before you begin:

- Create and configure the forwarding classes. See [“Configuring a Custom Forwarding Class for Each Queue” on page 206](#).
- 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 26 on page 260](#).

Table 26: 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

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 *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

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 on page 251](#)
- [Configuring Schedulers on page 252](#)
- [Configuring Scheduler Maps on page 252](#)

CHAPTER 7

Controlling Bandwidth Using Scheduler Rates

- [Oversubscribing Interface Bandwidth on page 263](#)
- [Configuring Scheduler Transmission Rate on page 272](#)
- [Providing a Guaranteed Minimum Rate on page 275](#)
- [PIR-Only and CIR Mode on page 280](#)
- [Excess Rate and Excess Priority Configuration Examples on page 280](#)
- [Controlling Remaining Traffic on page 285](#)
- [Bandwidth Sharing on Nonqueueing Packet Forwarding Engines Overview on page 288](#)
- [Configuring Rate Limits on Nonqueueing Packet Forwarding Engines on page 289](#)
- [Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs on page 290](#)
- [Example: Applying Scheduler Maps and Shaping Rate to DLCIs on page 299](#)
- [Example: Applying Scheduling and Shaping to VLANs on page 303](#)
- [Example: Limiting Egress Traffic on an Interface Using Port Shaping for CoS on page 311](#)

Oversubscribing Interface Bandwidth

The term *oversubscribing interface bandwidth* means configuring shaping rates (peak information rates [PIRs]) so that their sum exceeds the interface bandwidth.

On Channelized IQ PICs, Gigabit Ethernet IQ PICs, and FRF.15 and FRF.16 link services IQ (LSQ) interfaces on Services PICs, Multiservices PICs, and Multiservices DPCs, you can oversubscribe interface bandwidth. This means that the logical interfaces (and DLCIs within an FRF.15 or FRF.16 bundle) can be oversubscribed when there is leftover bandwidth. In the case of FRF.16 bundle interfaces, the physical interface can be oversubscribed. The oversubscription is capped to the configured PIR. Any unused bandwidth is distributed equally among oversubscribed logical interfaces or data-link connection identifiers (DLCIs), 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 290.

When configuring oversubscription for FRF.16 bundle interfaces, you can assign traffic control profiles that apply on a physical interface basis. When you apply traffic control profiles to FRF.16 bundles at the *logical* interface level, member link interface bandwidth is underutilized when there is a small proportion of traffic or no traffic at all on an individual DLCI. Support for traffic control features on the FRF.16 bundle physical interface level addresses this limitation.

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);
```



NOTE: When configuring oversubscription for FRF.16 bundle interfaces on a physical interface basis, you *must* specify **shaping-rate** as a **percentage**.

On LSQ interfaces, you can configure the shaping rate as a percentage from 1 through 100.

On IQ and IQ2 interfaces, you can configure the shaping rate as an absolute rate from 1000 through 6,400,000,000,000 bps.

For all MX Series router and EX Series switch interfaces, the shaping rate can be from 65,535 through 6,400,000,000,000 bps.

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.



NOTE: For channelized and Gigabit Ethernet IQ interfaces, the **shaping-rate** and **guaranteed-rate** statements are mutually exclusive. You cannot configure some logical interfaces to use a shaping rate and others to use a guaranteed rate. This means there are no service guarantees when you configure a PIR. For these interfaces, you can configure either a PIR or a committed information rate (CIR), but not both.

This restriction does not apply to Gigabit Ethernet IQ2 PICs or LSQ interfaces on Multiservices and Services PICs. For LSQ and Gigabit Ethernet IQ2 interfaces, you can configure both a PIR and a CIR on an interface. For more information about CIRs, see [“Providing a Guaranteed Minimum Rate” on page 275](#).

For more information about Gigabit Ethernet IQ2 PICs, see *CoS on Enhanced IQ2 PICs Overview*.

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:



NOTE: When configuring oversubscription for FRF.16 bundle interfaces on a physical interface basis, you *must* specify **delay-buffer-rate** as a percentage.

```
[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.

For LSQ interfaces, if you do not configure a delay-buffer rate, the guaranteed rate (CIR) is used to assign buffers. If you do not configure a guaranteed rate, the shaping rate (PIR) is used in the undersubscribed case, and the scaled shaping rate is used in the oversubscribed case.

On LSQ interfaces, you can configure the delay-buffer rate as a percentage from 1 through 100.

On IQ and IQ2 interfaces, you can configure the delay-buffer rate as an absolute rate from 1000 through 6,400,000,000,000 bps.

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 355](#). For an example showing how the delay-buffer rates are applied, see [“Examples: Oversubscribing Interface Bandwidth” on page 269](#).

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:

$$(\text{remaining delay-buffer rate} * \text{shaping rate}) / (\text{sum of shaping rates})$$

where the remaining delay-buffer rate is equal to:

$$(\text{interface speed}) - (\text{sum of configured delay-buffer rates})$$

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 252](#) and [“Configuring Scheduler Maps” on page 252](#).

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 355](#).

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;
```

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.

6. To enable scheduling for FRF.16 bundles physical interfaces, include the **no-per-unit-scheduler** statement at the **[edit interfaces *interface-name*]** hierarchy level:

```
[edit interfaces interface-name]  
no-per-unit-scheduler;
```

7. 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 27 on page 267](#) shows how the bandwidth and delay buffer are allocated in various configurations.

Table 27: 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.
You do not oversubscribe the interface. You configure a shaping rate at the [edit class-of-service interfaces <i>interface-name</i> unit <i>logical-unit-number</i>] 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 355 . 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.

Table 27: Bandwidth and Delay Buffer Allocations by Configuration Scenario (continued)

Configuration Scenario	Delay Buffer Allocation
You oversubscribe the interface. You configure a shaping rate. You do not configure a guaranteed rate. You do not configure a delay-buffer rate.	<p>Logical interface receives a delay buffer based on the scaled shaping rate:</p> $\text{scaled shaping rate} = (\text{shaping-rate} * [\text{physical interface bandwidth}]) / \text{SUM}(\text{shaping-rates of all logical interfaces on the physical interface})$ <p>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.</p>
You oversubscribe the interface. You configure a shaping rate. You configure a delay-buffer rate.	<p>Logical interface receives a delay buffer based on the delay-buffer rate. For example, on IQ and IQ2 interfaces:</p> <p>delay-buffer-rate <= 10 Mbps: 400-millisecond (ms) delay buffer delay-buffer-rate <= 20 Mbps: 300-ms delay buffer delay-buffer-rate <= 30 Mbps: 200-ms delay buffer delay-buffer-rate <= 40 Mbps: 150-ms delay buffer delay-buffer-rate > 40 Mbps: 100-ms delay buffer</p> <p>On LSQ DLCIs, if total bundle bandwidth < T1 bandwidth:</p> <p>delay-buffer-rate = 1 second</p> <p>On LSQ DLCIs, if total bundle bandwidth >= T1 bandwidth:</p> <p>delay-buffer-rate = 200 ms</p> <p>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 355.</p> <p>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.</p>
You oversubscribe the interface. You do not configure a shaping rate. You configure a guaranteed rate. You configure a delay-buffer rate.	<p>Logical interface receives a delay buffer based on the delay-buffer rate.</p>

Table 27: Bandwidth and Delay Buffer Allocations by Configuration Scenario (continued)

Configuration Scenario	Delay Buffer Allocation
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. This configuration is valid on LSQ interfaces and Gigabit Ethernet IQ2 interfaces only. On channelized interfaces, you cannot configure both a shaping rate (PIR) and a guaranteed rate (CIR).



NOTE: In Junos OS Release 13.3, IP packets with DLCI 0 or 1023 are identified as part of control traffic and routed to the high-priority queue. This oversubscribes the high-priority queue, which is reserved for frame relay control traffic. Oversubscribing the high-priority queue causes the frame relay Local Management Interface (LMI) packets to be dropped.

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***

Examples: Oversubscribing Interface Bandwidth

This section provides two examples: oversubscription of a channelized interface and oversubscription of an LSQ interface.

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 355.

```
chassis {
  fpc 3 {
    pic 0 {
      q-pic-large-buffer;
    }
  }
}
```

```

    }
  }
  interfaces {
    t1-3/0/0 {
      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 {
    t1-3/0/0 {
      unit 0 {
        output-traffic-control-profile tc-profile1;
      }
      unit 1 {
        output-traffic-control-profile tc-profile2;
      }
    }
  }
}

```

Oversubscribing an LSQ Interface with Scheduling Based on the Logical Interface

Apply a traffic-control profile to a logical interface representing a DLCI on an FRF.16 bundle:

```

  interfaces {
    lsq-1/3/0:0 {
      per-unit-scheduler;
      unit 0 {
        dlci 100;
      }
      unit 1 {
        dlci 200;
      }
    }
  }

  class-of-service {
    traffic-control-profiles {
      tc_0 {
        shaping-rate percent 100;
        guaranteed-rate percent 60;
        delay-buffer-rate percent 80;
      }
    }
  }

```

```

    }
    tc_1 {
        shaping-rate percent 80;
        guaranteed-rate percent 40;
    }
}
interfaces {
    lsq-1/3/0 {
        unit 0 {
            output-traffic-control-profile tc_0;
        }
        unit 1 {
            output-traffic-control-profile tc_1;
        }
    }
}
}

```

Oversubscribing an LSQ Interface with Scheduling Based on the Physical Interface

Apply a traffic-control profile to the physical interface representing an FRF.16 bundle:

```

interfaces {
    lsq-0/2/0:0 {
        no-per-unit-scheduler;
        encapsulation multilink-frame-relay-uni-nni;
        unit 0 {
            dlci 100;
            family inet {
                address 10.18.18.2/24;
            }
        }
    }
}
class-of-service {
    traffic-control-profiles {
        rlsq_tc {
            scheduler-map rlsq;
            shaping-rate percent 60;
            delay-buffer-rate percent 10;
        }
    }
    interfaces {
        lsq-0/2/0:0 {
            output-traffic-control-profile rlsq_tc;
        }
    }
}
scheduler-maps {
    rlsq {
        forwarding-class best-effort scheduler rlsq_scheduler;
        forwarding-class expedited-forwarding scheduler rlsq_scheduler1;
    }
}
schedulers {
    rlsq_scheduler {

```

```

    transmit-rate percent 20;
    priority low;
  }
  rlsq_scheduler1 {
    transmit-rate percent 40;
    priority high;
  }
}

```

On an FRF.15 bundle, apply the following configuration:

```

class-of-service {
  traffic-control-profiles {
    rlsq {
      scheduler-map sched_0;
      shaping-rate percent 40;
      delay-buffer-rate percent 50;
    }
  }
  interfaces lsq-2/0/0 {
    unit 0 {
      output-traffic-control-profile rlsq;
    }
  }
}
interfaces lsq-2/0/0 {
  per-unit-scheduler;
  unit 0 {
    encapsulation multilink-frame-relay-end-to-end;
    family inet {
      address 10.1.1.2/32;
    }
  }
}

```

Configuring Scheduler Transmission Rate

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.

On M Series routers other than the M120 and M320 routers, you should not configure a **buffer-size** larger than the **transmit-rate** for a rate-limited queue in a scheduler. If you do, the Packet Forwarding Engine will reject the CoS configuration. However, you can achieve

the same effect by removing the **exact** option from the transmit rate or specifying the buffer size using the **temporal** option.



NOTE: For 8-port, 12-port, and 48-port Fast Ethernet PICs, transmission scheduling is not supported.

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. For all MX Series router and EX Series switch interfaces, the rate can be from 65,535 through 6,400,000,000,000 bps. On all other platforms, the rate can be from 3200 through 6,400,000,000,000 bps.
- **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.



NOTE:

- Including the **exact** option is not supported on Enhanced Queuing Dense Port Concentrators (DPCs) on Juniper Network MX Series 5G Universal Routing Platforms.
- The configuration of the **transmit-rate percent 0 exact** statement at the **[edit class-of-service schedulers *scheduler-name*]** hierarchy is ineffective on T4000 routers with Type 5 FPCs.

- **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. On IQE PICs, the **rate-limit** option for the schedulers' transmit rate is implemented as a static policer. Therefore, these schedulers are not aware of congestion and the maximum rate possible on these schedulers is limited by the value specified in the **transmit-rate** statement. Even if there is no congestion, the queue cannot send traffic above the transmit rate due to the static policer.



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 252](#).

For more information about scheduler transmission rate, see the following sections:

- [Example: Configuring Scheduler Transmission Rate on page 274](#)
- [Allocation of Leftover Bandwidth on page 274](#)

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.

Juniper Networks M Series Multiservice Edge Routers and T Series Core Routers do not distribute leftover bandwidth in proportion to the configured transmit rate of the queues. Instead, the scheduler distributes the leftover bandwidth equally in round-robin fashion to queues that have negative bandwidth credit. All negative-credit queues can take the leftover bandwidth in equal share. This description suggests a simple round-robin distribution process among the queues with negative credits. In actual operation, a queue might change its bandwidth credit status from positive to negative and from negative to positive instantly while the leftover bandwidth is being distributed. Lower-rate queues tend to be allocated a larger share of leftover bandwidth, because their bandwidth credit is more likely to be negative at any given time, if they are overdriven persistently. Also, if there is a large packet size difference, (for example, queue 0 receives 64-byte packets, whereas queue 1 receives 1500-byte packets), then the actual leftover bandwidth distribution ratio can be skewed substantially, because each round-robin turn allows exactly one packet to be transmitted by a negative-credit queue, regardless of the packet size.

By default, on MX Series routers, the M320 Enhanced Type 4 FPCs, and T4000 routers with Type 5 FPCs and EX Series switches, 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, M Series and T Series routers distribute leftover bandwidth in equal shares for the queues with the same priority and same negative-credit status. MX Series routers and M320 Enhanced Type 4 FPCs, and EX Series switches, share excess bandwidth in the ratio of the transmit rates, but you can adjust this distribution.

Related Documentation

- [Configuring Schedulers for Priority Scheduling on page 322](#)
- [How Schedulers Define Output Queue Properties on page 248](#)
- [Configuring a Scheduler on page 476](#)
- *excess-rate*
- *schedulers*

Providing a Guaranteed Minimum Rate

On Gigabit Ethernet IQ PIC, EQ DPC, MIC, MPC, and Channelized IQ PIC interfaces, and on FRF.16 LSQ interfaces on Multiservices and Services PICs, 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.

On IQ and IQ2 interfaces, you can configure the guaranteed rate as an absolute rate from 1000 through 6,400,000,000,000 bps.



NOTE: For channelized and Gigabit Ethernet IQ interfaces, the **shaping-rate** and **guaranteed-rate** statements are mutually exclusive. You cannot configure some logical interfaces to use a shaping rate and others to use a guaranteed rate. This means there are no service guarantees when you configure a PIR. For these interfaces, you can configure either a PIR or a CIR, but not both.

This restriction does not apply to Gigabit Ethernet IQ2 PICs or LSQ interfaces on Multiservices and Services PICs. For LSQ and Gigabit Ethernet IQ2 interfaces, you can configure both a PIR and a CIR on an interface.

For more information about Gigabit Ethernet IQ2 PICs, see *CoS on Enhanced IQ2 PICs Overview*.

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.

On IQ and IQ2 interfaces, you can configure the delay-buffer rate as an absolute rate from 1000 through 6,400,000,000,000 bps.

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 355](#). For an example showing how the delay-buffer rates are applied, see [“Example: Providing a Guaranteed Minimum Rate” on page 279](#).

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 252](#) and [“Configuring Scheduler Maps” on page 252](#).

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 355](#).

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;
```

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.

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 28 on page 278](#) shows how the bandwidth and delay buffer are allocated in various configurations.

Table 28: 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 355 .
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 355 .

Verifying Configuration of Guaranteed Minimum Rate

To verify your configuration, you can issue the 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:

$$\text{delay-buffer-rate} < [8 \times 64 \text{ Kbps}]: 2 \text{ seconds of delay-buffer-rate}$$

For more information about this calculation, see [“Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size” on page 355](#).

```
chassis {
  fpc 3 {
    pic 0 {
      q-pic-large-buffer;
    }
  }
}
interfaces {
  t1-3/0/1 {
    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 {
  t1-3/0/1 {
    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).

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 29 on page 281](#). 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, there is no remaining excess bandwidth for the low-priority queues.

Table 29: 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 30 on page 281](#). 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 30: 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 31 on page 281](#). 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 31: 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 32 on page 282](#). 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 32: 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 33 on page 282](#). 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 33: 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
Q3	5%	20%	low	low	500 Mbps	50 + 0 = 50 Mbps

The behavior with the same (high) priority levels is shown in [Table 34 on page 283](#). 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 34: 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 35 on page 283](#). 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 35: 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
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 36 on page 283](#). 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 36: 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 37 on page 284](#). 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 37: 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;
    }
  }
}
```

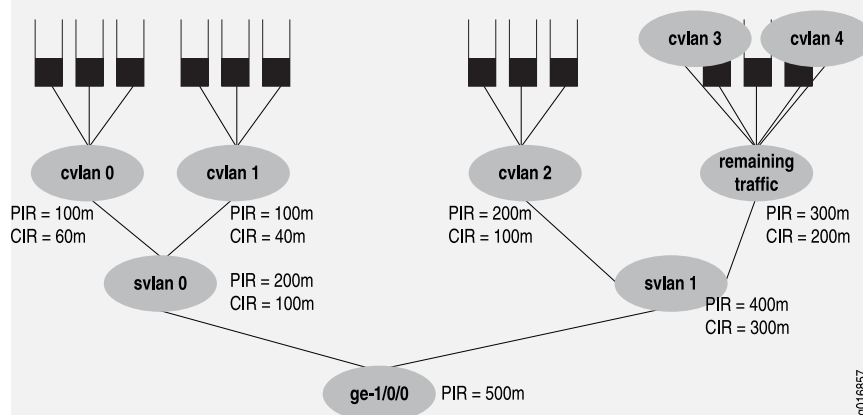
Controlling Remaining Traffic

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 31 on page 286](#). 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 is to configure and apply a traffic control profile for all remaining traffic on the interface.

Figure 31: Handling Remaining Traffic



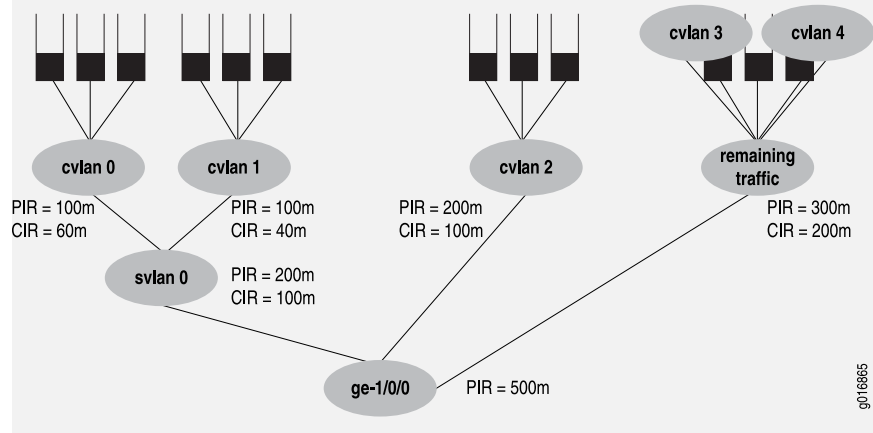
This example considers the case where customer VLANs 3 and 4 have no explicit traffic control profile, yet need to establish a shaping and guaranteed transmit rate for traffic heading for those customer VLANs. The solution is to add a traffic control profile to the **svlan1** interface set. This example 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 32 on page 287](#).

Figure 32: 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-if1** 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 Nonqueueing 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
- M120 Multiservice Edge Router (rate limit and excess priority only; excess rate is not configured by the user)
- M320 router with Enhanced FPCs (rate limit, excess rate, and excess priority)
- MX Series 5G Universal Routing Platform with nonqueueing 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 such as the IQ, IQ2, or IQE PICs, or MX Series routers queuing DPCs or EX Series switches, 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 178](#) 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 DLCIs 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 (DLCI) 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.

You can configure logical interface scheduling on the following PICs:

- Multiservices and Services PICs , on link services IQ (**lsq-**) interfaces
- Channelized E1 IQ PIC
- Channelized OC3 IQ PIC
- Channelized OC12 IQ PIC (Per-unit scheduling is not supported on T1 interfaces configured on this PIC.)
- Channelized STM1 IQ PIC
- Channelized T3 IQ PIC
- E3 IQ PIC
- Gigabit Ethernet IQ PIC
- Gigabit Ethernet IQ2 PIC
- IQE PICs

You can configure logical interface scheduling on the following MICs and MPCs as well as any MPC that contains a queuing chip:

- 16x10GE MPC
- MPC3E:
 - 2x10GE MIC with XFP
 - 10x10GE MIC with SFP+
 - 2x40GE MIC with QSFP+
 - 1x100GE MIC with CXP

- MPC4E:
 - 32x10GE with SFPP
 - 2x100GE + 8x10GE with SFPP
- MPC6E:
 - 24x10GE MIC with SFPP
 - 24x10GE MIC with SFPP OTN
 - 2x100GE MIC with CFP2 OTN
 - 4x100GE MIC with CXP

For Channelized and Gigabit Ethernet IQ PICs only, you can configure a shaping rate for a VLAN or DLCI 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 263](#).

Physical interfaces (for example, **t3-0/0/0**, **t3-0/0/0:0**, 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.

For Gigabit Ethernet IQ2 PIC PICs only, you can configure hierarchical traffic shaping, meaning the shaping is performed on both the physical interface and the logical interface. You can also configure input traffic scheduling and shared scheduling. For more information, see *CoS on Enhanced IQ2 PICs Overview*.

Logical interfaces (for example, **t3-0/0/0.0**, **ge-0/0/0.0**, and **t1-0/0/0:0.1**) support scheduling on DLCIs or VLANs only. Furthermore, logical interface scheduling is not supported on PICs that do not have IQ.



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 ge-0/0/0.1 or t1-0/0/0:0.1, where the logical unit number is 1.

Although channelized interfaces are generally thought of as logical or virtual, the Junos OS sees T3, T1, and NxDS0 interfaces within a channelized IQ PIC as physical interfaces. For example, both t3-0/0/0 and t3-0/0/0:1 are treated as physical interfaces by the Junos OS. In contrast, t3-0/0/0.2 and t3-0/0/0:1.2 are considered logical interfaces because they have the .2 at the end of the interface names.

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
```

Table 38 on page 293 shows the interfaces/PICs that support fine-grained queuing and scheduling.

Table 38: Fine-Grained Queuing and Scheduling Support by Interface or PIC Type

Interface Type	PIC Type	Supported	Example Configuration
IQ PICs			
Physical interfaces	ATM2 IQ	Yes	Example of supported configuration: <pre>[edit class-of-service interfaces at-0/0/0] scheduler-map map-1;</pre>
Channelized interfaces configured on IQ PICs	Channelized DS3 IQ	Yes	Example of supported configuration: <pre>[edit class-of-service interfaces t1-0/0/0:1] scheduler-map map-1;</pre>

Table 38: Fine-Grained Queuing and Scheduling Support by Interface or PIC Type (continued)

Interface Type	PIC Type	Supported	Example Configuration
Logical interfaces (DLCIs and VLANs only) configured on IQ PICs	Gigabit Ethernet IQ with VLAN tagging enabled	Yes	Example of supported configuration: [edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;
	E3 IQ with Frame Relay encapsulation	Yes	Example of supported configuration: [edit class-of-service interfaces e3-0/0/0 unit 1] scheduler-map map-1;
	Channelized OC3 IQ with Frame Relay encapsulation	Yes	Example of supported configuration: [edit class-of-service interfaces t1-1/0/0:1 unit 0] scheduler-map map-1;
	Channelized STM1 IQ with Frame Relay encapsulation	Yes	Example of supported configuration: [edit class-of-service interfaces e1-0/0/0:1 unit 1] scheduler-map map-1;
	Channelized T3 IQ with Frame Relay encapsulation	Yes	Example of supported configuration: [edit class-of-service interfaces t1-0/0/0 unit 1] scheduler-map map-1;
Logical interfaces configured on IQ PICs (interfaces that are not DLCIs or VLANs)	E3 IQ PIC with Cisco HDLC encapsulation	No	No
	ATM2 IQ PIC with LLC/SNAP encapsulation	No	No
	Channelized OC12 IQ PIC with PPP encapsulation	No	No
Non-IQ PICs			
Physical interfaces	T3	Yes	Example of supported configuration: [edit class-of-service interfaces t3-0/0/0] scheduler-map map-1;

Table 38: Fine-Grained Queuing and Scheduling Support by Interface or PIC Type (continued)

Interface Type	PIC Type	Supported	Example Configuration
Channelized OC12 PIC	Channelized OC12	Yes	Example of supported configuration: [edit class-of-service interfaces t3-0/0/0:1] scheduler-map map-1;
Channelized interfaces (except the Channelized OC12 PIC)	Channelized STM1	No	No
Logical interfaces	Fast Ethernet	No	No
	Gigabit Ethernet	No	No
	ATM1	No	No
	Channelized OC12	No	No

Table 39 on page 295 shows the MICs and MPCs that support fine-grained queuing and scheduling.

Table 39: Fine-Grained Queuing and Scheduling Support by MIC or MPC Type

MPC	MIC	Supported	Example Configuration
Fixed Configuration MPCs			
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;

Table 39: Fine-Grained Queuing and Scheduling Support by MIC or MPC Type (continued)

MPC	MIC	Supported	Example Configuration
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
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 39: 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: <pre>[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;</pre>
MPC2E P	No	No	No
MPC3E	10-Gigabit Ethernet MIC with SFP+	Yes	Example of supported configuration: <pre>[edit class-of-service interfaces xe-0/0/0 unit 1] scheduler-map map-1;</pre>
	40-Gigabit Ethernet MIC with QSFP+	Yes	Example of supported configuration: <pre>[edit class-of-service interfaces et-0/0/0 unit 1] scheduler-map map-1;</pre>
	100-Gigabit Ethernet MIC with CXP	Yes	Example of supported configuration: <pre>[edit class-of-service interfaces et-0/0/0 unit 1] scheduler-map map-1;</pre>
MPC6E	Any supported MIC	Yes	Example of supported configuration: <pre>[edit class-of-service interfaces et-0/0/0 unit 1] scheduler-map map-1;</pre>

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* for scaling information on non-queuing MPCs.

- 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;
```

- 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). The range is from 1000 through 6,400,000,000,000 bps. For the IQ2 Gigabit Ethernet PIC, the minimum is 80,000 bps, and for the IQ2 10 Gigabit Ethernet PIC, the minimum is 160,000 bps. For the 16x10GE MPC, the minimum is 250,000 bps, and for the MPC3E, MPC4E, and MPC6E, the minimum is 292,000 bps.

For FRF.16 bundles on link services interfaces, only shaping rates based on percentage are supported.

- Related Documentation**
- [per-unit-scheduler](#)
 - [Example: Applying Scheduling and Shaping to VLANs on page 303](#)
 - [Example: Applying Scheduler Maps and Shaping Rate to DLCIs on page 299](#)

Example: Applying Scheduler Maps and Shaping Rate to DLCIs

This example shows how to apply scheduler maps and shaping rates to individual logical interfaces.

- [Requirements on page 299](#)
- [Overview on page 299](#)
- [Configuration on page 300](#)

Requirements

This example uses the following hardware and software components:

- Junos OS Release 7.4 or later running on router line cards that support Intelligent Queuing (IQ).
- Junos OS Release 13.2 or later running on MX Series routers containing 16x10GE MPC or MPC3E line cards.
- Junos OS Release 13.3 or later running on MX Series routers containing MPC4E line cards.
- Junos OS Release 15.1 or later running on MX Series routers containing MPC6E line cards.

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.

This example shows how to define schedulers for logical interfaces through the direct use of scheduler maps and shaping rates.

In this example, we associate the scheduler **sched-map-logical-0** with logical interface **unit 0** on physical interface **t3-1/0/0**, and allocate 10 Mbps of transmission bandwidth to the logical interface. We also associate the scheduler **sched-map-logical-1** with logical interface **unit 1** on the same physical interface, **t3-1/0/0**, and allocate 20 Mbps of transmission bandwidth to the logical interface.

The allocated bandwidth is shared among the individual forwarding classes in the scheduler map. Although these schedulers are configured on a single physical interface, they are independent from each other. Traffic on one logical interface unit does not affect the transmission priority, bandwidth allocation, or drop behavior on the other logical interface unit.

For a similar example, see [“Example: Applying Scheduling and Shaping to VLANs” on page 303](#).

Configuration

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

```
set interfaces t3-1/0/0:1 per-unit-scheduler
set interfaces t3-1/0/0:1 encapsulation frame-relay
set interfaces t3-1/0/0:1 unit 0 dlci 100
set interfaces t3-1/0/0:1 unit 0 family inet address 10.1.1.0/24
set interfaces t3-1/0/0:1 unit 1 dlci 101
set interfaces t3-1/0/0:1 unit 1 family inet address 10.1.1.1/24
set class-of-service interfaces t3-1/0/0:1 unit 0 scheduler-map sched-map-logical-0
set class-of-service interfaces t3-1/0/0:1 unit 0 shaping-rate 10m
set class-of-service interfaces t3-1/0/0:1 unit 1 scheduler-map sched-map-logical-1
set class-of-service interfaces t3-1/0/0:1 unit 1 shaping-rate 20m
set class-of-service scheduler-maps sched-map-logical-0 forwarding-class best-effort
scheduler sched-best-effort-0
set class-of-service scheduler-maps sched-map-logical-0 forwarding-class
assured-forwarding scheduler sched-bronze-0
set class-of-service scheduler-maps sched-map-logical-0 forwarding-class
expedited-forwarding scheduler sched-silver-0
set class-of-service scheduler-maps sched-map-logical-0 forwarding-class
network-control scheduler sched-gold-0
set class-of-service scheduler-maps sched-map-logical-1 forwarding-class best-effort
scheduler sched-best-effort-1
set class-of-service scheduler-maps sched-map-logical-1 forwarding-class
assured-forwarding scheduler sched-bronze-1
set class-of-service scheduler-maps sched-map-logical-1 forwarding-class
expedited-forwarding scheduler sched-silver-1
set class-of-service scheduler-maps sched-map-logical-1 forwarding-class
network-control scheduler sched-gold-1
set class-of-service schedulers sched-best-effort-0 transmit-rate 4m
set class-of-service schedulers sched-bronze-0 transmit-rate 3m
set class-of-service schedulers sched-silver-0 transmit-rate 2m
set class-of-service schedulers sched-gold-0 transmit-rate 1m
set class-of-service schedulers sched-best-effort-1 transmit-rate 8m
set class-of-service schedulers sched-bronze-1 transmit-rate 6m
set class-of-service schedulers sched-silver-1 transmit-rate 4m
set class-of-service schedulers sched-gold-1 transmit-rate 2m
```

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 *CLI User Guide*.

1. Configure the device interfaces.

```
[edit interfaces]
```

```

user@PE1# set t3-1/0/0:1 per-unit-scheduler
user@PE1# set t3-1/0/0:1 encapsulation frame-relay
user@PE1# set t3-1/0/0:1 unit 0 dlc1 100
user@PE1# set t3-1/0/0:1 unit 0 family inet address 10.1.1.0/24
user@PE1# set t3-1/0/0:1 unit 1 dlc1 101
user@PE1# set t3-1/0/0:1 unit 1 family inet address 10.1.1.1/24

```

2. Define the schedulers.

```

[edit class-of-service]
user@PE1# set schedulers sched-best-effort-0 transmit-rate 4m
user@PE1# set schedulers sched-bronze-0 transmit-rate 3m
user@PE1# set schedulers sched-silver-0 transmit-rate 2m
user@PE1# set schedulers sched-gold-0 transmit-rate 1m
user@PE1# set schedulers sched-best-effort-1 transmit-rate 8m
user@PE1# set schedulers sched-bronze-1 transmit-rate 6m
user@PE1# set schedulers sched-silver-1 transmit-rate 4m
user@PE1# set schedulers sched-gold-1 transmit-rate 2m

```

3. Define the scheduler maps.

```

[edit class-of-service]
user@PE1# set scheduler-maps sched-map-logical-0 forwarding-class best-effort
scheduler sched-best-effort-0
user@PE1# set scheduler-maps sched-map-logical-0 forwarding-class
assured-forwarding scheduler sched-bronze-0
user@PE1# set scheduler-maps sched-map-logical-0 forwarding-class
expedited-forwarding scheduler sched-silver-0
user@PE1# set scheduler-maps sched-map-logical-0 forwarding-class
network-control scheduler sched-gold-0
user@PE1# set scheduler-maps sched-map-logical-1 forwarding-class best-effort
scheduler sched-best-effort-1
user@PE1# set scheduler-maps sched-map-logical-1 forwarding-class
assured-forwarding scheduler sched-bronze-1
user@PE1# set scheduler-maps sched-map-logical-1 forwarding-class
expedited-forwarding scheduler sched-silver-1
user@PE1# set scheduler-maps sched-map-logical-1 forwarding-class
network-control scheduler sched-gold-1

```

4. Apply the scheduler maps and shaping rates to the logical interfaces.

```

[edit class-of-service]
user@PE1# set interfaces t3-1/0/0:1 unit 0 scheduler-map sched-map-logical-0
user@PE1# set interfaces t3-1/0/0:1 unit 0 shaping-rate 10m
user@PE1# set interfaces t3-1/0/0:1 unit 1 scheduler-map sched-map-logical-1
user@PE1# set interfaces t3-1/0/0:1 unit 1 shaping-rate 20m

```

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.

```
[edit interfaces]
user@PE1# show
t3-1/0/0:1 {
    encapsulation frame-relay;
    per-unit-scheduler;
}

[edit class-of-service]
user@PE1# show
interfaces {
    t3-1/0/0:1 {
        unit 0 {
            scheduler-map sched-map-logical-0;
            shaping-rate 10m;
        }
        unit 1 {
            scheduler-map sched-map-logical-1;
            shaping-rate 20m;
        }
    }
}
scheduler-maps {
    sched-map-logical-0 {
        forwarding-class best-effort scheduler sched-best-effort-0;
        forwarding-class assured-forwarding scheduler sched-bronze-0;
        forwarding-class expedited-forwarding scheduler sched-silver-0;
        forwarding-class network-control scheduler sched-gold-0;
    }
    sched-map-logical-1 {
        forwarding-class best-effort scheduler sched-best-effort-1;
        forwarding-class assured-forwarding scheduler sched-bronze-1;
        forwarding-class expedited-forwarding scheduler sched-silver-1;
        forwarding-class network-control scheduler sched-gold-1;
    }
}
schedulers {
    sched-best-effort-0 {
        transmit-rate 4m;
    }
    sched-bronze-0 {
        transmit-rate 3m;
    }
    sched-silver-0 {
        transmit-rate 2m;
    }
    sched-gold-0 {
```

```
    transmit-rate 1m;
  }
  sched-best-effort-1 {
    transmit-rate 8m;
  }
  sched-bronze-1 {
    transmit-rate 6m;
  }
  sched-silver-1 {
    transmit-rate 4m;
  }
  sched-gold-1 {
    transmit-rate 2m;
  }
}
```

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

- Related Documentation**
- [Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs on page 290](#)
 - [Example: Applying Scheduling and Shaping to VLANs on page 303](#)
 - *per-unit-scheduler*

Example: Applying Scheduling and Shaping to VLANs

This example shows how to apply schedulers to individual logical interfaces.

- [Requirements on page 303](#)
- [Overview on page 304](#)
- [Configuration on page 304](#)

Requirements

This example uses the following hardware and software components:

- Junos OS Release 7.4 or later running on router line cards that support Intelligent Queuing (IQ).
- Junos OS Release 13.2 or later running on MX Series routers containing 16x10GE MPC or MPC3E line cards.
- Junos OS Release 13.3 or later running on MX Series routers containing MPC4E line cards.
- Junos OS Release 15.1 or later running on MX Series routers containing MPC6E line cards.

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

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 10.1.1/24
set interfaces xe-9/0/3 unit 2 vlan-id 102
set interfaces xe-9/0/3 unit 2 family inet address 10.2.1/24
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

```

**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 *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 DLCIs and VLANs on page 290](#)
- [Example: Applying Scheduler Maps and Shaping Rate to DLCIs on page 299](#)
- *per-unit-scheduler*

Example: Limiting Egress Traffic on an Interface Using Port Shaping for CoS

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 on page 311](#)
- [Overview on page 311](#)
- [Configuration on page 312](#)
- [Verification on page 316](#)

Requirements

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

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

Overview

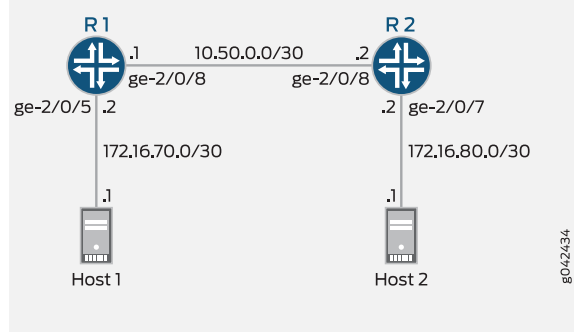
The purpose of this example is to demonstrate how port shaping enables you to shape the traffic passing through an interface to a rate that is less than the line rate for that interface. When you configure port shaping on an interface, you are essentially specifying a value that indicates the maximum 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 33 on page 312](#).

Figure 33: Port Shaping Scenario



Configuration

CLI Quick Configuration

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

Device R1 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 loopback-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 loopback-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 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
```

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*.

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 loopback-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 loopback-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 loopback-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 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 Device R2, enter **commit** from configuration mode.

Verification

Confirm that the configuration is working properly.

- [Clearing the Counters on page 316](#)
- [Sending TCP Traffic into the Network and Monitoring the Port Shaping on page 317](#)

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 Feature Guide*
 - *Example: Configuring a Two-Rate Three-Color Policer*

CHAPTER 8

Setting Transmission Order Using Scheduler Priorities and Hierarchical Scheduling

- [Priority Scheduling Overview on page 319](#)
- [Platform Support for Priority Scheduling on page 321](#)
- [Configuring Schedulers for Priority Scheduling on page 322](#)
- [Associating Schedulers with Fabric Priorities on page 323](#)
- [Hierarchical Class of Service Overview on page 325](#)
- [Hierarchical Class of Service Network Scenarios on page 328](#)
- [Understanding Hierarchical Scheduling on page 329](#)
- [Priority Propagation in Hierarchical Scheduling on page 332](#)
- [Configuring Hierarchical Schedulers for CoS on page 334](#)
- [Hierarchical Schedulers and Traffic Control Profiles on page 335](#)
- [Example: Building a Four-Level Hierarchy of Schedulers on page 337](#)

Priority Scheduling Overview

The Junos OS supports multiple levels of transmission priority, which in order of increasing priority are **low**, **medium-low**, **medium-high**, and **high**, and **strict-high**. This allows the software to service higher-priority queues before lower-priority queues.

Priority scheduling determines the order in which an output interface transmits traffic from its queues, thus ensuring that queues containing important traffic are provided better access to the outgoing interface. Junos OS accomplishes priority scheduling by examining the assigned priority of each individual queue and whether each individual queue is within its defined bandwidth profile. Junos OS 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 interface (or output logical interface if per-unit scheduling is enabled on that interface) are divided into sets based on their priority. Any such set contains queues of the same priority.

Junos OS 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.

The Junos OS 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-priority queues that are currently in profile. These queues are serviced fourth in a weighted round-robin fashion.
5. 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.
6. 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.
7. 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.
8. 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.

Strict-High Priority Configuration Overview

You can configure one queue 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.

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.

**Related
Documentation**

- [How Schedulers Define Output Queue Properties on page 248](#)

Platform Support for Priority Scheduling

Hardware platforms support queue priorities in different ways:

- On all platforms, you can configure one queue per interface to have strict-high priority.
- Strict-high priority works differently on Multiservices and Services PIC link services IQ (**lsq-**) interfaces. For link services IQ interfaces, a queue with strict-high priority might starve all the other queues. For more information, see the *Junos OS Services Interfaces Library for Routing Devices*.
- The priority levels you configure map to hardware priority levels. These priority mappings depend on the FPC type in which the PIC is installed.

[Table 40 on page 322](#) shows the priority mappings by FPC type. Note, for example, that on Juniper Networks M320 Multiservice Edge Routers FPCs, T Series Core Routers FPCs and T Series Enhanced FPCs, the software priorities **medium-low** and **medium-high** behave similarly because they map to the same hardware priority level.

Table 40: Scheduling Priority Mappings by FPC Type

Priority Levels	Mappings for FPCs	Mappings for M320 FPCs and T Series Enhanced FPCs	Mappings for M120 FEBs
low	0	0	0
medium-low	0	1	1
medium-high	1	1	2
high	1	2	3
strict-high (full interface bandwidth)	1	2	3

- Related Documentation**
- [How Schedulers Define Output Queue Properties on page 248](#)
 - [Configuring Schedulers for Priority Scheduling on page 322](#)

Configuring 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**, **medium-low**, **medium-high**, **high**, or **strict-high**. 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. For more information, see “[Platform Support for Priority Scheduling](#)” on page 321.

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 a Gigabit Ethernet interface, **ge-0/0/0**.

```
[edit class-of-service]
user@host# set interfaces ge-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;
  }
}
ge-0/0/0 {
  scheduler-map be-map;
}
```

5. Save your configuration.

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

Related Documentation

- [Priority Scheduling Overview on page 319](#)
- [How Schedulers Define Output Queue Properties on page 248](#)
- [Platform Support for Priority Scheduling on page 321](#)

Associating Schedulers with Fabric Priorities

On Juniper Networks M320 routers, MX Series routers, T Series routers and EX Series switches only, 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 on page 229](#)

Hierarchical Class of Service Overview

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, and physical port.

This allows you to support the requirements of different services, applications, and users on the same physical device and physical infrastructure.

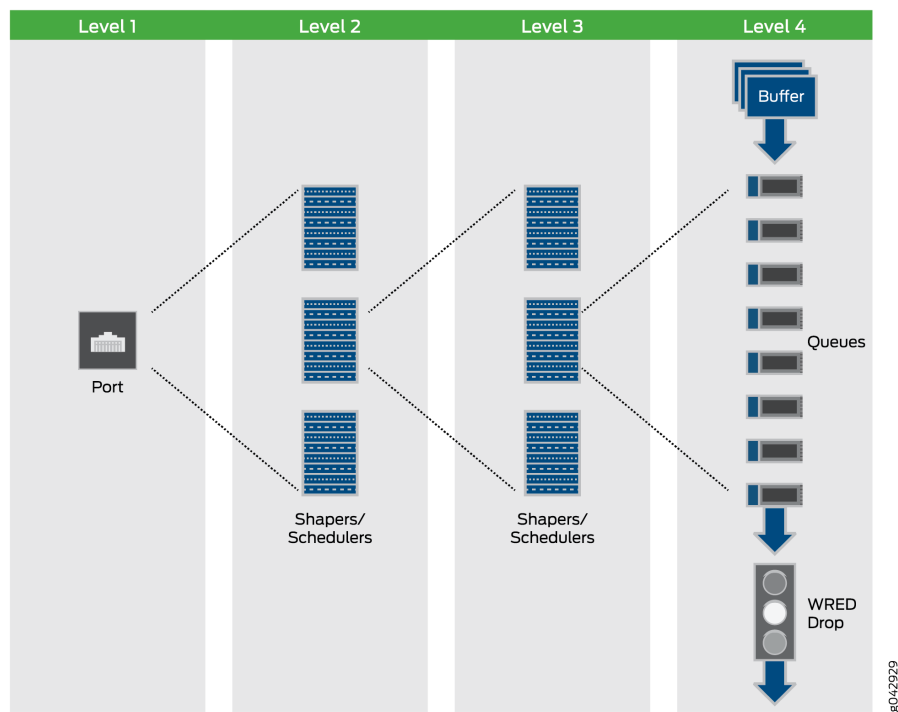
HCoS is implemented primarily using traffic classifiers at the ingress and hierarchical schedulers and shapers at the egress.

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, is the function that controls 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 34 on page 326](#).

Figure 34: 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 3D Universal Edge Routers Overview*
- *CoS Features and Limitations on MX Series Routers*
- *CoS Features of the Router Hardware, PIC, MIC, and MPC Interface Families*
- [How Schedulers Define Output Queue Properties on page 248](#)
- *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 devices 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.

Table 41: Resources for Learning More About HCoS (continued)

Document	Description
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 Feature Guide*.

Related Documentation

- [Hierarchical Class of Service for Subscriber Management Overview](#)
- [Hierarchical Class of Service Network Scenarios on page 328](#)
- [Understanding Hierarchical Scheduling on page 329](#)

Hierarchical Class of Service Network Scenarios

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 on page 325](#)
- [Hierarchical Class of Service for Subscriber Management Overview](#)

Understanding Hierarchical Scheduling

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 on page 330](#)
- [Scheduler Node-Level Designations in Hierarchical Scheduling on page 330](#)
- [Hierarchical Scheduling at Non-Leaf Nodes on page 331](#)

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: Beginning with Junos OS Release 16.1, certain MPCs on MX Series devices support up to five levels of scheduler hierarchies. The concepts presented in this topic apply similarly to five scheduler hierarchy levels.

Table 42 on page 331 describes the possible combinations of scheduler nodes and their corresponding node level designations for a hierarchical queuing MIC or MPC.

Table 42: 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 interface	—	One or more logical interfaces	One or more queues
Interface-sets (collections of logical interfaces) configured, but no traffic-control profiles configured on logical interfaces	Physical interface	—	Interface-set	One or more queues
Fully configured scheduler nodes	Physical interface	Interface-set	One or more logical interfaces	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.

Release History Table

Release	Description
16.1	Beginning with Junos OS Release 16.1, certain MPCs on MX Series devices support up to five levels of scheduler hierarchies.

Priority Propagation in Hierarchical Scheduling

Priority propagation is performed for MX Series router output Interfaces on Enhanced Queuing DPCs, MICs, and MPCs, and for M Series and T Series router output interfaces on IQ2E PICs. 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 43 on page 332](#).

Table 43: 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 44 on page 333](#).

Table 44: 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 45 on page 333](#).

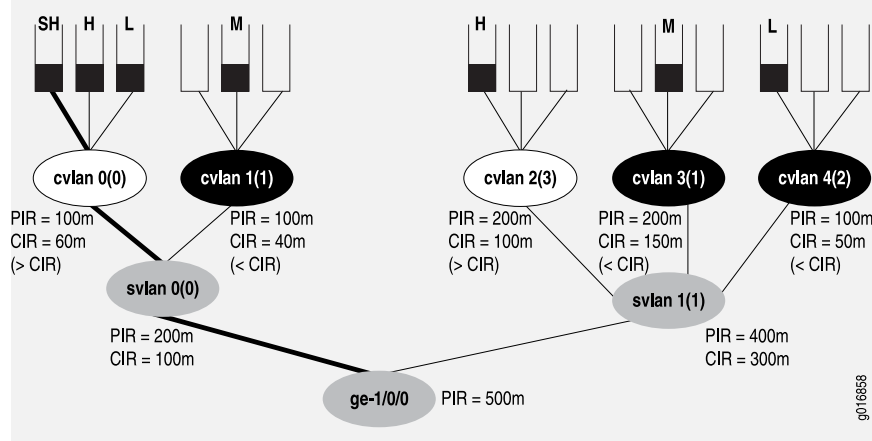
Table 45: Internal Node Queue Priority for PIR-Only Mode

Configured Priority	Hardware Priority
Strict-high	0
High	0
Medium-high	1
Medium-low	1
Low	2

A physical interface with hierarchical schedulers configured is shown in [Figure 35 on page 334](#). 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 35: 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 44 on page 333](#) 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 cvlan 2 (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

- *CoS on Enhanced IQ2 PICs Overview*
- *Enhanced Queuing DPC CoS Properties*
- *CoS Features and Limitations on MIC and MPC Interfaces*
- *Understanding Hierarchical Scheduling for MIC and MPC Interfaces*

Configuring Hierarchical Schedulers for CoS

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.

On Juniper Networks MX Series 5G Universal Routing Platforms and systems with Enhanced IQ2 (IQ2E) PICs, you can apply CoS shaping and scheduling at one of four different levels, including the VLAN set level. You can only use this configuration on MX Series routers or IQ2E PICs. Beginning with Junos OS Release 16.1, certain MPCs 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 is unique to MX Series routers)
- 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 on page 329](#)
- *Understanding Hierarchical Scheduling for MIC and MPC Interfaces*
- *CoS on Enhanced IQ2 PICs Overview*
- *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 Level 4 of the hierarchy.



NOTE: Beginning with Junos OS Release 16.1, certain MPCs on MX Series devices 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 are determined by the scheduler map.

Release History Table

Release	Description
16.1	Beginning with Junos OS Release 16.1, certain MPCs on MX Series devices support up to five levels of scheduler hierarchies.

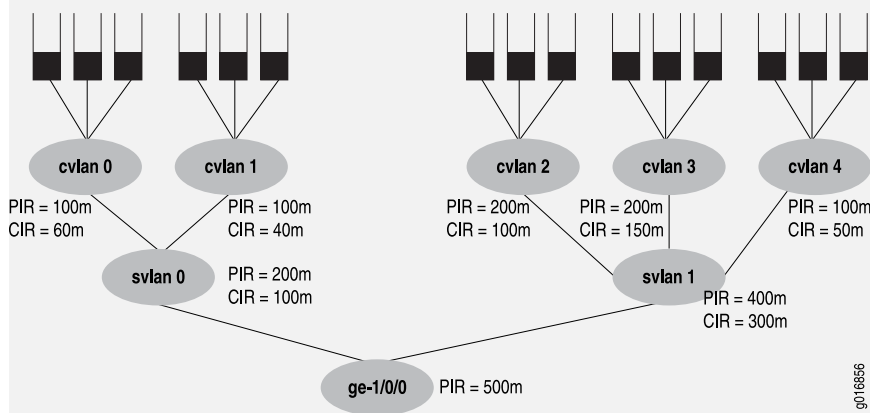
Related Documentation

- [Oversubscribing Interface Bandwidth on page 263](#)
- [Providing a Guaranteed Minimum Rate on page 275](#)
- [Configuring Scheduler Maps on page 252](#)
- [Configuring Traffic Control Profiles for Shared Scheduling and Shaping](#)

Example: Building a Four-Level Hierarchy of Schedulers

This section provides a more complete example of building a 4-level hierarchy of schedulers. The configuration parameters are shown in [Figure 36 on page 337](#). The queues are shown at the top of the figure with the other three levels of the hierarchy below.

Figure 36: 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 on page 338](#)
- [Configuring the Interfaces on page 338](#)
- [Configuring the Traffic Control Profiles on page 339](#)
- [Configuring the Schedulers on page 340](#)
- [Configuring the Drop Profiles on page 340](#)
- [Configuring the Scheduler Maps on page 341](#)
- [Applying the Traffic Control Profiles on page 341](#)

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 [“Managing Congestion Using RED Drop Profiles and Packet Loss Priorities” on page 343](#).

```
[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 340](#).

```
[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.

CHAPTER 9

Controlling Congestion Using Scheduler RED Drop Profiles and Buffers

- [Managing Congestion Using RED Drop Profiles and Packet Loss Priorities on page 343](#)
- [Defining Packet Drop Behavior by Configuring RED Drop Profiles on page 347](#)
- [Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers on page 351](#)
- [Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows on page 353](#)
- [Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size on page 355](#)
- [Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy on page 371](#)
- [Example: Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy on page 373](#)

Managing Congestion Using RED Drop Profiles and Packet Loss Priorities

You can configure two parameters to control congestion at the output stage. The first parameter defines the *delay-buffer bandwidth*, which 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. For more information, see [“Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size” on page 355](#).

The second parameter defines the *drop probabilities* across the range of delay-buffer occupancy, supporting the *random early detection (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, the Junos OS provides the option of enabling 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.

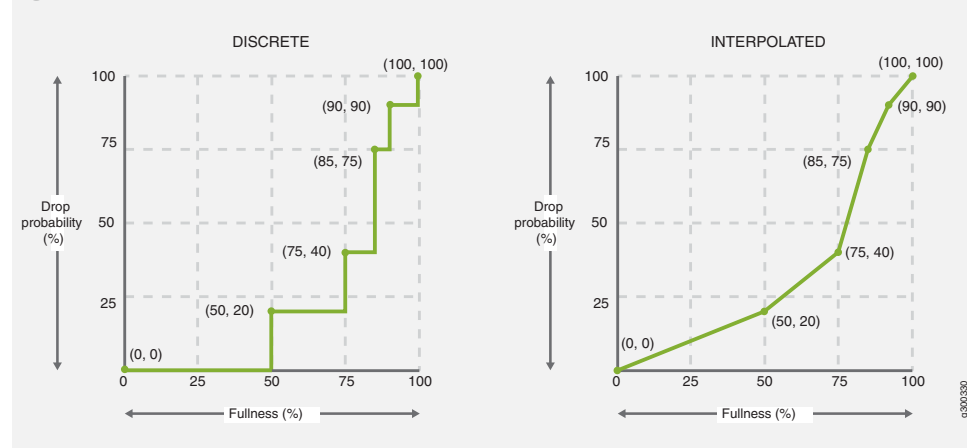
When you configure drop profiles, there are two important values: the queue fullness and the drop probability. The *queue fullness* represents a percentage of the memory used to store packets in relation to the total amount that has been allocated for that specific queue. Similarly, the *drop probability* 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, as shown in [Figure 37 on page 344](#).

The maximum number of queue fullness levels supported per drop profile is based on the line card:

- 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*.
- 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.

[Figure 37 on page 344](#) 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 reaches the head of the queue, a random number between 0 and 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 37: Discrete and Interpolated Drop Profiles



Drop profiles are created by defining multiple fill levels and drop probabilities and can be illustrated by graphs in which the x axis represents the fill level and the y axis represents the drop probability.

To create the discrete profile graph as shown in [Figure 37 on page 344](#) 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 an interpolated drop profile is specified, 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.

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 (low, medium-low, medium-high, or high) and protocol.

You can configure a maximum of 32 different drop profiles.

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 ];
    }
  }
}
```

If you configure no drop profiles on Juniper Networks M320 Multiservice Edge Routers or T Series Core Routers, random early detection (RED) is in effect by default and functions as the primary mechanism for managing congestion. In the default RED drop profile, when the fill-level is 0 percent, the drop probability is 0 percent. When the fill-level is 100 percent, the drop probability is 100 percent.

As a backup method for managing congestion, tail dropping takes effect when congestion of small packets occurs. On M320 and T Series Core Routers, the software supports *tail-RED*, which means that when tail dropping occurs, the software uses RED to execute intelligent tail drops. On other routers, the software executes tail drops unconditionally.

- Related Documentation**
- *drop-probability (Interpolated Value)*
 - *drop-probability (Percentage)*

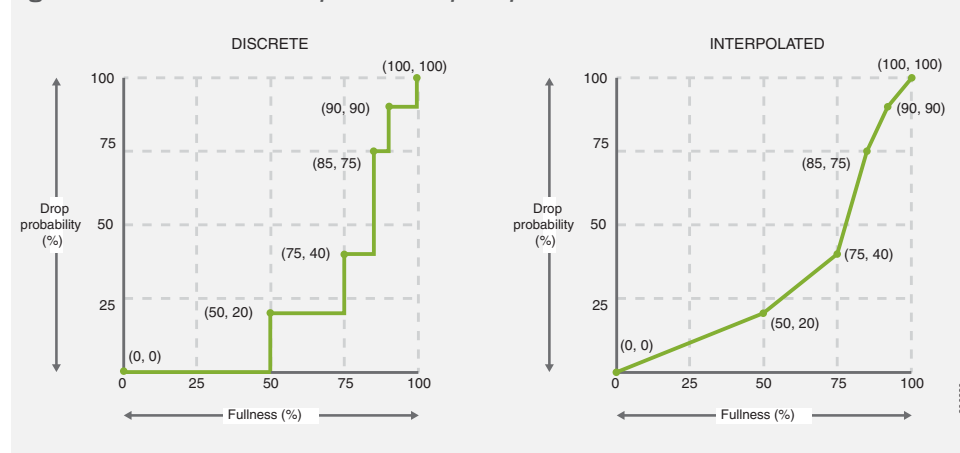
Defining Packet Drop Behavior by Configuring RED Drop Profiles

You enable *random early detection (RED)* by applying a drop profile to a scheduler. When RED is operational on an interface, the queue no longer drops packets from the tail of the queue. Rather, packets are dropped after they reach the head of the queue.

To configure a drop profile, include either the **interpolate** statement and its options, or the fill-level and drop-probability **percentage** values. These two alternatives enable you to configure either each drop probability at up to 64 fill-level/drop-probability paired values, or a profile represented as a series of line segments, as discussed in [“Managing Congestion Using RED Drop Profiles and Packet Loss Priorities” on page 343](#).

For example, the following shows a discrete configuration and an interpolated configuration that correspond to the graphs in [Figure 37 on page 344](#). The values defined in the configurations are matched to represent the data points in the graph lines.

Figure 38: Discrete and Interpolated Drop Profiles



Creating 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;
    }
  }
}
```

To create the discrete profile graph as shown in [Figure 37 on page 344](#) 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) 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.

A smoother graph line can be created 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 defined data points, which you define as follows:

Creating 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-probability percentage
```

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
```



NOTE: 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 351.

Related Documentation

- [Managing Congestion Using RED Drop Profiles and Packet Loss Priorities](#) on page 343

Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers

RED drop profiles take action on outgoing packets. When tricolor marking is enabled, M320, MX Series, and T Series routers 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 [“Defining Packet Drop Behavior by Configuring RED Drop Profiles” on page 347](#).

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 11](#).



NOTE: On Juniper Network MX Series 5G Universal Routing Platforms, T4000 Core Routers, EX Series switches, and PTX Series Packet Transport Routers, 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**

- [Defining Packet Drop Behavior by Configuring RED Drop Profiles on page 347](#)
- [Managing Congestion Using RED Drop Profiles and Packet Loss Priorities on page 343](#)

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 38 and [“Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields”](#) on page 97.

However, on Juniper Networks M320 Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, and T Series Core Routers and EX Series switches 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.

On M320 routers and T Series routers with Enhanced II Flexible PIC Concentrators (FPCs) and tricolor marking enabled, you can set the PLP with a BA or multifield classifier, as described in [“Configuring Behavior Aggregate Classifiers”](#) on page 53 and [“Using Multifield Classifiers to Set Packet Loss Priority”](#) on page 101.

On T Series routers with different Packet Forwarding Engines (non-Enhanced Scaling and Enhanced Scaling FPCs), you can configure PLP bit copying for ingress and egress unicast and multicast traffic. To configure, include the *copy-plp-all* statement at the **[edit class-of-service]** hierarchy level.

The following example shows a two-step procedure to override the default PLP settings on M320 routers.

The first part of this example specifies that while the DSCP code points are 110, the loss priority is set to **high**; however, on M320 routers, 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
```

For M320 routers, 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 expedited-forwarding;
    }
  }
}
```

5. Save your configuration.

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

Related Documentation

- [Mapping PLP to RED Drop Profiles](#)
- [Defining Packet Drop Behavior by Configuring RED Drop Profiles on page 347](#)
- [Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers on page 351](#)
- [Configuring Schedulers on page 252](#)

Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size

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, as shown in [Table 46 on page 356](#).
- The remaining buffer available. The remainder is the buffer percentage that is not assigned to other queues. For example, if you assign 40 percent of the delay buffer to queue 0, allow queue 3 to keep the default allotment of 5 percent, and assign the remainder to queue 7, then queue 7 uses approximately 55 percent of the delay buffer.
- 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. The buffer size temporal value per queue differs by routing device type, as shown in [Table 46 on page 356](#). The maximums apply to the logical interface, not each queue.



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.

Table 46: Buffer Size Temporal Value Ranges by Routing Device Type

Routing Devices	Temporal Value Ranges
M320 and T Series router FPCs, Type 1 and Type 2	1 through 80,000 microseconds
M320 and T Series router FPCs, Type 3. All ES cards (Type 1, 2, 3, and 4).	1 through 50,000 microseconds For PICs with greater than 40 Gbps of total bandwidth, the maximum temporal buffer size that can be configured for a scheduler is 40,000 microseconds instead of 50,000 microseconds.
M120 router FEBs and MX Series router nonenhanced Queuing DPCs, and EX Series switches	1 through 100,000 microseconds
M5, M7i, M10, and M10i router FPCs	1 through 100,000 microseconds
Other M Series router FPCs	1 through 200,000 microseconds
PTX Series Packet Transport Routers	1 through 100,000 microseconds
IQ PICs on all routers	1 through 100,000 microseconds
With Large Buffer Sizes Enabled	
IQ PICs on all routers	1 through 500,000 microseconds
Gigabit Ethernet IQ VLANs	
With shaping rate up to 10 Mbps	1 through 400,000 microseconds
With shaping rate up to 20 Mbps	1 through 300,000 microseconds
With shaping rate up to 30 Mbps	1 through 200,000 microseconds
With shaping rate up to 40 Mbps	1 through 150,000 microseconds
With shaping rate above 40 Mbps	1 through 100,000 microseconds

For more information about configuring delay buffers, see the following subtopics:

- [Configuring Large Delay Buffers for Slower Interfaces on page 357](#)
- [Configuring the Maximum Delay Buffer for NxDS0 Interfaces on page 361](#)
- [Example: Configuring Large Delay Buffers for Slower Interfaces on page 364](#)
- [Example: Configuring the Delay Buffer Value for a Scheduler on page 365](#)
- [Example: Configuring the Physical Interface Shaping Rate on page 367](#)
- [Complete Configuration on page 367](#)
- [Enabling and Disabling the Memory Allocation Dynamic per Queue on page 369](#)

Configuring Large Delay Buffers for Slower Interfaces

By default, T1, E1, and NxDS0 interfaces and DLCIs configured on channelized IQ PICs are limited to 100,000 microseconds of delay buffer. (The default average packet size on the IQ PIC is 40 bytes.) For these interfaces, it might be necessary to configure a larger buffer size to prevent congestion and packet dropping. You can do so on the following PICs:

- Channelized IQ
- 4-port E3 IQ
- Gigabit Ethernet IQ and IQ2

Congestion and packet dropping occur when large bursts of traffic are received by slower interfaces. This happens when faster interfaces pass traffic to slower interfaces, which is often the case when edge devices receive traffic from the core of the network. For example, a 100,000-microsecond T1 delay buffer can absorb only 20 percent of a 5000-microsecond burst of traffic from an upstream OC3 interface. In this case, 80 percent of the burst traffic is dropped.

[Table 47 on page 357](#) shows some recommended buffer sizes needed to absorb typical burst sizes from various upstream interface types.

Table 47: Recommended Delay Buffer Sizes

Length of Burst	Upstream Interface	Downstream Interface	Recommended Buffer on Downstream Interface
5000 microseconds	OC3	E1 or T1	500,000 microseconds
5000 microseconds	E1 or T1	E1 or T1	100,000 microseconds
1000 microseconds	T3	E1 or T1	100,000 microseconds

To ensure that traffic is queued and transmitted properly on E1, T1, and NxDS0 interfaces and DLCIs, you can configure a buffer size larger than the default maximum. To enable larger buffer sizes to be configured:

1. Include the **q-pic-large-buffer (large-scale | small-scale)** statement at the **[edit chassis fpc slot-number pic pic-number]** hierarchy level.

```
[edit]
user@host# edit chassis fpc slot-number pic pic-number
user@host# set q-pic-large-buffer large-scale
```

If you specify the **large-scale** option, the feature supports a larger number of interfaces. If you specify **small-scale**, the default, then the feature supports a smaller number of interfaces.

When you include the **q-pic-large-buffer** statement in the configuration, the larger buffer is transparently available for allocation to scheduler queues. The larger buffer maximum varies by interface type, as shown in [Table 48 on page 358](#).

Table 48: Maximum Delay Buffer with q-pic-large-buffer Enabled by Interface

Platform, PIC, or Interface Type	Maximum Buffer Size
With Large Buffer Sizes Not Enabled	
M320 and T Series router FPCs, Type 1 and Type 2	80,000 microseconds
M320 and T Series router FPCs, Type 3	50,000 microseconds
Other M Series router FPCs	200,000 microseconds
IQ PICs on all routers	100,000 microseconds
With Large Buffer Sizes Enabled	
Channelized T3 and channelized OC3 DLCIs—Maximum sizes vary by shaping rate:	
With shaping rate from 64,000 through 255,999 bps	4,000,000 microseconds
With shaping rate from 256,000 through 511,999 bps	2,000,000 microseconds
With shaping rate from 512,000 through 1,023,999 bps	1,000,000 microseconds
With shaping rate from 1,024,000 through 2,048,000 bps	500,000 microseconds
With shaping rate from 2,048,001 bps through 10 Mbps	400,000 microseconds
With shaping rate from 10,000,001 bps through 20 Mbps	300,000 microseconds
With shaping rate from 20,000,001 bps through 30 Mbps	200,000 microseconds
With shaping rate from 30,000,001 bps through 40 Mbps	150,000 microseconds
With shaping rate from 40,000,001 bps and above	100,000 microseconds
NxDSO IQ Interfaces—Maximum sizes vary by channel size:	
1xDSO through 3xDSO	4,000,000 microseconds
4xDSO through 7xDSO	2,000,000 microseconds
8xDSO through 15xDSO	1,000,000 microseconds
16xDSO through 32xDSO	500,000 microseconds
Other IQ interfaces	500,000 microseconds

If you configure a delay buffer larger than the new maximum, the candidate configuration can be committed successfully. However, the setting is rejected by the packet forwarding component and a system log warning message is generated.

For interfaces that support DLCI queuing, the large buffer is supported for DLCIs on which the configured shaping rate is less than or equal to the physical interface bandwidth. For instance, when you configure a Frame Relay DLCI on a Channelized T3 IQ PIC, and you configure the shaping rate to be 1.5 Mbps, the amount of delay buffer that can be allocated to the DLCI is 500,000 microseconds, which is equivalent to a T1 delay buffer. For more information about DLCI queuing, see [“Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs” on page 290](#).

For NxDSO interfaces, the larger buffer sizes can be up to 4,000,000 microseconds, depending on the number of DSO channels in the NxDSO interface. For slower NxDSO interfaces with fewer channels, the delay buffer can be relatively larger than for faster NxDSO interfaces with more channels. This is shown in [Table 50 on page 362](#).

You can allocate the delay buffer as either a percentage or a temporal value. The resulting delay buffer is calculated differently depending how you configure the delay buffer, as shown in [Table 49 on page 359](#).

Table 49: Delay-Buffer Calculations

Delay Buffer Configuration	Formula	Example
Percentage	$\text{available interface bandwidth} * \text{configured percentage buffer-size} * \text{maximum buffer} = \text{queue buffer}$	<p>If you configure a queue on a T1 interface to use 30 percent of the available delay buffer, the queue receives 28,125 bytes of delay buffer:</p> <pre> sched-expedited { transmit-rate percent 30; buffer-size percent 30; } </pre> <p> $1.5 \text{ Mbps} * 0.3 * 500,000 \text{ microseconds} = 225,000 \text{ bits}$ $= 28,125 \text{ bytes}$ </p>
Temporal	$\text{available interface bandwidth} * \text{configured percentage transmit-rate} * \text{configured temporal buffer-size} = \text{queue buffer}$	<p>If you configure a queue on a T1 interface to use 500,000 microseconds of delay buffer and you configure the transmission rate to be 20 percent, the queue receives 18,750 bytes of delay buffer:</p> <pre> sched-best { transmit-rate percent 20; buffer-size temporal 500000; } </pre> <p> $1.5 \text{ Mbps} * 0.2 * 500,000 \text{ microseconds} = 150,000 \text{ bits}$ $= 18,750 \text{ bytes}$ </p>

Table 49: Delay-Buffer Calculations (continued)

Delay Buffer Configuration	Formula	Example
Percentage, with buffer size larger than transmit rate		<p>In this example, the delay buffer is allocated twice the transmit rate. Maximum delay buffer latency can be up to twice the 500,000-microsecond delay buffer if the queue's transmit rate cannot exceed the allocated transmit rate.</p> <pre> sched-extra-buffer { transmit-rate percent 10; buffer-size percent 20; } </pre>
FRF.16 LSQ bundles	<p>For total bundle bandwidth < T1 bandwidth, the delay-buffer rate is 1 second.</p> <p>For total bundle bandwidth >= T1 bandwidth, the delay-buffer rate is 200 milliseconds (ms).</p>	

Configuring the Maximum Delay Buffer for NxDSO Interfaces

Because $N \times \text{DSO}$ interfaces carry less bandwidth than a T1 or E1 interface, the buffer size on an $N \times \text{DSO}$ interface can be relatively larger, depending on the number of DSO channels combined. The maximum delay buffer size is calculated with the following formula:

$$\text{Interface Speed} * \text{Maximum Delay Buffer Time} = \text{Delay Buffer Size}$$

For example, a 1xDSO interface has a speed of 64 kilobits per second (Kbps). At this rate, the maximum delay buffer time is 4,000,000 microseconds. Therefore, the delay buffer size is 32 kilobytes (KB):

$$64 \text{ Kbps} * 4,000,000 \text{ microseconds} = 32 \text{ KB}$$

Table 50 on page 362 shows the delay-buffer calculations for 1xDSO through 32xDSO interfaces.

Table 50: $N \times \text{DSO}$ Transmission Rates and Delay Buffers

Interface Speed	Delay Buffer Size
1xDSO Through 4xDSO: Maximum Delay Buffer Time Is 4,000,000 Microseconds	
1xDSO: 64 Kbps	32 KB
2xDSO: 128 Kbps	64 KB
3xDSO: 192 Kbps	96 KB
4xDSO Through 7xDSO: Maximum Delay Buffer Time Is 2,000,000 Microseconds	
4xDSO: 256 Kbps	64 KB
5xDSO: 320 Kbps	80 KB
6xDSO: 384 Kbps	96 KB
7xDSO: 448 Kbps	112 KB
8xDSO Through 15xDSO: Maximum Delay Buffer Time Is 1,000,000 Microseconds	
8xDSO: 512 Kbps	64 KB
9xDSO: 576 Kbps	72 KB
10xDSO: 640 Kbps	80 KB
11xDSO: 704 Kbps	88 KB
12xDSO: 768 Kbps	96 KB
13xDSO: 832 Kbps	104 KB

Table 50: NxDSO Transmission Rates and Delay Buffers (continued)

Interface Speed	Delay Buffer Size
14xDSO: 896Kbps	112 KB
15xDSO: 960 Kbps	120 KB
16xDSO Through 32xDSO: Maximum Delay Buffer Time Is 500,000 Microseconds	
16xDSO: 1024 Kbps	64 KB
17xDSO: 1088 Kbps	68 KB
18xDSO: 1152 Kbps	72 KB
19xDSO: 1216 Kbps	76 KB
20xDSO: 1280 Kbps	80 KB
21xDSO: 1344 Kbps	84 KB
22xDSO: 1408 Kbps	88 KB
23xDSO: 1472 Kbps	92 KB
24xDSO: 1536 Kbps	96 KB
25xDSO: 1600 Kbps	100 KB
26xDSO: 1664 Kbps	104 KB
27xDSO: 1728 Kbps	108 KB
28xDSO: 1792 Kbps	112 KB
29xDSO: 1856 Kbps	116 KB
30xDSO: 1920 Kbps	120 KB
31xDSO: 1984 Kbps	124 KB

Table 50: NxDSO Transmission Rates and Delay Buffers (continued)

Interface Speed	Delay Buffer Size
32xDSO: 2048 Kbps	128 KB

Example: Configuring Large Delay Buffers for Slower Interfaces

Set large delay buffers on interfaces configured on a Channelized OC12 IQ PIC. The CoS configuration binds a scheduler map to the interface specified in the chassis configuration. For information about the delay-buffer calculations in this example, see [Table 49 on page 359](#).

To configure a large delay buffer:

1. Specify the FPC and PIC for which you want to configure large delay buffers.

```
[edit]
user@host# edit chassis fpc 0 pic 0
```

2. Enable large delay buffering.

```
[edit chassis fpc 0 pic 0]
user@host# set q-pic-large-buffer
```

3. Specify the maximum number of queues per interface.

```
[edit chassis fpc 0 pic 0]
user@host# set max-queues-per-interface 8
```

4. Verify the configuration.

```
[edit chassis fpc 0 pic 0]
user@host# show
```

```
q-pic-large-buffer {
    large-scale;
}
max-queues-per-interface 8;
```

5. Save the configuration.

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

Example: Configuring the Delay Buffer Value for a Scheduler

You can assign to a physical or logical interface, a scheduler map that is composed 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 that of the specified transmit rate of 10 percent. Assuming that the **sched-best** scheduler is assigned to a T1 interface, this scheduler receives 20 percent of the total 500,000 microseconds of the T1 interface's delay buffer. Therefore, the scheduler receives 18,750 bytes of delay buffer:

$$\text{available interface bandwidth} * \text{configured percentage buffer-size} * \text{maximum buffer} = \text{queue buffer}$$

$$1.5 \text{ Mbps} * 0.2 * 500,000 \text{ microseconds} = 150,000 \text{ bits} = 18,750 \text{ bytes}$$

Assuming that the **sched-exped** scheduler is assigned to a T1 interface, this scheduler receives 300,000 microseconds of the T1 interface's 500,000-microsecond delay buffer with the traffic rate at 20 percent. Therefore, the scheduler receives 11,250 bytes of delay buffer:

$$\text{available interface bandwidth} * \text{configured percentage transmit-rate} * \text{configured temporal buffer-size} = \text{queue buffer}$$

$$1.5 \text{ Mbps} * 0.2 * 300,000 \text{ microseconds} = 90,000 \text{ bits} = 11,250 \text{ bytes}$$

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: Configuring 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. For more information, see [Table 50 on page 362](#).

This example configures the shaping rate on a T1 interface to 200 Kbps, which means that the T1 interface bandwidth is set to 200 Kbps instead of 1.5 Mbps. Because 200 Kbps is less than 4xDS0, 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 t1-0/0/0:1:1
```

2. Specify the shaping rate.

```
[edit class-of-service interfaces t1-0/0/0:1:1]
user@host# set shaping-rate 200k
```

3. Verify the configuration.

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

```
interfaces {
  t1-0/0/0:1:1 {
    shaping-rate 200k;
  }
}
```

4. Save the configuration.

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

Complete Configuration

This example shows a Channelized OC12 IQ PIC in FPC slot 0, PIC slot 0 and a channelized T1 interface with Frame Relay encapsulation. It also shows a scheduler map configuration on the physical interface.

```
chassis {
```

```
fpc 0 {
  pic 0 {
    q-pic-large-buffer;
    max-queues-per-interface 8;
  }
}
interfaces {
  coc12-0/0/0 {
    partition 1 oc-slice 1 interface-type coc1;
  }
  coc1-0/0/0:1 {
    partition 1 interface-type t1;
  }
  t1-0/0/0:1:1 {
    encapsulation frame-relay;
    unit 0 {
      family inet {
        address 10.1.1.1/24;
      }
      dlci 100;
    }
  }
}
class-of-service {
  interfaces {
    t1-0/0/0:1:1 {
      scheduler-map smap-1;
    }
  }
  scheduler-maps {
    smap-1 {
      forwarding-class best-effort scheduler sched-best;
      forwarding-class expedited-forwarding scheduler sched-exped;
      forwarding-class assured-forwarding scheduler sched-assure;
      forwarding-class network-control scheduler sched-network;
    }
  }
  schedulers {
    sched-best {
      transmit-rate percent 40;
      buffer-size percent 40;
    }
    sched-exped {
      transmit-rate percent 30;
      buffer-size percent 30;
    }
    sched-assure {
      transmit-rate percent 20;
      buffer-size percent 20;
    }
    sched-network {
      transmit-rate percent 10;
      buffer-size percent 10;
    }
  }
}
```

```

    }
  }
}

```

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.

For Juniper Networks M320 Multiservice Edge Routers, MX Series 5G Universal Routing Platforms, T Series Core Routers, and EX Series Ethernet Switches only, the MAD mechanism is enabled unless the delay buffer is configured with a temporal setting for a given queue. 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. All M320, MX Series, and T Series router and EX Series switches' FPCs and Packet Forwarding Engines support MAD. No Modular Port Concentrators (MPCs) and IQ, IQ2, IQ2E or IQE PICs support MAD.

To enable the MAD mechanism on supported hardware:

1. 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 18K, 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;
    }
  }
}
```

- 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, IQ-PICs can be configured to use *weighted average* buffer occupancy information. This option is configured on a per-PIC basis and applies to the following IQ-PICs:

- Channelized T1/T3
- Channelized E1/E3
- Channelized OC3/STM1
- Channelized OC12

If you configure this feature on an unsupported PIC, you see an error message.

If you configure this feature on a channelized OC12 intelligent queuing (IQ) interface, the PIC reboots.

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 IQ-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 Q-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 on page 373](#)
- [red-buffer-occupancy](#)

Example: Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy

This topic provides two examples for configuring the weighted RED buffer occupancy feature to manage traffic bursts.

- [Requirements on page 373](#)
- [Overview on page 373](#)
- [Configuration on page 373](#)

Requirements

Weighted RED buffer occupancy is configured on a per-PIC basis and applies to only the following IQ-PICs:

- Channelized T1/T3
- Channelized E1/E3
- Channelized OC3/STM1
- Channelized OC12

If you configure this feature on an unsupported PIC, you see an error message.



NOTE: If you configure this feature on a channelized OC12 intelligent queuing (IQ) interface, the PIC reboots.

Overview

To manage traffic bursts on IQ 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

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 Q-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 Q-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 Q-PIC to Use a Weight Value of 1/2 in Average Buffer Occupancy Calculations

Step-by-Step Procedure

To configure a Q-PIC to use a weight value of 1/2 in average buffer occupancy calculations:

1. Specify the Q-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 Q-PIC to Use a Weight Value of 1/4 in Average Buffer Occupancy Calculations

Step-by-Step Procedure

To configure a Q-PIC to use a weight value of 1/4 in average buffer occupancy calculations:

1. Specify the Q-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 on page 371](#)
- *red-buffer-occupancy*

Altering Outgoing Packet Headers Using Rewrite Rules to Ensure Forwarding Behavior

- [Rewriting Packet Headers to Ensure Forwarding Behavior on page 378](#)
- [Applying Default Rewrite Rules on page 379](#)
- [Configuring Rewrite Rules on page 381](#)
- [Configuring Rewrite Rules Based on PLP on page 383](#)
- [Applying IEEE 802.1p Rewrite Rules to Dual VLAN Tags on page 383](#)
- [Applying IEEE 802.1ad Rewrite Rules to Dual VLAN Tags on page 385](#)
- [Rewriting IEEE 802.1p Packet Headers with an MPLS EXP Value on page 386](#)
- [Setting IPv6 DSCP and MPLS EXP Values Independently on page 388](#)
- [Configuring DSCP Values for IPv6 Packets Entering the MPLS Tunnel on page 388](#)
- [Setting Ingress DSCP Bits for Multicast Traffic over Layer 3 VPNs on page 390](#)
- [Applying Rewrite Rules to Output Logical Interfaces on page 391](#)
- [Rewriting MPLS and IPv4 Packet Headers on page 393](#)
- [Defining a Custom Frame Relay Loss Priority Map on page 398](#)
- [Example: Per-Node Rewriting of EXP Bits on page 399](#)
- [Example: Rewriting CoS Information at the Network Border to Enforce CoS Strategies on page 400](#)
- [Example: Remarking Diffserv Code Points to MPLS EXPs to Carry CoS Profiles Across a Service Provider's L3VPN MPLS Network on page 410](#)
- [Example: Remarking Diffserv Code Points to 802.1P PCPs to Carry CoS Profiles Across a Service Provider's VPLS Network on page 436](#)

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.

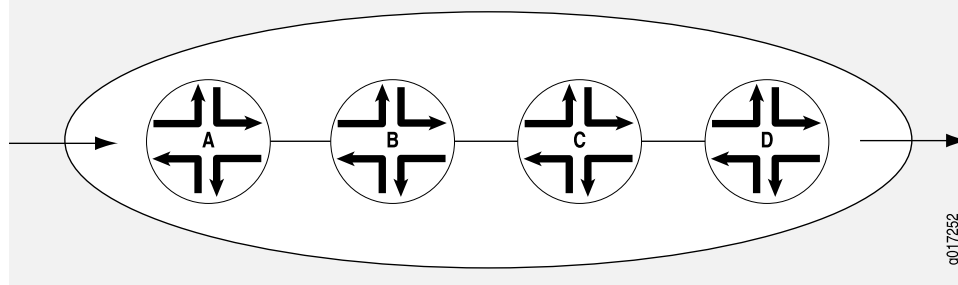
In effect, the rewrite rule performs the opposite function of the behavior aggregate (BA) classifier used when the packet enters the routing device. As the packet leaves the routing platform, the final CoS action is generally the application of a rewrite rule.

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 39 on page 378 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 39: 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);
    }
  }
}
```

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 380 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 42, and the default forwarding classes shown in “Default Forwarding Classes” on page 202.

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 380, the software maps the header bits of those packets to the code-point aliases in the last column in Table 51 on page 380. 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 42.

Table 51: Default Packet Header Rewrite Mappings

Map from Forwarding Class	PLP Value	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

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 42 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 42 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 42 and Table 51 on page 380.

```
class-of-service {
  interfaces {
    so-1/2/3 {
      unit 0 {
```

```

rewrite-rules {
    dscp default;
}
}
}
}
}

```

Related Documentation • [Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic on page 38](#)

Configuring Rewrite Rules

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);
        }
    }
}

```



NOTE: 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**.

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 11](#).

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.

Starting with Junos OS Release 18.1R1, 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.

On the M320, T1600, and MX960 routers and 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.

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.



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.

Release History Table

Release	Description
18.1	Starting with Junos OS Release 18.1R1, 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.

Related Documentation

- [Applying Rewrite Rules to Output Logical Interfaces on page 391](#)
- [Applying Egress Interface Rewrite Rules to the IEEE 802.1p Field for All Host Outbound Traffic on the Interface](#)

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 199.

Applying IEEE 802.1p Rewrite Rules to Dual VLAN Tags

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

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 on Juniper Networks M320 Multiservice Edge Routers, MX Series Ethernet Service Routers, T Series Core Routers, and EX Series switches that have a peer connection to an M Series Multiservice Edge Router, MX Series router, T Series router, or EX Series switch, 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. 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 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.

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.

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 {
  so-3/1/0 {
    unit 0 {
      rewrite-rules {
        exp exp-ieee-table;
        ieee-802.1 default;
      }
    }
  }
}

```

Setting IPv6 DSCP and MPLS EXP Values Independently

On the M120, M320 with Enhanced III FPCs, MX Series 5G Universal Routing Platforms, and EX Series switches, 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:

- This feature is supported only on M120, M320 with Enhanced III FPCs, MX Series Ethernet Services routers, and EX Series switches.
- 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 on page 388](#)

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 on page 388](#)

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 378](#).

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 *Multicast Protocols Feature 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

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;
}

```

On M120, M320 with an Enhanced III FPC, MX Series routers and T 4000 routers with Type 5 FPCs and EX Series switches, 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.

For IQ PICs, you can configure only 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. If you configure more than one IEEE 802.1 rewrite rule for the IQ PIC, the configuration check fails.

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 ssf-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.



NOTE: On M Series routers only, if you include the `control-word` statement at the `[edit protocols l2circuit neighbor address interface interface-name]` hierarchy level, the software cannot rewrite MPLS EXP bits.

DSCP and DSCP IPv6 rewrite rules are supported on M Series and T Series routers when non-queuing PICs are installed, but are disabled when queuing PICs are installed with the following exceptions:

- On M320 routers, DSCP rewrite is supported on IQ, IQ2, IQE, and IQ2E PICs when used with the Enhanced III FPC.
- On M120 routers, DSCP rewrite is supported on IQ, IQ2, IQE, and IQ2E PICs.

DSCP and DCSP IPv6 rewrite rules are supported on MX Series routers with MPC/MIC interfaces and on EX Series switches.

Inet6 precedence rewrite rules are supported on MX Series routers with MPC/MIC interfaces.

DSCP rewrite rules are not supported on T Series routers when IQ, IQ2, IQE, IQ2E, SONET/SDH OC48/STM16 IQE, or PD-5-10XGE-SFPP PICs are installed.

For IQ PICs, you can configure only 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.

On M320 and T Series routers (except for T4000 routers with Type 5 FPCs), for a single interface, you cannot enable a rewrite rule on a subset of forwarding classes. You must assign a rewrite rule to either none of the forwarding classes or all of the forwarding classes. When you assign a rewrite rule to a subset of forwarding classes, the commit does not fail, and the subset of forwarding classes works as expected. However, the forwarding classes to which the rewrite rule is not assigned are rewritten to all zeros.

For example, if you configure a Differentiated Services code point (DSCP) rewrite rule, the bits in the forwarding classes to which you do not assign the rewrite rule are rewritten to 000000; if you configure an IP precedence rewrite rule, the bits in the forwarding classes to which you do not assign the rewrite rule are rewritten to 000.

Related Documentation

- [Setting IPv6 DSCP and MPLS EXP Values Independently on page 388](#)
- [Configuring DSCP Values for IPv6 Packets Entering the MPLS Tunnel on page 388](#)

Rewriting MPLS and IPv4 Packet Headers

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.

The default MPLS EXP rewrite rules are shown in [Table 52 on page 394](#).

Table 52: 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.

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**—Applies the rewrite rule to MPLS packets and writes the CoS value to MPLS headers.
- **mpls-inet-both**—Applies the rewrite rule to VPN MPLS packets with IPv4 payloads. On Juniper Networks M120 Multiservice Edge Routers, M320 Multiservice Edge Routers, and T Series Core Routers (except T4000 routers), writes the CoS value to the MPLS and IPv4 headers. On other M Series Multiservice Edge Router routers, causes all ingress MPLS LSP packets with IPv4 payloads to be initialized with **000** code points for the MPLS EXP value, and the configured rewrite code point for IP precedence.

- **mpls-inet-both-non-vpn**—Applies the rewrite rule to non-VPN MPLS packets with IPv4 payloads. On Juniper Networks M120 Multiservice Edge Routers, M320 Multiservice Edge Routers, and T Series Core Routers, writes the CoS value to the MPLS and IPv4 headers. On other M Series Multiservice Edge Routers, causes all ingress MPLS LSP packets with IPv4 payloads to be initialized with **000** code points for the MPLS EXP value, and the configured rewrite code point for IP precedence.

On M120 routers, M320 routers with Enhanced-III FPCs, and 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 379](#).

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

On M320 and T Series routers, configure rewrite tables and apply them in various ways to achieve the following results:

- For interface **so-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 **so-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.
 - IPv4 packets (non-VPN) that enter the LSPs on interface **so-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.
 - Non-IPv4 packets that enter the LSPs on interface **so-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 **so-3/1/0**, IPv4 packets transmitted over a non-LSP layer are initialized with values from IP precedence rewrite table **rule2**.
- For interface **so-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.

- For interface **so-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.

```
[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 {
  so-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.
      }
    }
  }
  so-3/1/1 {
    unit 0 {
      rewrite-rules {
        exp exp-inet-table protocol [mpls mpls-inet-both];
      }
    }
  }
}
```

```
}
}
```

Example: Simultaneous DSCP and EXP Rewrite

On M120 routers, M320 routers with Enhanced-III FPCs, and MX Series routers, 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.

Defining a Custom Frame Relay Loss Priority Map

You can apply a classifier to the same interface on which you configure a Frame Relay loss priority value. The Frame Relay loss priority map is applied first, followed by the classifier. The classifier can change the loss priority to a higher value only (for example, from low to high). If the classifier specifies a loss priority with a lower value than the current loss priority of a particular packet, the classifier does not change the loss priority of that packet.

To define a custom Frame Relay loss priority map:

1. At the **[edit class-of-service loss-priority-maps]** hierarchy level in configuration mode, specify the loss priority map for the Frame Relay DE bit.

```
[edit class-of-service loss-priority-maps]
user@host# set frame-relay-de name loss-priority level code-points [ alias | bits ];
```

For example:

```
[edit class-of-service loss-priority-maps]
user@host# set frame-relay-de fr_rw loss-priority low code-points 0;
user@host# set frame-relay-de fr_rw loss-priority high code-points 0;
user@host# set frame-relay-de fr_rw loss-priority medium-low code-points 1;
user@host# set frame-relay-de fr_rw loss-priority medium-high code-points 1;
```



NOTE: The loss priority map does not take effect until you apply it to a logical interface.

2. Apply a rule to a logical interface.

```
[edit class-of-service interfaces interface-name unit logical-unit-number
  loss-priority-maps]
user@host# set frame-relay-de name;
```

For example:

```
[edit class-of-service interfaces so-1/0/0 unit 0 loss-priority-maps]
user@host# set frame-relay-de fr_rw;
```

3. Verify the configuration in operational mode.

```
user@host> show class-of-service loss-priority-map
Loss-priority-map: frame-relay-de-fr_rw, Code point type: frame-relay-de,
Index: 38
```

Code point	Loss priority
0	low
0	high
1	medium-low
1	medium-high

Related Documentation • [frame-relay-de](#)

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

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 on page 400](#)
- [Overview on page 400](#)
- [Configuration on page 402](#)
- [Verification on page 409](#)

Requirements

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

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

Overview

The purpose of this example is to demonstrate CoS rewriting at a network border to convey the traffic'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. A partial table of the default rewrite rule mappings is shown in the following table.

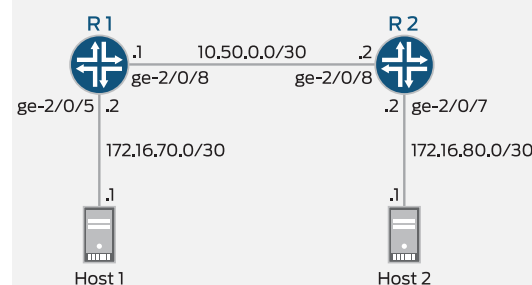
Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
expedited-forwarding	low	ef
expedited-forwarding	high	ef
assured-forwarding	low	af11
assured-forwarding	high	af12(DSCP/DSCP IPv6/EXP)
best-effort	low	be
best-effort	high	be
network-control	low	nc1/cs6
network-control	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 40 on page 401](#).

Figure 40: 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

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 loopback-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 *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 loopback-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 loopback-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

Confirm that the configuration is working properly.

- [Clearing the Firewall Counters on page 409](#)
- [Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Monitoring the Results on page 409](#)

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 on page 260](#)

Example: Remarking Diffserv Code Points to MPLS EXPs to Carry CoS Profiles Across a Service Provider's L3VPN MPLS Network

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 on page 411](#)
- [Overview on page 411](#)
- [Configuration on page 413](#)
- [Verification on page 434](#)

Requirements

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

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

Overview

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 42 on page 413](#).

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.

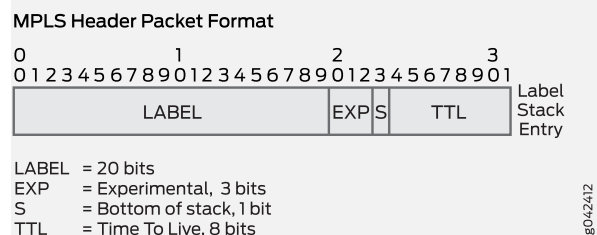
Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
expedited-forwarding	low	ef
expedited-forwarding	high	ef
assured-forwarding	low	af11
assured-forwarding	high	af12 (DSCP/DSCP IPv6/EXP)
best-effort	low	be
best-effort	high	be
network-control	low	nc1/cs6
network-control	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
assured-forwarding	high	101
network-control	low	110
network-control	high	111

Figure 41 on page 412 shows the MPLS packet structure.

Figure 41: MPLS Packet Structure



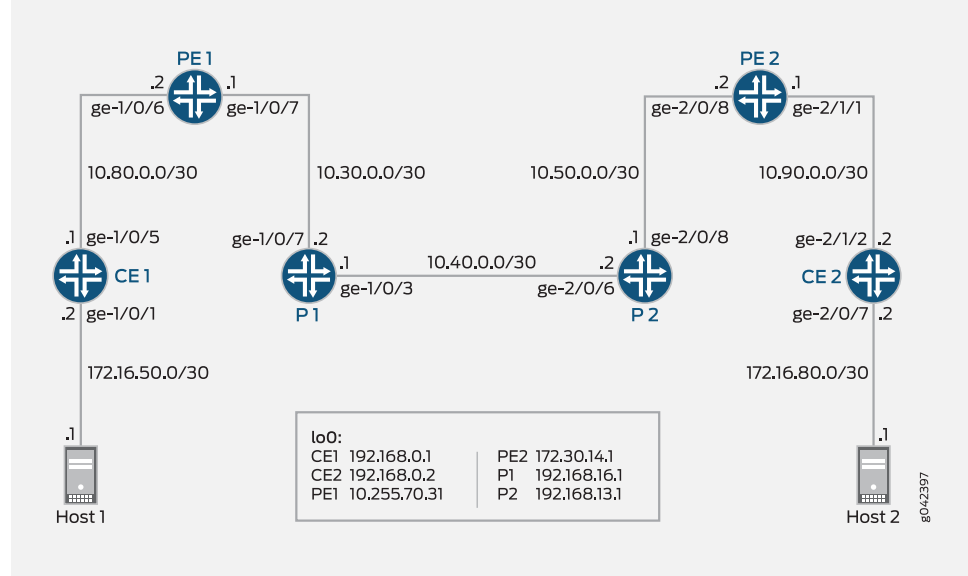
NOTE: In addition to providing the necessary information to complete the purpose of this example, this example also includes all of the commands required to re-create the Layer 3 VPN (L3VPN) network as shown in [Figure 42 on page 413](#). A full explanation of the tasks required to configure an L3VPN network is not included in this example. If you require more information regarding configuring an L3VPN network, refer to the *Layer 3 VPNs Feature 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 42 on page 413](#).

Figure 42: Rewriting CoS Information at the Network Border to Transit an MPLS Network Scenario



Configuration

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

```
Device 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 dscp4 forwarding-class expedited-forwarding
  loss-priority low code-points ef
set class-of-service classifiers dscp dscp4 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 dscp4
set class-of-service interfaces ge-2/1/1 unit 0 rewrite-rules dscp dscp4-rw
set class-of-service rewrite-rules dscp dscp4-rw forwarding-class expedited-forwarding
  loss-priority low code-point ef
set class-of-service rewrite-rules dscp dscp4-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 *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 dscp4 forwarding-class
expedited-forwarding loss-priority low code-points ef
user@PE1# set class-of-service classifiers dscp dscp4 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 dscp4
user@PE1# set class-of-service interfaces ge-1/0/6 unit 0 rewrite-rules dscp
dscp4-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 dscp4-rw forwarding-class
expedited-forwarding loss-priority low code-point ef
user@PE1# set class-of-service rewrite-rules dscp dscp4-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 dscp4 forwarding-class
expedited-forwarding loss-priority low code-points ef
user@PE2# set class-of-service classifiers dscp dscp4 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 dscp4
user@PE2# set class-of-service interfaces ge-2/1/1 unit 0 rewrite-rules dscp
dscp4-rw
user@PE2# set class-of-service rewrite-rules dscp dscp4-rw forwarding-class
expedited-forwarding loss-priority low code-point ef
user@PE2# set class-of-service rewrite-rules dscp dscp4-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 dscp4 {
    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 dscp4;
            }
            rewrite-rules {
                dscp dscp4-rw;
            }
        }
    }
    ge-1/0/7 {
        unit 0 {
            classifiers {
                exp exp-in;
            }
            rewrite-rules {
                exp exp-out;
            }
        }
    }
}
rewrite-rules {
    dscp dscp4-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 dscp4 {  
    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 dscp4;
            }
            rewrite-rules {
                dscp dscp4-rw;
            }
        }
    }
}
rewrite-rules {
    dscp dscp4-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

Confirm that the configuration is working properly by verifying that the DSCP code points are maintained from CE1 to CE2.

- [Clearing the Firewall Counters on page 434](#)
- [Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Monitoring the Results on page 434](#)

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 on page 260](#)

## Example: Remarking Diffserv Code Points to 802.1P PCPs to Carry CoS Profiles Across a Service Provider's VPLS Network

---

This configuration example explains how to implement class-of-service (CoS) capabilities over a Virtual Private LAN Service (VPLS) network.

- [Requirements on page 436](#)
- [Overview on page 436](#)
- [Configuration on page 439](#)
- [Verification on page 458](#)

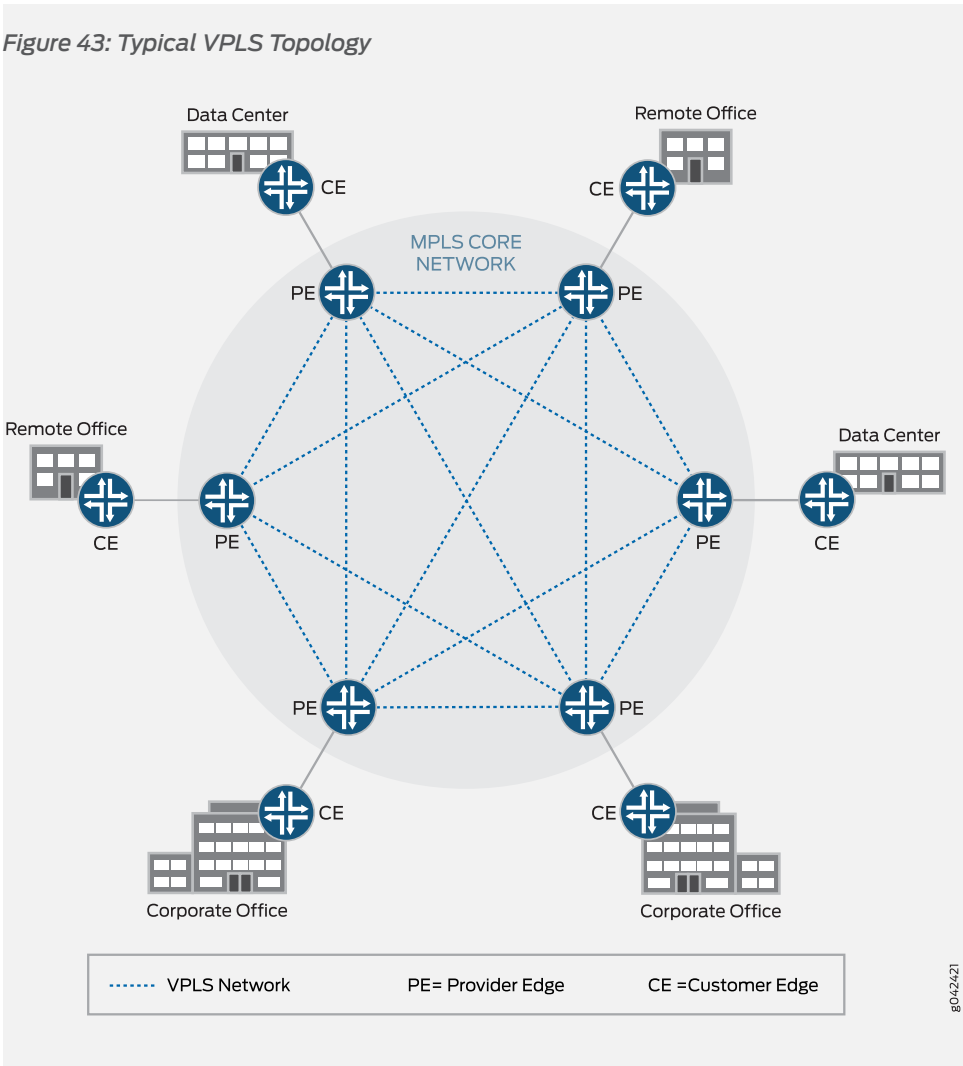
### Requirements

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

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

### Overview

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 43 on page 437](#) for an example of a 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.

| 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                                            |

| Map from Forwarding Class | PLP Value | MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases |
|---------------------------|-----------|-------------------------------------------------|
| 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            |
| assured-forwarding   | high          | 101            |
| network-control      | low           | 110            |
| network-control      | high          | 111            |



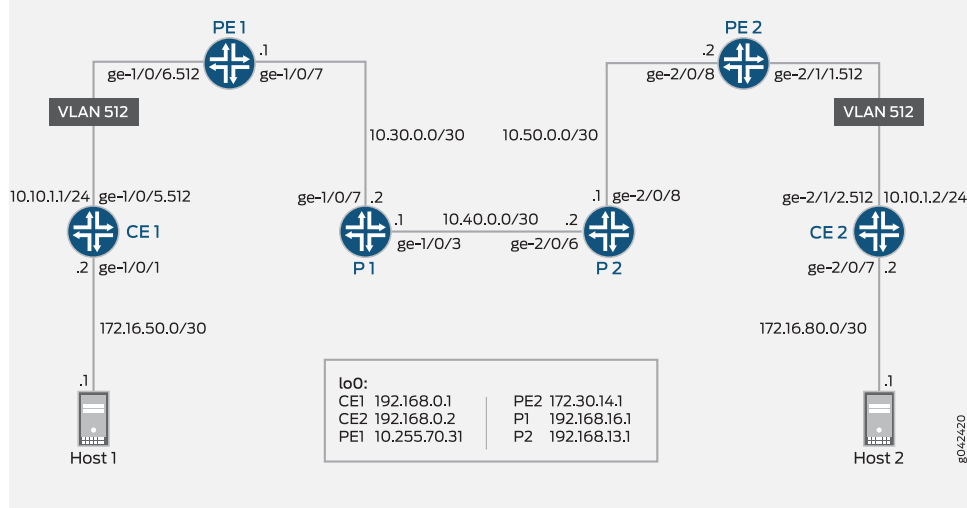
**NOTE:** In addition to providing the necessary information to complete the purpose of this example, this example also includes all of the commands required to recreate the VPLS network as shown in [Figure 44 on page 439](#). A full explanation of the tasks required to configure a VPLS network is not included in this example. If you need more information regarding configuring a VPLS network, see the *VPLS Feature 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](http://www.juniper.net/books).

## Topology

This example uses the topology in [Figure 44 on page 439](#).

Figure 44: VPLS with CoS Scenario



## Configuration

### CLI Quick Configuration

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

#### Device 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 *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

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 on page 458](#)
- [Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Verifying the Results on page 458](#)

---

### 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 on page 260](#)



## CHAPTER 11

# Altering Class of Service Values in Packets Exiting the Network Using IPv6 DiffServ

- [Resources for CoS with DiffServ for IPv6 on page 461](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Terms and Acronyms for CoS with DiffServ for IPv6 on page 462](#)
- [Default DSCP Mappings on page 463](#)
- [Default Forwarding Classes on page 464](#)
- [Juniper Networks Default Forwarding Classes on page 467](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [Configuring a Firewall Filter for an MF Classifier on Customer Interfaces on page 470](#)
- [Applying the Firewall Filter to Customer Interfaces on page 471](#)
- [Assigning Forwarding Classes to Output Queues on page 471](#)
- [Configuring Rewrite Rules on page 472](#)
- [DSCP IPv6 Rewrites and Forwarding Class Maps on page 473](#)
- [Applying Rewrite Rules to an Interface on page 473](#)
- [Configuring RED Drop Profiles on page 474](#)
- [Configuring BA Classifiers on page 474](#)
- [Applying a BA Classifier to an Interface on page 475](#)
- [Configuring a Scheduler on page 476](#)
- [Configuring Scheduler Maps on page 476](#)
- [Applying a Scheduler Map to an Interface on page 477](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

## Resources for CoS with DiffServ for IPv6

---

For additional information about CoS using DiffServ for IPv6, see the following:

- [RFC 1924, \*A Compact Representation of IPv6 Addresses\*](#)
- [RFC 2474, \*Definition of the Differentiated Services Field \(DS Field\) in the IPv4 and IPv6 Headers\*](#)

- RFC 2475, *An Architecture for Differentiated Services*
- RFC 2640, *Internet Protocol, Version 6 (IPv6) Specification*
- RFC 2983, *Differentiated Service and Tunnels*
- RFC 3260, *New Terminology and Clarifications for DiffServ*
- RFC 3317, *Differentiated Services Quality of Service Policy Information Base*
- RFC 3513, *IP Version 6 Addressing Architecture*

**Related  
Documentation**

- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)

---

## System Requirements for CoS with DiffServ for IPv6

To implement CoS with DiffServ for IPv6, your system must meet these minimum requirements:

- Junos OS Release 8.2 or later for MX Series routers
- Junos OS Release 6.3 or later for M Series and T Series routers
- Three Juniper Networks M Series, MX Series, or T Series routers
- For M Series routers, Enhanced FPCs capable of supporting DSCPs and, for MF classifiers, Internet Processor II ASICs

**Related  
Documentation**

- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

---

## Terms and Acronyms for CoS with DiffServ for IPv6

### C

|                         |                                                                                                                   |
|-------------------------|-------------------------------------------------------------------------------------------------------------------|
| <b>class of service</b> | A set of forwarding class parameters that define different treatment for different traffic flows.                 |
| <b>classifier</b>       | A method of reading a sequence of bits in a packet header or label and determining the packet's forwarding class. |

### D

|                                                  |                                                                                                                            |
|--------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| <b>Differentiated Services (DiffServ)</b>        | A standards-based method of associating CoS parameters with traffic flows and their forwarding classes.                    |
| <b>Differentiated Services code point (DSCP)</b> | Values for a 6-bit field defined for IPv4 and IPv6 packet headers that can be used to enforce CoS distinctions in routers. |

- Related Documentation**
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
  - [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
  - [Example: Configuring DiffServ for IPv6 on page 477](#)

## Default DSCP Mappings

Table 53 on page 463 shows the mapping of DiffServ service class meanings (aliases) to DSCPs.

*Table 53: Default DSCP Mappings*

| DiffServ Service Class Alias | IPv4 and IPv6 DSCP Mapping |
|------------------------------|----------------------------|
| ef                           | 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                           | 000000                     |
| cs1                          | 001000                     |
| cs2                          | 010000                     |
| cs3                          | 011000                     |
| cs4                          | 100000                     |

Table 53: Default DSCP Mappings (continued)

| DiffServ Service Class Alias | IPv4 and IPv6 DSCP Mapping |
|------------------------------|----------------------------|
| cs5                          | 101000                     |
| nc1/cs6                      | 110000                     |
| nc2/cs7                      | 111000                     |

None of the aliases are established by DiffServ specifications. The aliases are well-known only through usage. For example, it is widely accepted that the alias for DSCP 101110 is **ef** (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 is usually 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 (these packets are not delay-sensitive). Typically, these packets represent routing protocol hello or keepalive messages and loss of these packets jeopardizes proper network operation, so delay is preferable to discard.

#### Related Documentation

- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

## Default Forwarding Classes

Table 54 on page 465 shows the default forwarding class and packet loss priority (PLP) for the well-known DSCPs. It is important to note that although several DSCPs map to

the **expedited-forwarding** and **assured-forwarding** classes, by default no resources are assigned to these forwarding classes. All of these settings can be changed through configuration.

*Table 54: Default Behavior Aggregate Classification*

| DSCP and DSCP IPv6 | Forwarding Class     | PLP  |
|--------------------|----------------------|------|
| ef                 | expedited-forwarding | low  |
| af11               | assured-forwarding   | low  |
| af12               | assured forwarding   | high |
| af13               | assured forwarding   | high |
| af21               | best-effort          | low  |
| af22               | best-effort          | low  |
| af23               | best-effort          | low  |
| af31               | best-effort          | low  |
| af32               | best-effort          | low  |
| af33               | best-effort          | low  |
| af41               | best-effort          | low  |
| af42               | best-effort          | low  |
| af43               | best-effort          | low  |
| be                 | best-effort          | low  |
| cs1                | best-effort          | low  |
| cs2                | best-effort          | low  |
| cs3                | best-effort          | low  |
| cs4                | best-effort          | low  |
| cs5                | best-effort          | low  |
| nc1/cs6            | network-control      | low  |
| nc2/cs7            | network control      | low  |
| other              | best-effort          | low  |

Table 55 on page 466 shows the router forwarding classes associated with each well-known DSCP code point and the resources assigned to their output queues.

**Table 55: Forwarding Classes and Queues Classification**

| DSCP Alias | DSCP Bits | Forwarding Class     | PLP  | Queue |
|------------|-----------|----------------------|------|-------|
| ef         | 101110    | expedited-forwarding | low  | 1     |
| af11       | 001010    | assured-forwarding   | low  | 2     |
| af12       | 001100    | assured-forwarding   | high | 2     |
| af13       | 001110    | assured-forwarding   | high | 2     |
| af21       | 010010    | best-effort          | low  | 0     |
| af22       | 010100    | best-effort          | low  | 0     |
| af23       | 010110    | best-effort          | low  | 0     |
| af31       | 011010    | best-effort          | low  | 0     |
| af32       | 011100    | best-effort          | low  | 0     |
| af33       | 011110    | best-effort          | low  | 0     |
| af41       | 100010    | best-effort          | low  | 0     |
| af42       | 100100    | best-effort          | low  | 0     |
| af43       | 100110    | best-effort          | low  | 0     |
| be         | 000000    | best-effort          | low  | 0     |
| cs1        | 001000    | best-effort          | low  | 0     |
| cs2        | 010000    | best-effort          | low  | 0     |
| cs3        | 011000    | best-effort          | low  | 0     |
| cs4        | 100000    | best-effort          | low  | 0     |
| cs5        | 101000    | best-effort          | low  | 0     |
| nc1/cs6    | 110000    | network-control      | low  | 3     |
| nc2/cs7    | 111000    | network-control      | low  | 3     |
| other      | —         | best-effort          | low  | 0     |



[Table 56 on page 467](#) shows the resources assigned to the four forwarding classes in this example.

*Table 56: Resources Assigned to Queues*

| Queue | Forwarding Class            | Transmit Rate | Buffer Size | Priority        |
|-------|-----------------------------|---------------|-------------|-----------------|
| 0     | <b>be</b> (data)            | 40%           | 40%         | Low             |
| 1     | <b>ef</b> (financial)       | 10%           | 10%         | High            |
| 2     | <b>af</b> (audiovisual)     | 45%           | 45%         | High (with RED) |
| 3     | <b>nc</b> (network control) | 5%            | 5%          | Low             |

The table shows how the 95 percent of output link transmission rate and buffer size (queue) resources assigned by default to Q0 (best-effort) are distributed to Q1 (expedited forwarding) and Q2 (assured forwarding). The audiovisual traffic consumes more bandwidth than other applications, but the financial information, although critical, is carried in fewer packets. In keeping with DiffServ specifications, a RED drop profile is applied to the assured forwarding class. The financial data has a strict set of traffic parameters that must be respected.

The three DiffServ assured forwarding classes supported (**af11**, **af12**, and **af13**, with low, medium, and high packet drop probability, respectively) are distinguished by using a low PLP and RED drop profile for **af11** and a high PLP and RED for **af12** and **af13**. All of these parameters should be closely monitored initially for performance and adjusted as necessary.

#### Related Documentation

- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

## Juniper Networks Default Forwarding Classes

Most M Series routers have only four queues built into the hardware. M120, M320, MX Series, and T Series routers can be configured for up to eight queues. If a classifier does not assign a packet to any other queue (for example, for other than well-known DSCPs that have not been added to the classifier), the packet is assigned by default to the class associated with queue 0 (Q0).

[Table 57 on page 468](#) shows the four forwarding classes and queues to which Juniper Networks classifiers assign a packet based on the DSCP values in arriving packet headers.

Table 57: Default Forwarding Classes

| Forwarding Class Name | Queue   |
|-----------------------|---------|
| best-effort           | queue 0 |
| expedited-forwarding  | queue 1 |
| assured-forwarding    | queue 2 |
| network-control       | queue 3 |

Each forwarding class has an associated scheduler priority. Only two forwarding classes, **best-effort** and **network-control** (Q0 and Q3), are actually referenced in the default scheduler configuration. However, you can manually configure resources for the **expedited-forwarding** and **assured-forwarding** classes (Q1 and Q2).

The default scheduler settings are not visible in the output of the **show class-of-service** command; rather, they are implicit.

#### Default Scheduler

```
[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;
 }
}
```

By default, the **best-effort** forwarding class (Q0) receives 95 percent of the output link bandwidth and buffer space, and the **network-control** forwarding class (Q3) receives 5 percent of the output link bandwidth and buffer space. The default drop profile provides *tail drop*, where the buffer fills and then discards all packets until there is space in the buffer again. There are no schedulers for the **expedited-forwarding** or **assured-forwarding** classes because by default no resources are assigned to Q1 and Q2.

All **af** classes other than **af1x** are mapped to **best-effort**, since RFC 2597 prohibits a node from aggregating classes. In effect, mapping to **best-effort** implies that the node does not support that class.

- Related Documentation**
- [Understanding How Class of Service Manages Congestion and Controls Service Levels in the Network on page 4](#)
  - [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
  - [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
  - [Example: Configuring DiffServ for IPv6 on page 477](#)

---

## Roadmap for Configuring CoS with IPv6 DiffServ

---

To configure class of service (CoS) over IPv6, you must:

- Configure a multifield (MF) classifier for IPv6 to detect packets of interest to CoS and assign the packet to the proper forwarding class independently of Differentiated Services Code Point (DSCP). See [“Configuring a Firewall Filter for an MF Classifier on Customer Interfaces” on page 470](#).

Next, apply the MF classifier to the appropriate interface. See [“Applying the Firewall Filter to Customer Interfaces” on page 471](#).

- Assign the forwarding classes established by the MF classifier to output queues. See [“Assigning Forwarding Classes to Output Queues” on page 471](#).
- Configure rewrite rules to replace DSCPs on packets received from the customer with the values expected by other routers. See [“Configuring Rewrite Rules” on page 472](#).

Next, apply the rewrite rules to the appropriate interface. See [“Applying Rewrite Rules to an Interface” on page 473](#).

- Configure behavior aggregate (BA) classifiers for IPv6 on network interfaces because the DSCPs have been explicitly rewritten on the edge routers. See [“Configuring BA Classifiers” on page 474](#).

Next, apply the BA classifier to the appropriate interface. See [“Applying a BA Classifier to an Interface” on page 475](#).

- Configure random early detection (RED) drop profiles to determine the probability of DiffServ assured forwarding packets being discarded under congested conditions. See [“Configuring RED Drop Profiles” on page 474](#).
- Configure schedulers to assign resources, priorities, and drop profiles to output queues. See [“Configuring a Scheduler” on page 476](#).
- Configure a scheduler map to assign a forwarding class to a scheduler. See [“Configuring Scheduler Maps” on page 476](#).

Next, apply the scheduler map to the appropriate interface. See [“Applying a Scheduler Map to an Interface” on page 477](#).

- Related Documentation**
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
  - [Example: Configuring DiffServ for IPv6 on page 477](#)

## Configuring a Firewall Filter for an MF Classifier on Customer Interfaces

You configure an MF classifier for IPv6 to detect packets of interest to CoS and assign the packet to the proper forwarding class independently of DSCP. To configure an MF classifier on a customer-facing link, configure a policer for the expedited forwarding traffic and a firewall filter to classify traffic.

```
[edit firewall]
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 {
 0:0: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.
}
}
}
```

**Related Documentation**

- [Applying the Firewall Filter to Customer Interfaces on page 471](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

---

## Applying the Firewall Filter to Customer Interfaces

You apply an MF classifier firewall filter for IPv6 to customer interfaces. To apply an MF classifier firewall filter on customer-facing links, apply the classifier as an input filter at the **[edit interfaces]** hierarchy level.

```
[edit interfaces]
so-0/0/1 {
 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;
 }
 }
}
```

**Related Documentation**

- [Configuring a Firewall Filter for an MF Classifier on Customer Interfaces on page 470](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

---

## Assigning Forwarding Classes to Output Queues

You must assign the forwarding classes established by the MF classifier to output queues. To assign a forwarding class to an output queue, include the **forwarding-classes** statement at the **[edit class-of-service]** hierarchy level.

```
[edit class-of-service]
forwarding-classes {
 queue 0 be-DATA-class;
 queue 1 ef-FIN-class;
 queue 2 af-AV-class;
 queue 3 nc-CONTROL-class;
}
```

**Related  
Documentation**

- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

## Configuring Rewrite Rules

---

You configure rewrite rules to replace DSCPs on packets received from the customer with the values expected by other routers. Rewrite rules use the forwarding class information and packet loss priority (PLP) used internally by the router to establish the DSCP on outbound packets. To configure rewrite rules, include the **rewrite-rules** statement at the **[edit class-of-service]** hierarchy level.

```
[edit class-of-service]
rewrite-rules rewrite-IPv6-dscps {
 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;
 }
}
```

**Related  
Documentation**

- [Applying Rewrite Rules to an Interface on page 473](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

## DSCP IPv6 Rewrites and Forwarding Class Maps

You cannot configure a DSCP IPv6 rewrite rule and output forwarding class map on the same logical interface (unit). These must be used on different logical interfaces. 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.

This example shows the warning and error message that results when the default DSCP IPv6 rewrite rule is configured on logical interface **ge-1/0/4.0** with output forwarding class map **vg1**.

```
[edit class-of-service]
interfaces {
 ge-1/0/4 {
 unit 0 {
 ##
 ## Warning: DSCP-IPv6 rewrite and forwarding class map not allowed on same unit
 ##
 output-forwarding-class-map vg1;
 rewrite-rules {
 dscp-ipv6 default;
 }
 }
 }
}
```

```
user@router# commit
```

```
[edit class-of-service interfaces ge-1/0/4 unit 0 output-forwarding-class-map]
'output-forwarding-class-map vg1'
DSCP-IPv6 rewrite and forwarding class map not allowed on same unit
error: commit failed: (statements constraint check failed)
```

Related Documentation • [Applying Forwarding Classes to Interfaces on page 226](#)

## Applying Rewrite Rules to an Interface

To apply the configured rewrite rules, include the **rewrite-rules** statement at the **[edit class-of-service interfaces]** hierarchy level.

```
[edit class-of-service interfaces]
so-0/1/1 {
 unit 0 {
 rewrite-rules {
 dscp-ipv6 rewrite-IPv6-dscps;
 }
 }
}
```

- Related Documentation**
- [Configuring Rewrite Rules on page 472](#)
  - [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
  - [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
  - [Example: Configuring DiffServ for IPv6 on page 477](#)

---

## Configuring RED Drop Profiles

You configure RED drop profiles to determine the probability of DiffServ assured forwarding packets being discarded under congested conditions. To configure RED drop profiles for assured forwarding without the PLP bit set and with the PLP bit set, include the **drop-profiles** statement at the **[edit class-of-service]** hierarchy level.

```
[edit class-of-service]
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];
 }
 }
}
```

Assured forwarding traffic with the PLP bit set has a more aggressive drop probability than traffic without the PLP bit set.

- Related Documentation**
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
  - [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
  - [Example: Configuring DiffServ for IPv6 on page 477](#)

---

## Configuring BA Classifiers

You configure BA classifiers for IPv6 on network interfaces because the DSCPs have been explicitly rewritten on the edge routers. To configure a BA classifier for IPv6 DSCPs, include the **dscp-ipv6** statement and give the classifier a name. Then import the default classifier and specify the forwarding class, loss priority, and code points for each established traffic class at the **[edit class-of-service]** hierarchy level.

```
[edit class-of-service]
```



```
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;
 }
 }
}
```

**Related Documentation**

- [Applying a BA Classifier to an Interface on page 475](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

---

## Applying a BA Classifier to an Interface

To apply the configured classifier, include the **classifiers** statement at the **[edit class-of-service interfaces]** hierarchy level.

```
[edit class-of-service interfaces]
so-0/1/1 {
 unit 0 {
 classifiers {
 dscp-ipv6 IPv6-classifier;
 }
 }
}
```

**Related Documentation**

- [Configuring BA Classifiers on page 474](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

## Configuring a Scheduler

You configure schedulers to assign resources, priorities, and drop profiles to output queues. To configure a scheduler, include the **schedulers** statement at the **[edit class-of-service]** hierarchy level.

```
[edit class-of-service]
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;
 }
}
```

### Related Documentation

- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

## Configuring Scheduler Maps

You configure a scheduler map to assign a forwarding class to a scheduler. To configure a scheduler map, include the **scheduler-maps** statement and scheduler name at the **[edit class-of-service]** hierarchy level.

```
[edit class-of-service]
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;
}
}
```

**Related  
Documentation**

- [Applying a Scheduler Map to an Interface on page 477](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

---

## Applying a Scheduler Map to an Interface

To apply the configured scheduler map, include the **scheduler-map** statement at the **[edit class-of-service]** hierarchy level.

```
[edit class-of-service]
interfaces {
 so-1/0/1 {
 scheduler-map diffserv-cos-map;
 }
}
```

**Related  
Documentation**

- [Configuring Scheduler Maps on page 476](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)
- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Example: Configuring DiffServ for IPv6 on page 477](#)

---

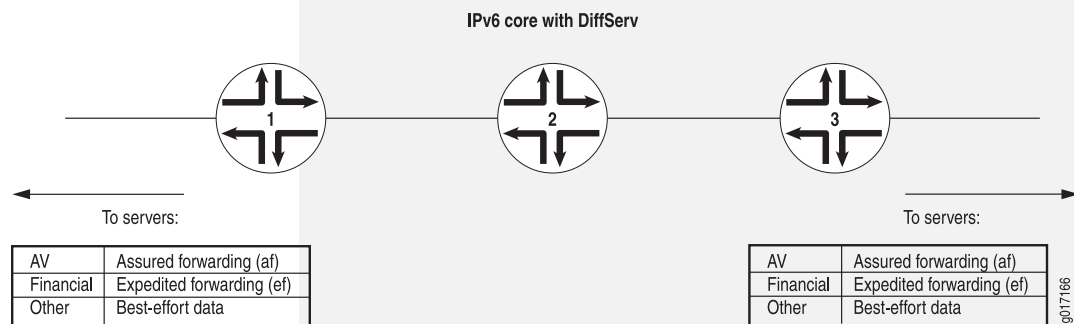
## Example: Configuring DiffServ for IPv6

### Configuration

The example assigns expedited forwarding to Q1 and a subset of the assured forwarding classes (**af1x**) to Q2, and distributes resources among all four forwarding classes.

[Figure 45 on page 478](#) shows the topology of the three routers and links that are used as a case study in this chapter.

Figure 45: Basic IPv6 DiffServ Topology



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 46: IPv6 DiffServ Configuration

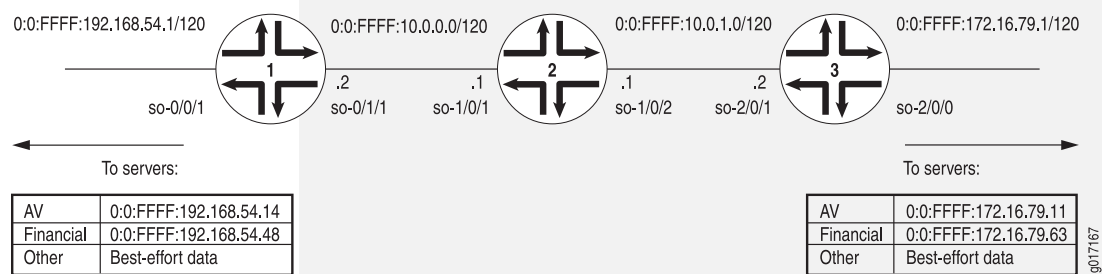


Figure 46 on page 478 shows the complete topology for IPv6 DiffServ, complete with interfaces and IPv6 addresses. The IPv4-mapped IPv6 address format described in RFC 5952 is used.

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 {
 O: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
```

| Interface  | Index | Table Index/<br>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
...

```

#### Related Documentation

- [System Requirements for CoS with DiffServ for IPv6 on page 462](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ on page 469](#)