

# Junos® OS

---

## Class of Service User Guide (Routers and EX9200 Switches)

Published  
2024-12-17



Juniper Networks, Inc.  
1133 Innovation Way  
Sunnyvale, California 94089  
USA  
408-745-2000  
[www.juniper.net](http://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 Class of Service User Guide (Routers and EX9200 Switches)*  
Copyright © 2024 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 This Guide | xxiv

## 1

### Overview

Understanding How Class of Service Manages Congestion and Defines Traffic Forwarding Behavior | 2

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

How CoS Applies to Packet Flow Across a Network | 5

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

Mapping CoS Component Inputs to Outputs | 10

Default Junos OS CoS Settings | 14

Packet Flow Through the Junos OS CoS Process Overview | 18

Configuring Basic Packet Flow Through the Junos OS CoS Process | 23

Define Classifiers | 26

Apply Classifiers to Incoming Packets on Interfaces | 27

Define Policers to Limit Traffic and Control Congestion | 28

Define Drop Profiles | 29

Assign Each Forwarding Class to a Queue | 29

Define Schedulers | 29

Define Scheduler Maps | 30

Define CoS Header Rewrite Rules | 31

Apply Scheduler Maps and Rewrite Rules to Egress Interfaces | 31

Example: Classifying All Traffic from a Remote Device by Configuring Fixed Interface-Based Classification | 32

Requirements | 33

Overview | 33

Configuration | 35

Verification | 38

Interface Types That Do Not Support Junos OS CoS | 40



## Configuring Class of Service

### Assigning Service Levels with Behavior Aggregate Classifiers | 43

Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 44

Default IP Precedence Classifier | 48

Default DSCP and DSCP IPv6 Classifiers | 49

Default MPLS EXP Classifier | 51

Default IEEE 802.1p Classifier | 53

Default IEEE 802.1ad Classifier | 54

Default Aliases for CoS Value Bit Patterns Overview | 56

Defining Aliases for CoS Value Bit Patterns | 61

Configuring Behavior Aggregate Classifiers | 63

Applying Behavior Aggregate Classifiers to Logical Interfaces | 66

Example: Configuring and Applying a Default DSCP Behavior Aggregate Classifier | 71

Requirements | 72

Overview | 72

Configuration | 76

Verification | 79

Example: Configuring Behavior Aggregate Classifiers | 82

Requirements | 82

Overview | 82

Configuration | 83

Verification | 86

Understanding DSCP Classification for VPLS | 89

Example: Configuring DSCP Classification for VPLS | 91

Requirements | 92

Overview | 92

Configuration | 92

Configuring Class of Service for MPLS LSPs | 95

Applying DSCP Classifiers to MPLS Traffic | 99



Applying a DSCP Classifier to MPLS Packets on a Core-facing Interface | 100

Applying a DSCP Classifier to MPLS Traffic for L3VPN/VPLS | 102

Applying MPLS EXP Classifiers to Routing Instances | 105

Configuring and Applying Custom MPLS EXP Classifiers to Routing Instances | 106

Applying Global Classifiers and Wildcard Routing Instances | 107

Applying Global MPLS EXP Classifiers to Routing Instances | 108

Applying Classifiers by Using Wildcard Routing Instances | 110

Verifying the Classifiers Associated with Routing Instances | 111

Applying MPLS EXP Classifiers for Explicit-Null Labels | 112

Manage Ingress Oversubscription with Traffic Class Maps | 115

Ingress Oversubscription at the Packet Forwarding Engine | 115

Configure Traffic Class Maps | 115

Example: Configuring Traffic Class Maps | 119

Requirements | 119

Overview and Topology | 120

Configuration | 121

Verification | 127

**Assigning Service Levels with Multifield Classifiers | 132**

Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields | 132

Configuring Multifield Classifiers | 133

Using Multifield Classifiers to Set Packet Loss Priority | 137

Example: Configuring and Applying a Firewall Filter for a Multifield Classifier | 139

Requirements | 139

Overview | 139

Configuration | 141

Verification | 145

Example: Classifying Packets Based on Their Destination Address | 148

Requirements | 148

Overview | 148

Configuration | 148

Example: Configuring and Verifying a Complex Multifield Filter | 151



## **Controlling Network Access with Traffic Policing | 156**

Controlling Network Access Using Traffic Policing Overview | **156**

Effect of Two-Color Policers on Shaping Rate Changes | **162**

Configuring Policers Based on Logical Interface Bandwidth | **163**

Example: Limiting Inbound Traffic at Your Network Border by Configuring an Ingress Single-Rate Two-Color Policer | **167**

Requirements | **168**

Overview | **168**

Configuration | **171**

Verification | **177**

Example: Performing CoS at an Egress Network Boundary by Configuring an Egress Single-Rate Two-Color Policer | **179**

Requirements | **180**

Overview | **180**

Configuration | **183**

Verification | **189**

Example: Limiting Inbound Traffic Within Your Network by Configuring an Ingress Single-Rate Two-Color Policer and Configuring Multifield Classifiers | **192**

Requirements | **192**

Overview | **192**

Configuration | **196**

Verification | **204**

Example: Limiting Outbound Traffic Within Your Network by Configuring an Egress Single-Rate Two-Color Policer and Configuring Multifield Classifiers | **209**

Requirements | **210**

Overview | **210**

Configuration | **215**

Verification | **224**

Overview of Tricolor Marking Architecture | **229**

Enabling Tricolor Marking and Limitations of Three-Color Policers | **232**

Configuring and Applying Tricolor Marking Policers | **234**

Defining a Tricolor Marking Policer | **235**



Applying Tricolor Marking Policers to Firewall Filters | 238

Applying Firewall Filter Tricolor Marking Policers to Interfaces | 239

Example: Configuring and Applying a Single-Rate Tricolor Marking Policer | 239

Configuring Single-Rate Tricolor Marking | 241

Configuring Two-Rate Tricolor Marking | 245

Example: Configuring and Verifying Two-Rate Tricolor Marking | 249

Requirements | 249

Overview | 249

Configuration | 250

Verification | 259

Applying Firewall Filter Tricolor Marking Policers to Interfaces | 260

Policer Overhead to Account for Rate Shaping in the Traffic Manager | 262

Policer Overhead to Account for Rate Shaping Overview | 262

Example: Configuring Policer Overhead to Account for Rate Shaping | 262

Requirements | 263

Overview | 263

Configuration | 264

Verification | 272

**Defining Forwarding Behavior with Forwarding Classes | 274**

Understanding How Forwarding Classes Assign Classes to Output Queues | 275

Default Forwarding Classes | 277

Configuring a Custom Forwarding Class for Each Queue | 281

Configuring Up to 16 Custom Forwarding Classes | 283

Enabling Eight Queues on Interfaces | 286

Assigning Multiple Forwarding Classes and Default Forwarding Classes | 288

Examples: Configuring Up to 16 Forwarding Classes | 289

Classifying Packets by Egress Interface | 291

Forwarding Policy Options Overview | 294

Configuring CoS-Based Forwarding | 296

Example: Configuring CoS-Based Forwarding | 299



Example: Configuring CoS-Based Forwarding for Different Traffic Types	303
Example: Configuring CoS-Based Forwarding for IPv6	304
Applying Forwarding Classes to Interfaces	305
Understanding Queuing and Marking of Host Outbound Traffic	306
Forwarding Classes and Fabric Priority Queues	308
Default Routing Engine Protocol Queue Assignments	309
Assigning Forwarding Class and DSCP Value for Routing Engine-Generated Traffic	313
Example: Writing Different DSCP and EXP Values in MPLS-Tagged IP Packets	315
Change the Default Queuing and Marking of Host Outbound Traffic	318
Example: Configure Different Queuing and Marking Defaults for Outbound Routing Engine and Distributed Protocol Handler Traffic	320
Requirements	320
Overview	320
Configuration	321
Verification	326
Overriding the Input Classification	330
<b>Defining Output Queue Properties with Schedulers  </b>	<b>332</b>
How Schedulers Define Output Queue Properties	332
Default Schedulers Overview	336
Configuring Schedulers	338
Configuring Scheduler Maps	338
Applying Scheduler Maps Overview	339
Applying Scheduler Maps to Physical Interfaces	340
Configuring Traffic Control Profiles for Shared Scheduling and Shaping	340
Configuring an Input Scheduler on an Interface	343
Understanding Interface Sets	344
Configuring Interface Sets	345



Interface Set Caveats | **348**

Configuring Internal Scheduler Nodes | **350**

Example: Configuring and Applying Scheduler Maps | **351**

Requirements | **351**

Overview | **351**

Configuration | **352**

Verification | **354**

**Controlling Bandwidth with Scheduler Rates | 356**

Oversubscribing Interface Bandwidth | **356**

Configuring Scheduler Transmission Rate | **369**

Providing a Guaranteed Minimum Rate | **372**

PIR-Only and CIR Mode | **377**

Excess Rate and Excess Priority Configuration Examples | **378**

Controlling Remaining Traffic | **385**

Bandwidth Sharing on Nonqueuing Packet Forwarding Engines Overview | **389**

Configuring Rate Limits on Nonqueuing Packet Forwarding Engines | **390**

Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs | **391**

Example: Applying Scheduler Maps and Shaping Rate to DLCIs | **401**

Requirements | **401**

Overview | **402**

Configuration | **402**

Example: Applying Scheduling and Shaping to VLANs | **407**

Requirements | **407**

Overview | **408**

Configuration | **408**

Applying a Shaping Rate to Physical Interfaces Overview | **417**

Configuring the Shaping Rate for Physical Interfaces | **417**

Example: Limiting Egress Traffic on an Interface Using Port Shaping for CoS | **419**

Requirements | **419**



- Overview | 419
- Configuration | 420
- Verification | 426

Configuring Input Shaping Rates for Both Physical and Logical Interfaces | 428

## **Setting Transmission Order with Scheduler Priorities and Hierarchical Scheduling | 430**

Priority Scheduling Overview | 430

Configuring Schedulers for Priority Scheduling | 433

Associating Schedulers with Fabric Priorities | 434

Hierarchical Class of Service Overview | 436

Hierarchical Class of Service Network Scenarios | 439

Understanding Hierarchical Scheduling | 441

Priority Propagation in Hierarchical Scheduling | 444

Hierarchical CoS for Metro Ethernet Environments | 448

Hierarchical Schedulers and Traffic Control Profiles | 449

Example: Building a Four-Level Hierarchy of Schedulers | 451

Hierarchical Class of Service for Network Slicing | 457

Configuring Ingress Hierarchical CoS | 461

## **Controlling Congestion with Scheduler RED Drop Profiles, Buffers, PFC, and ECN | 464**

RED Drop Profiles for Congestion Management | 464

- Manage Congestion with RED Drop Profiles and Packet Loss Priorities | 465
- Configure RED Drop Profiles to Define Packet Drop or ECN Behaviors | 467
- Platform-Specific RED Drop Profile Behavior | 469

Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers | 471

Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows | 473

Mapping PLP to RED Drop Profiles | 475

Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size | 478

- Configuring Large Delay Buffers for Slower Interfaces | 481
- Configuring the Maximum Delay Buffer for NxDS0 Interfaces | 486



Example: Configuring Large Delay Buffers for Slower Interfaces | 488

Example: Configuring the Delay Buffer Value for a Scheduler | 489

Example: Configuring the Physical Interface Shaping Rate | 492

Complete Configuration | 493

Enabling and Disabling the Memory Allocation Dynamic per Queue | 494

Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy | 496

Example: Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy | 498

Requirements | 499

Overview | 499

Configuration | 499

Understanding PFC Using DSCP at Layer 3 for Untagged Traffic | 502

Configuring DSCP-based PFC for Layer 3 Untagged Traffic | 504

PFC Watchdog | 506

PFC Watchdog Overview | 506

How to Configure PFC Watchdog | 507

Enable PFC Watchdog | 507

Detection | 508

Mitigation | 509

Restoration | 509

Verification | 509

Monitoring PFC Watchdog | 510

CoS Explicit Congestion Notification | 511

Example: Configuring Static and Dynamic ECN | 522

Requirements | 523

Overview | 523

Configuration | 525

Verification | 529

**Altering Outgoing Packet Headers Using Rewrite Rules | 531**

Rewriting Packet Headers to Ensure Forwarding Behavior | 532

Applying Default Rewrite Rules | 535

Configuring Rewrite Rules | 538



Configuring Rewrite Rules Based on PLP	540
Applying IEEE 802.1p Rewrite Rules to Dual VLAN Tags	541
Applying IEEE 802.1ad Rewrite Rules to Dual VLAN Tags	543
Rewriting IEEE 802.1p Packet Headers with an MPLS EXP Value	544
Setting IPv6 DSCP and MPLS EXP Values Independently	546
Configuring DSCP Values for IPv6 Packets Entering the MPLS Tunnel	547
Setting Ingress DSCP Bits for Multicast Traffic over Layer 3 VPNs	549
Applying Rewrite Rules to Output Logical Interfaces	550
Rewriting MPLS and IPv4 Packet Headers	553
Rewriting the EXP Bits of All Three Labels of an Outgoing Packet	558
Defining a Custom Frame Relay Loss Priority Map	560
Example: Per-Node Rewriting of EXP Bits	562
Example: Rewriting CoS Information at the Network Border to Enforce CoS Strategies	563
Requirements	564
Overview	564
Configuration	566
Verification	575
Example: Remarking Diffserv Code Points to MPLS EXPs to Carry CoS Profiles Across a Service Provider's L3VPN MPLS Network	577
Requirements	578
Overview	578
Configuration	581
Verification	607
Example: Remarking Diffserv Code Points to 802.1P PCPs to Carry CoS Profiles Across a Service Provider's VPLS Network	609
Requirements	610
Overview	610
Configuration	614
Verification	638
Assigning Rewrite Rules on a Per-Customer Basis Using Policy Maps	640



- Policy Maps Overview | 640
- Configuring Policy Maps | 642
- Platform-Specific Policy Map Behavior | 645

#### Host Outbound Traffic IEEE802.1p Rewrite | 646

- Configuring a Global Default IEEE 802.1p Value for All Host Outbound Traffic | 646
- Apply Egress Interface Rewrite Rules to the IEEE 802.1p Field for All Host Outbound Traffic on an Interface | 647
- Platform-Specific Host Outbound Traffic Rewrite Behavior | 649

### Altering Class of Service Values in Packets Exiting the Network Using IPv6 DiffServ | 650

#### Resources for CoS with DiffServ for IPv6 | 651

#### System Requirements for CoS with DiffServ for IPv6 | 651

#### Terms and Acronyms for CoS with DiffServ for IPv6 | 652

- classifier | 652
- class of service | 652
- Differentiated Services (DiffServ) | 652
- Differentiated Services code point (DSCP) | 652

#### Default DSCP Mappings | 652

#### Default Forwarding Classes | 655

#### Juniper Networks Default Forwarding Classes | 659

#### Roadmap for Configuring CoS with IPv6 DiffServ | 661

#### Configuring a Firewall Filter for an MF Classifier on Customer Interfaces | 662

#### Applying the Firewall Filter to Customer Interfaces | 663

#### Assigning Forwarding Classes to Output Queues | 664

#### Configuring Rewrite Rules | 665

#### DSCP IPv6 Rewrites and Forwarding Class Maps | 666

#### Applying Rewrite Rules to an Interface | 667

#### Configuring RED Drop Profiles | 667

#### Configuring BA Classifiers | 668

#### Applying a BA Classifier to an Interface | 669



Configuring a Scheduler | 670

Configuring Scheduler Maps | 671

Applying a Scheduler Map to an Interface | 671

Example: Configuring DiffServ for IPv6 | 672

## Configuring Platform-Specific Functionality

**Configuring Class of Service on ACX Series Universal Metro Routers | 694**

CoS on ACX Series Routers Features Overview | 694

Understanding CoS CLI Configuration Statements on ACX Series Routers | 697

DSCP Propagation and Default CoS on ACX Series Routers | 700

Configuring CoS on ACX Series Routers | 704

Classifiers and Rewrite Rules at the Global, Physical, and Logical Interface Levels Overview | 705

Configuring Classifiers and Rewrite Rules at the Global and Physical Interface Levels | 707

Applying DSCP and DSCP IPv6 Classifiers on ACX Series Routers | 709

Schedulers Overview for ACX Series Routers | 710

Shared and Dedicated Buffer Memory Pools on ACX Series Routers | 714

CoS for PPP and MLPPP Interfaces on ACX Series Routers | 717

CoS for NAT Services on ACX Series Routers | 730

Hierarchical Class of Service in ACX Series Routers | 731

Storm Control on ACX Series Routers Overview | 751

**Configuring Class of Service on MX Series 5G Universal Routing Platforms | 753**

Junos CoS on MX Series 5G Universal Routing Platforms Overview | 753

CoS Features and Limitations on MX Series Routers | 754

Configuring and Applying IEEE 802.1ad Classifiers | 758

Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | 760

Example: Performing Output Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | 762

| Requirements | 762



- Overview | 763
- Configuration | 763
- Verification | 777

CoS-Based Interface Counters for IPv4 or IPv6 Aggregate on Layer 2 | 781

Enabling a Timestamp for Ingress and Egress Queue Packets | 783

**Configuring Class of Service on PTX Series Packet Transport Routers | 785**

CoS Features and Limitations on PTX Series Routers | 785

CoS Feature Differences Between PTX Series Packet Transport Routers and T Series Routers | 790

Understanding Scheduling on PTX Series Routers | 793

Virtual Output Queues on PTX Series Packet Transport Routers | 797

- Introduction to Virtual Output Queues on PTX Series Packet Transport Routers | 797
- Understanding How VOQ Works on PTX Series Routers | 802
- Fabric Scheduling and Virtual Output Queues on PTX Series Routers | 805
- Understanding the Packet Forwarding Engine Fairness and Virtual Output Queue Process | 807
- VOQ Queue-depth Monitoring | 809

Example: Configuring Excess Rate for PTX Series Packet Transport Routers | 813

- Requirements | 813
- Overview | 813
- Configuration | 814
- Verification | 820

Identifying the Source of RED Dropped Packets on PTX Series Routers | 821

Configuring Queuing and Shaping on Logical Interfaces on PTX Series Routers | 829

Example: Configuring Queuing and Shaping on Logical Interfaces in PTX Series Packet Transport Routers | 833

- Requirements | 834
- Overview | 834
- Configuration | 834

Example: Configuring Strict-Priority Scheduling on a PTX Series Router | 842

- Requirements | 842
- Overview | 842
- Configuration | 844



Verification | 850

CoS Support on EVPN VXLANs | 853

Understanding CoS on VXLAN Interfaces | 854

Configuring CoS on VXLAN Interfaces | 855

Implementing CoS on VXLAN Interfaces (Junos OS Evolved) | 858

Platform-Specific CoS on EVPN-VxLANs Behavior | 859

Understanding CoS CLI Configuration Statements on PTX Series Routers | 862

Classification Based on Outer Header of Decapsulation Tunnel | 866

Overview of Classification Based on Outer Header of Decapsulation Tunnel | 866

Configuring Classification Based on Outer Header of Decapsulation Tunnel | 867

## Configuring Line Card-Specific and Interface-Specific Functionality

Feature Support of Line Cards and Interfaces | 871

CoS Features of the Router Hardware, PIC, MIC, and MPC Interface Families | 871

Scheduling on the Router Hardware, PIC, MIC, and MPC Interface Families | 872

Schedulers on the Router Hardware, PIC, MIC, and MPC Families | 873

Queuing Parameters for the Router Hardware, PIC, MIC, and MPC Interface Families | 874

Configuring Class of Service for Tunnels | 876

CoS for Tunnels Overview | 876

Tunneling and BA Classifiers | 878

Configuring CoS for GRE and IP-IP Tunnels | 878

Example: Configuring CoS for GRE or IP-IP Tunnels | 879

Requirements | 880

Overview | 880

Configuration | 881

Configuring Router A | 883

Configuring Router B | 888

Configuring Router C | 889

Verification | 890

Configuring a GRE Tunnel to Copy ToS Bits to the Outer IP Header | 891

Configuring Class of Service on Services PICs | 892



CoS on Services PICs Overview | **892**

Configuring CoS Rules on Services PICs | **894**

    Configuring Match Conditions in a CoS Rule | **896**

    Configuring Actions in a CoS Rule | **898**

        Configuring Application Profiles | **898**

        Configuring Reflexive and Reverse CoS Actions | **899**

Configuring CoS Rule Sets on Services PICs | **901**

Example: Configuring CoS Rules on Services PICs | **902**

Packet Rewriting on Services Interfaces | **904**

Multiservices PIC ToS Translation | **904**

Fragmentation by Forwarding Class Overview | **905**

Configuring Fragmentation by Forwarding Class | **906**

Configuring Drop Timeout Interval for Fragmentation by Forwarding Class | **907**

Example: Configuring Fragmentation by Forwarding Class | **910**

    Requirements | **910**

    Overview | **910**

    Configuration | **910**

    Verification | **914**

Allocating Excess Bandwidth Among Frame Relay DLCIs on Multiservices PICs | **916**

Configuring Rate Limiting and Sharing of Excess Bandwidth on Multiservices PICs | **918**

**Configuring Class of Service on IQ and Enhanced IQ (IQE) PICs | 922**

CoS on Enhanced IQ PICs Overview | **922**

Calculation of Expected Traffic on IQE PIC Queues | **923**

Configuring the Junos OS to Support Eight Queues on IQ Interfaces for T Series and M320 Routers | **950**

BA Classifiers and ToS Translation Tables | **951**

Configuring ToS Translation Tables | **952**

Configuring Hierarchical Layer 2 Policers on IQE PICs | **957**



Configuring Excess Bandwidth Sharing on IQE PICs | 959

Configuring Rate-Limiting Policers for High Priority Low-Latency Queues on IQE PICs | 966

Applying Scheduler Maps and Shaping Rate to Physical Interfaces on IQ PICs | 970

Examples: Applying a Shaping Rate | 971

Shaping Rate Calculations | 971

Example: Applying a Shaping Rate to a Clear-Channel T1 Interface on a Channelized T1 IQ PIC | 972

Example: Applying a Shaping Rate to a Clear-Channel E1 Interface on a Channelized E1 IQ PIC | 974

Examples: Applying a Scheduler Map and Shaping Rate to Physical Interfaces on IQ PICs | 976

Example: Applying a Scheduler Map and Shaping Rate to a DS0 Channel of a Channelized T1 Interface on a Channelized T1 IQ PIC | 976

Example: Applying a Scheduler Map and Shaping Rate to DS0 Channels of a Channelized E1 Interface on a Channelized E1 IQ PIC | 978

Applying a Scheduler Map and Shaping Rate to a Clear-Channel T3 Interface on a Channelized DS3 IQ PIC | 982

Applying a Scheduler Map and Shaping Rate to Fractional T1 Interfaces on a Channelized DS3 IQ PIC | 984

Applying a Scheduler Map and Shaping Rate to a DS0 Channel of a T1 Interface in a Channelized T3 Interface on a Channelized DS3 IQ PIC | 989

Applying Scheduler Maps to Chassis-Level Queues | 991

Applying Custom Schedulers to Packet Forwarding Component Queues | 993

Examples: Scheduling Packet Forwarding Component Queues | 994

Example: Applying a Chassis Scheduler Map to a 2-Port IQ PIC | 994

Example: Configuring Two T3 Interfaces on a Channelized DS3 IQ PIC | 996

Example: Applying Normal Schedulers to Two T3 Interfaces | 998

Example: Applying a Chassis Scheduler to Two T3 Interfaces | 1001

**Configuring Class of Service on Ethernet IQ2 and Enhanced IQ2 PICs | 1004**

CoS on Enhanced IQ2 PICs Overview | 1005

CoS Features and Limitations on IQ2 and IQ2E PICs (M Series and T Series) | 1006

Differences Between Gigabit Ethernet IQ and Gigabit Ethernet IQ2 PICs | 1007

Shaping Granularity Values for Enhanced Queuing Hardware | 1010

Ethernet IQ2 PIC RTT Delay Buffer Values | 1013



Configuring BA Classifiers for Bridged Ethernet	1013
Setting the Number of Egress Queues on IQ2 and Enhanced IQ2 PICs	1016
Configuring the Number of Schedulers per Port for Ethernet IQ2 PICs	1017
Applying Scheduler Maps to Chassis-Level Queues	1019
Applying Custom Schedulers to Packet Forwarding Component Queues	1021
Examples: Scheduling Packet Forwarding Component Queues	1022
Example: Applying a Chassis Scheduler Map to a 2-Port IQ PIC	1022
Example: Configuring Two T3 Interfaces on a Channelized DS3 IQ PIC	1025
Example: Applying Normal Schedulers to Two T3 Interfaces	1027
Example: Applying a Chassis Scheduler to Two T3 Interfaces	1029
CoS for L2TP Tunnels on Ethernet Interface Overview	1031
Configuring CoS for L2TP Tunnels on Ethernet Interfaces	1033
Configuring LNS CoS for Link Redundancy	1034
Example: Configuring L2TP LNS CoS Support for Link Redundancy	1035
Requirements	1035
Overview	1036
Configuration	1037
Verification	1038
Configuring Shaping on 10-Gigabit Ethernet IQ2 PICs	1040
Configuring Per-Unit Scheduling for GRE Tunnels Using IQ2 and IQ2E PICs	1042
Understanding Burst Size Configuration on IQ2 and IQ2E Interfaces	1044
Configuring Burst Size for Shapers on IQ2 and IQ2E Interfaces	1045
Configuring a CIR and a PIR on Ethernet IQ2 Interfaces	1046
Example: Configuring Shared Resources on Ethernet IQ2 Interfaces	1049
Configuring and Applying IEEE 802.1ad Classifiers	1053
Configuring Rate Limits to Protect Lower Queues on IQ2 and Enhanced IQ2 PICs	1055
Simple Filters Overview	1057
Configuring a Simple Filter	1058



## **Configuring Class of Service on 10-Gigabit Ethernet LAN/WAN PICs with SFP+ | 1060**

CoS on 10-Gigabit Ethernet LAN/WAN PIC with SFP+ Overview | **1060**

BA and Fixed Classification on 10-Gigabit Ethernet LAN/WAN PIC with SFP+ Overview | **1061**

DSCP Rewrite for the 10-Gigabit Ethernet LAN/WAN PIC with SFP+ | **1062**

Configuring DSCP Rewrite for the 10-Gigabit Ethernet LAN/WAN PIC | **1065**

Queuing on 10-Gigabit Ethernet LAN/WAN PICs Properties | **1066**

Mapping Forwarding Classes to CoS Queues on 10-Gigabit Ethernet LAN/WAN PICs | **1067**

Scheduling and Shaping on 10-Gigabit Ethernet LAN/WAN PICs Overview | **1068**

Example: Configuring Shaping Overhead on 10-Gigabit Ethernet LAN/WAN PICs | **1069**

## **Configuring Class of Service on Enhanced Queuing DPCs | 1071**

Enhanced Queuing DPC CoS Properties | **1071**

Configuring Rate Limits on Enhanced Queuing DPCs | **1074**

Configuring WRED on Enhanced Queuing DPCs | **1076**

Configuring MDRR on Enhanced Queuing DPCs | **1078**

Configuring Excess Bandwidth Sharing | **1081**

Configuring Customer VLAN (Level 3) Shaping on Enhanced Queuing DPCs | **1086**

Simple Filters Overview | **1089**

Configuring Simple Filters on Enhanced Queuing DPCs | **1090**

Configuring a Simple Filter | **1092**

## **Configuring Class of Service on MICs, MPCs, and MLCs | 1094**

CoS Features and Limitations on MIC and MPC Interfaces | **1095**

Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview | **1098**

Verifying the Number of Dedicated Queues Configured on MIC and MPC Interfaces | **1101**

Scaling of Per-VLAN Queuing on Non-Queuing MPCs | **1102**

Increasing Available Bandwidth on Rich-Queuing MPCs by Bypassing the Queuing Chip | **1108**

Flexible Queuing Mode | **1110**



- Flexible Queuing Mode Overview | **1110**
- Upgrading non-HQoS MPCs to Support Flexible Queuing | **1111**
- Disabling Flexible Queuing for non-HQoS MPCs to Optimize Power Utilization | **1112**

Multifield Classifier for Ingress Queuing on MX Series Routers with MPC | **1113**

Example: Configuring a Filter for Use as an Ingress Queuing Filter | **1114**

- Requirements | **1115**
- Overview | **1115**
- Configuration | **1115**

Ingress Queuing Filter with Policing Functionality | **1118**

- Understanding the Ingress Queuing Policing Filter | **1118**
- Example: Configuring a Filter for Use as an Ingress Queuing Policing Filter | **1119**
  - Requirements | **1119**
  - Overview | **1119**
  - Configuration | **1120**

Ingress Rate Limiting on MX Series Routers with MPCs | **1124**

Rate Shaping on MIC and MPC Interfaces | **1126**

Per-Priority Shaping on MIC and MPC Interfaces Overview | **1128**

Example: Configuring Per-Priority Shaping on MIC and MPC Interfaces | **1134**

Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates | **1142**

Example: Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates | **1143**

Traffic Burst Management on MIC and MPC Interfaces Overview | **1146**

Understanding Hierarchical Scheduling for MIC and MPC Interfaces | **1150**

Configuring Ingress Hierarchical CoS on MIC and MPC Interfaces | **1152**

Configuring a CoS Scheduling Policy on Logical Tunnel Interfaces | **1154**

Per-Unit Scheduling and Hierarchical Scheduling for MPC Interfaces | **1157**

Managing Dedicated and Remaining Queues for Static CoS Configurations on MIC and MPC Interfaces | **1160**

- Configuring the Maximum Number of Queues for MIC and MPC Interfaces | **1160**
- Configuring Remaining Common Queues on MIC and MPC Interfaces | **1161**



Excess Bandwidth Distribution on MIC and MPC Interfaces Overview	1162
Bandwidth Management for Downstream Traffic in Edge Networks Overview	1163
Scheduler Delay Buffering on MIC and MPC Interfaces	1165
Managing Excess Bandwidth Distribution on Static Interfaces on MICs and MPCs	1166
Drop Profiles on MIC and MPC Interfaces	1169
Intelligent Oversubscription on MIC and MPC Interfaces Overview	1170
Jitter Reduction in Hierarchical CoS Queues	1172
Example: Reducing Jitter in Hierarchical CoS Queues	1175
Requirements	1175
Overview	1175
Configuration	1176
Verification	1181
CoS on Ethernet Pseudowires in Universal Edge Networks Overview	1183
CoS Scheduling Policy on Logical Tunnel Interfaces Overview	1184
Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks	1184
CoS for L2TP LNS Inline Services Overview	1186
Configuring Static CoS for an L2TP LNS Inline Service	1187
CoS on Circuit Emulation ATM MICs Overview	1190
Configuring CoS on Circuit Emulation ATM MICs	1192
Understanding IEEE 802.1p Inheritance push and swap from a Transparent Tag	1194
Configuring IEEE 802.1p Inheritance push and swap from the Transparent Tag	1195
CoS on Application Services Modular Line Card Overview	1198
<b>Configuring Class of Service on Aggregated, Channelized, and Gigabit Ethernet Interfaces  </b>	<b>1201</b>
Limitations on CoS for Aggregated Interfaces	1201
Policer Support for Aggregated Ethernet Interfaces Overview	1203
Understanding Schedulers on Aggregated Interfaces	1205



Examples: Configuring CoS on Aggregated Interfaces | 1205

Hierarchical Schedulers on Aggregated Ethernet Interfaces Overview | 1209

Configuring Hierarchical Schedulers on Aggregated Ethernet Interfaces | 1210

Example: Configuring Scheduling Modes on Aggregated Interfaces | 1211

Enabling VLAN Shaping and Scheduling on Aggregated Interfaces | 1217

Class of Service on demux Interfaces | 1219

Overview of CoS on demux Interfaces | 1220

Configuring CoS on demux Interfaces | 1220

Example: Configuring Per-Unit Schedulers for Channelized Interfaces | 1223

Applying Layer 2 Policers to Gigabit Ethernet Interfaces | 1226

5

## Configuration Statements and Operational Commands

Junos CLI Reference Overview | 1230



# About This Guide

Use this guide to understand and configure class of service (CoS) features in Junos OS to define service levels that provide different delay, jitter, and packet loss characteristics to particular applications served by specific traffic flows. Applying CoS features to each device in your network ensures quality of service (QoS) for traffic throughout your entire network. This guide applies to all Juniper routing devices and the EX9200 line of switches.

## RELATED DOCUMENTATION

---

[Day One: Junos QoS for IOS Engineers](#)

---

[Day One: Deploying Basic QoS](#)

---

[Advanced Junos CoS Troubleshooting Cookbook](#)



# 1

PART

## Overview

---

Understanding How Class of Service Manages Congestion and Defines Traffic Forwarding Behavior | 2

---



## CHAPTER 1

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

## IN THIS CHAPTER

- Understanding How Class of Service Manages Congestion and Controls Service Levels in the Network | 2
- How CoS Applies to Packet Flow Across a Network | 5
- The Junos OS CoS Components Used to Manage Congestion and Control Service Levels | 6
- Mapping CoS Component Inputs to Outputs | 10
- Default Junos OS CoS Settings | 14
- Packet Flow Through the Junos OS CoS Process Overview | 18
- Configuring Basic Packet Flow Through the Junos OS CoS Process | 23
- Example: Classifying All Traffic from a Remote Device by Configuring Fixed Interface-Based Classification | 32
- Interface Types That Do Not Support Junos OS CoS | 40

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

### IN THIS SECTION

- CoS Applications | 3
- CoS Standards | 4

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](#) | 6

## How CoS Applies to Packet Flow Across a Network

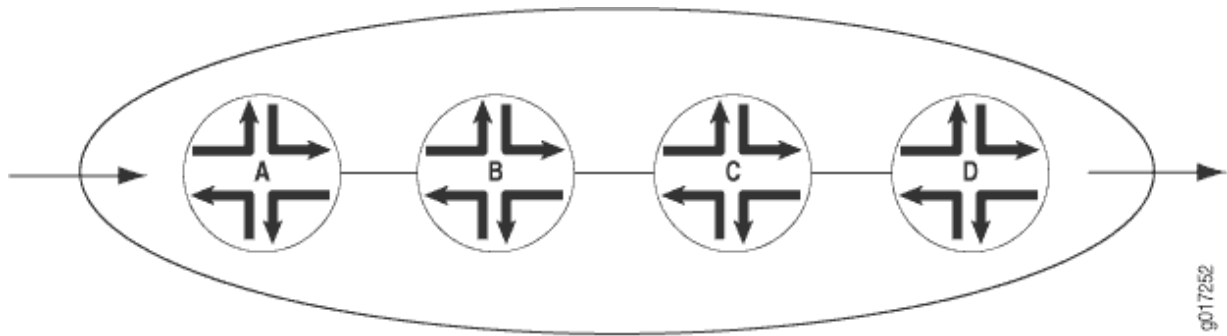
CoS works by examining traffic entering at the edge of your network. The edge routing devices classify traffic into defined service groups to provide the special treatment of traffic across the network. For example, voice traffic can be sent across certain links, and data traffic can use other links.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

To support CoS, you must configure each 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 6](#), Router A is receiving traffic from a customer network. As each packet enters, Router A examines the packet's current CoS settings and classifies the traffic into one of the groupings defined by the Internet service provider (ISP). This definition allows Router A to prioritize its resources for servicing the traffic streams it is receiving. In addition, Router A might alter the CoS settings (forwarding class and loss priority) of the packets to better match the ISP's traffic groups. When Router B receives the packets, it examines the CoS settings, determines the appropriate traffic group, and processes the packet according to those settings. It then transmits the packets to Router C, which performs the same actions. Router D also examines the packets and determines the appropriate group. Because Router D sits at the far end of the network, the ISP might decide once again to alter the CoS settings of the packets before Router D transmits them to the neighboring network.



**Figure 1: Packet Flow Across the Network**



#### RELATED DOCUMENTATION

[The Junos OS CoS Components Used to Manage Congestion and Control Service Levels](#) | 6

## 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.

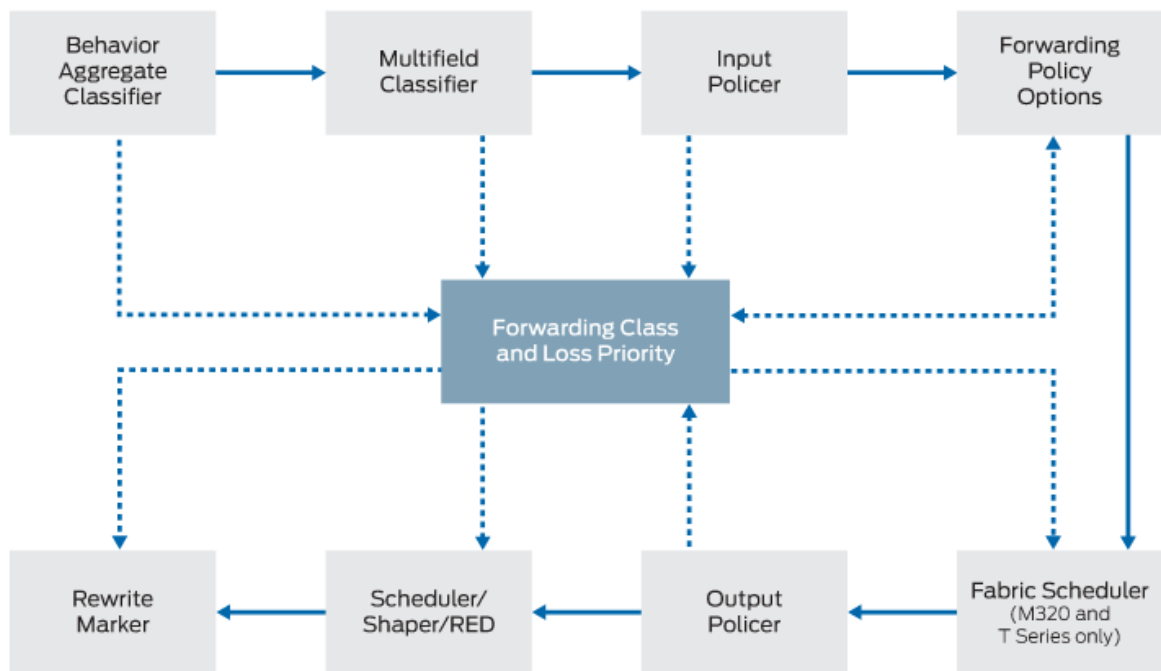
DiffServ specifications establish a six-bit field in the IPv4 and IPv6 packet header to indicate the service class that should be applied to the packet. The bit values in the DiffServ field form DiffServ code points (DSCPs) that can be set by the application or a router on the edge of a DiffServ-enabled 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 7](#) 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 44](#) 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 132](#) 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 275](#) 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 473 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 294 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 332 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* 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 532](#) for more information on rewrite rules.

## RELATED DOCUMENTATION

[Understanding How Class of Service Manages Congestion and Controls Service Levels in the Network | 2](#)

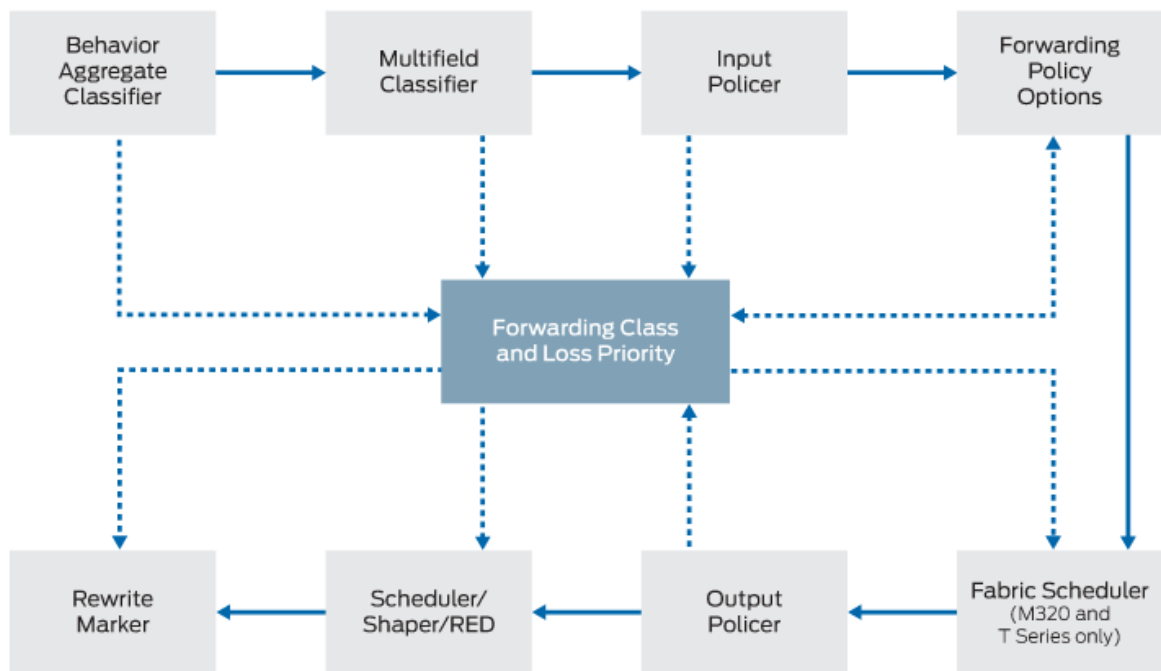
## Mapping CoS Component Inputs to Outputs

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

[Figure 3 on page 11](#) shows the components of the Junos OS CoS features, illustrating the sequence in which they interact.



Figure 3: Packet Flow Through CoS-Configurable Components



8017213





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

When you configure a mapping, you set the outputs for a given set of inputs, as shown in [Table 1 on page 12](#).

**Table 1: CoS Mappings—Inputs and Outputs**

CoS Mappings	Inputs	Outputs	Comments
classifiers	code-points	forwarding-class loss-priority	The map sets the forwarding class and PLP for a specific set of code points.
drop-profile-map	loss-priority protocol	drop-profile	The map sets the drop profile for a specific PLP and protocol type.
scheduler-maps	forwarding-class	scheduler	This map assigns a forwarding class to a specific scheduler.
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 63](#) 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 471](#) 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 338](#) 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 538](#) 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 | 44](#)

[Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers | 471](#)

[Configuring Scheduler Maps | 338](#)

[Applying Default Rewrite Rules | 535](#)

## 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

<a href="#">Default Forwarding Classes   277</a>
<a href="#">Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic   44</a>
<a href="#">Default Schedulers Overview   336</a>
<a href="#">Forwarding Classes and Fabric Priority Queues   308</a>



## Packet Flow Through the Junos OS CoS Process Overview

### IN THIS SECTION

- [Packet Flow Within Routers Overview](#) | 22

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" on page 952](#).

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 20](#) and [Figure 5 on page 21](#) show the components of the Junos OS CoS features, illustrating the sequence in which they interact.



Figure 4: CoS Classifier, Queues, and Scheduler

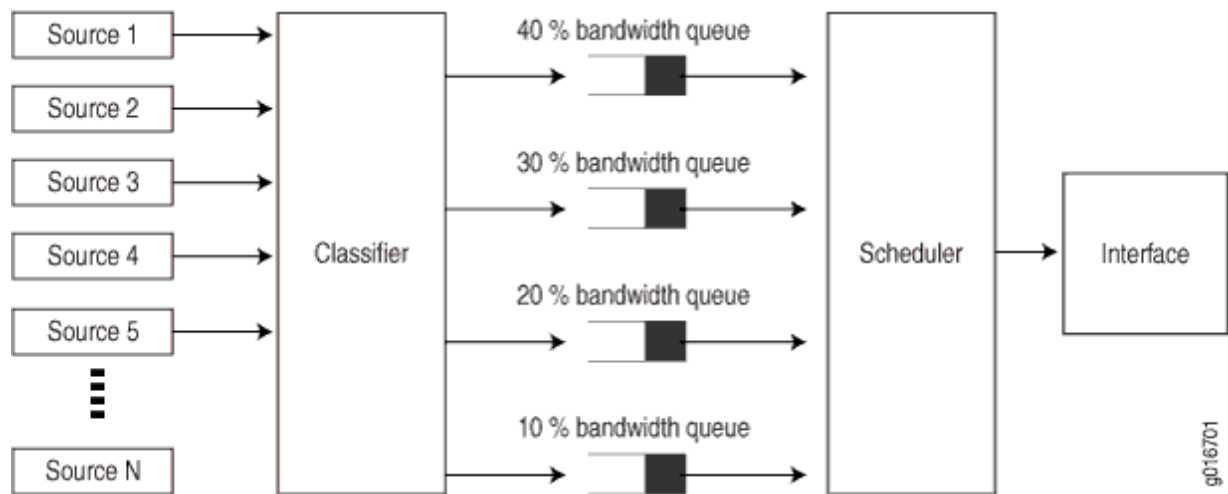
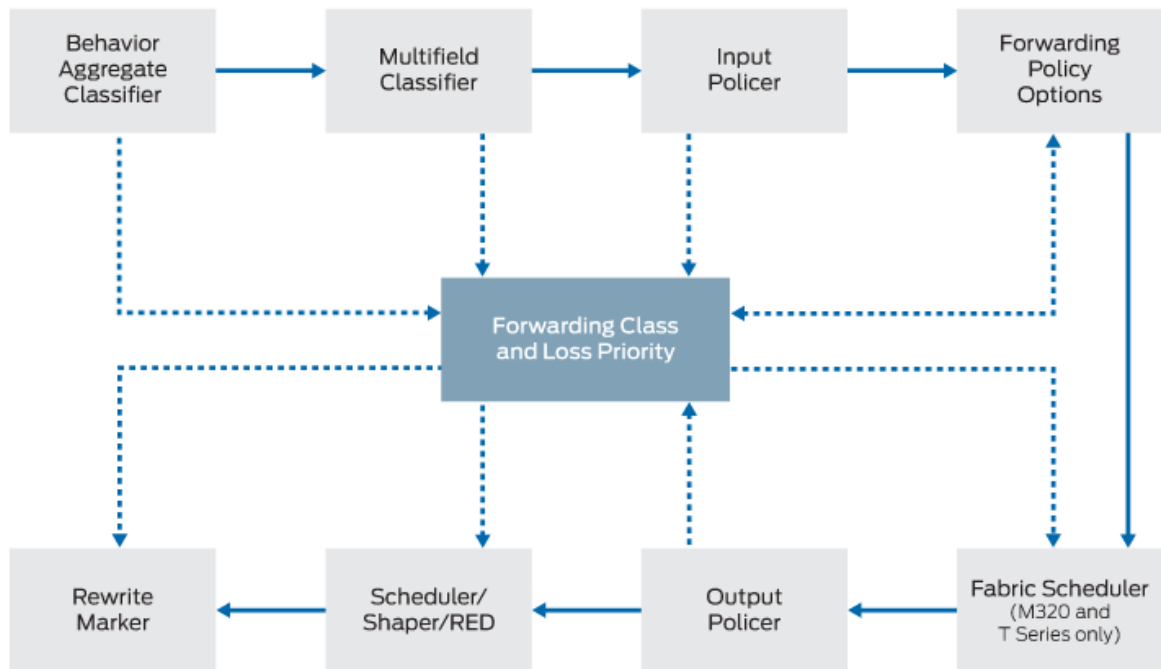




Figure 5: Packet Flow Through CoS- Configurable Components





Each outer box in [Figure 5 on page 21](#) 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](#) | 23



## Configuring Basic Packet Flow Through the Junos OS CoS Process

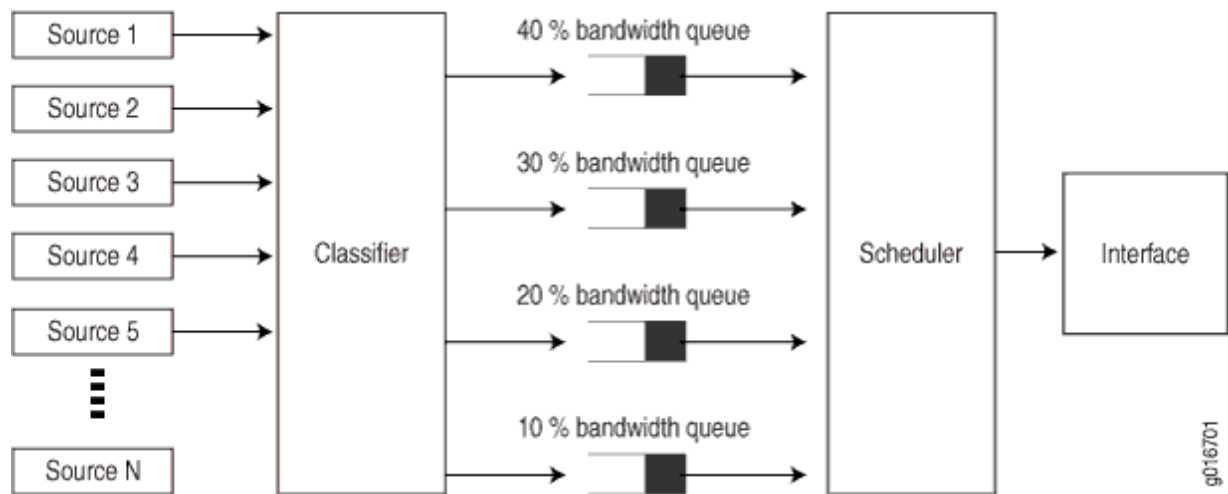
### IN THIS SECTION

- [Define Classifiers | 26](#)
- [Apply Classifiers to Incoming Packets on Interfaces | 27](#)
- [Define Policers to Limit Traffic and Control Congestion | 28](#)
- [Define Drop Profiles | 29](#)
- [Assign Each Forwarding Class to a Queue | 29](#)
- [Define Schedulers | 29](#)
- [Define Scheduler Maps | 30](#)
- [Define CoS Header Rewrite Rules | 31](#)
- [Apply Scheduler Maps and Rewrite Rules to Egress Interfaces | 31](#)

[Figure 6 on page 24](#) and [Figure 7 on page 25](#) show the components of the Junos OS CoS features, illustrating the sequence in which they interact.



Figure 6: CoS Classifier, Queues, and Scheduler









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

## Define Classifiers

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

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

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

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



```

        accept;
    }
}
}
}

```

## SEE ALSO

[Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 44](#)

[Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields | 132](#)

## Apply Classifiers to Incoming Packets on Interfaces

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

```

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

```

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

```

[edit interfaces]
fe-0/0/2 {
  unit 0 {

```



```

        family inet {
            filter {
                input classify;
            }
            address 10.12.0.13/30;
        }
    }
}

```

## Define Policers to Limit Traffic and Control Congestion

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

```

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

```

## SEE ALSO

| *Controlling Network Access Using Traffic Policing Overview*



## Define Drop Profiles

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

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

## Assign Each Forwarding Class to a Queue

To provide differentiated services to each forwarding class, assign each forwarding class to 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](#) | 275

## Define Schedulers

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

```
[edit class-of-service]
schedulers { #
  data-scheduler {
    buffer-size percent 50;
```



```

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

```

## SEE ALSO

| [How Schedulers Define Output Queue Properties](#) | 332

## Define Scheduler Maps

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

```

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

```



**SEE ALSO**

| [Configuring Scheduler Maps](#) | 338

**Define CoS Header Rewrite Rules**

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

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

**SEE ALSO**

| [Rewriting Packet Headers to Ensure Forwarding Behavior](#) | 532

**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 | 339](#)
- [Applying Rewrite Rules to Output Logical Interfaces | 550](#)

Change History Table

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

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

RELATED DOCUMENTATION

- [Packet Flow Through the Junos OS CoS Process Overview | 18](#)

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

IN THIS SECTION

- [Requirements | 33](#)
- [Overview | 33](#)
- [Configuration | 35](#)
- [Verification | 38](#)



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

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



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

## Overview

### IN THIS SECTION

- [Topology | 34](#)

A fixed interface classifier is the simplest way to classify all packets from a specific interface to a forwarding class. You typically use this approach on edge routers to classify all traffic from a remote 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;
```

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

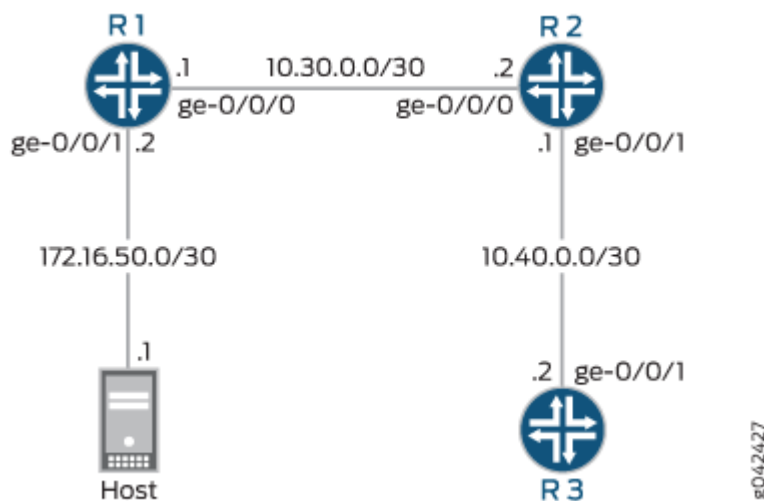


**Video:** [Class of Service Basics, Part 2: Classification Learning Byte](#)

## Topology

Figure 8 on page 34 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 35 shows the configuration for all of the Juniper Networks devices in Figure 8 on page 34. The section "Step-by-Step Procedure" on page 36 describes the steps on Device R2.

## Configuration

### IN THIS SECTION

- Procedure | 35

### Procedure

#### CLI Quick Configuration

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

#### Device R1

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

#### Device R2

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



```
set class-of-service forwarding-classes queue 2 Voice
set class-of-service forwarding-classes queue 3 NC
set class-of-service interfaces ge-0/0/0 unit 0 forwarding-class Voice
```

### Device R3

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

## Step-by-Step Procedure

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

To enable the default DSCP 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-of-service  
forwarding-classes {  
    queue 0 BE-data;  
    queue 1 Premium-data;  
    queue 2 Voice;  
    queue 3 NC;  
}  
interfaces {  
    ge-0/0/0 {  
        unit 0 {  
            forwarding-class Voice;  
        }  
    }  
}
```

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

## Verification

### IN THIS SECTION

- [Verifying a Fixed-Interface Classifier](#) | 39

Confirm that the configuration is working properly.



## Verifying a Fixed-Interface Classifier

### Purpose

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

### Action

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

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

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

This example uses the packet generator hping.

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

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



```
len=46 ip=10.40.0.2 ttl=62 id=8646 sport=0 flags=RA seq=20 win=0 rtt=7.1 ms
len=46 ip=10.40.0.2 ttl=62 id=8647 sport=0 flags=RA seq=21 win=0 rtt=2.0 ms
len=46 ip=10.40.0.2 ttl=62 id=8648 sport=0 flags=RA seq=22 win=0 rtt=1.7 ms
len=46 ip=10.40.0.2 ttl=62 id=8649 sport=0 flags=RA seq=23 win=0 rtt=1.8 ms
len=46 ip=10.40.0.2 ttl=62 id=8651 sport=0 flags=RA seq=24 win=0 rtt=1.8 ms
```

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

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

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

## Meaning

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

## RELATED DOCUMENTATION

[Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 44](#)

[Default Aliases for CoS Value Bit Patterns Overview | 56](#)

## 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.



# 2

PART

## Configuring Class of Service

---

Assigning Service Levels with Behavior Aggregate Classifiers | 43

Assigning Service Levels with Multifield Classifiers | 132

Controlling Network Access with Traffic Policing | 156

Defining Forwarding Behavior with Forwarding Classes | 274

Defining Output Queue Properties with Schedulers | 332

Controlling Bandwidth with Scheduler Rates | 356

Setting Transmission Order with Scheduler Priorities and Hierarchical Scheduling | 430

Controlling Congestion with Scheduler RED Drop Profiles, Buffers, PFC, and ECN | 464

Altering Outgoing Packet Headers Using Rewrite Rules | 531

Altering Class of Service Values in Packets Exiting the Network Using IPv6 DiffServ | 650

---



# Assigning Service Levels with Behavior Aggregate Classifiers

## IN THIS CHAPTER

- Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 44
- Default IP Precedence Classifier | 48
- Default DSCP and DSCP IPv6 Classifiers | 49
- Default MPLS EXP Classifier | 51
- Default IEEE 802.1p Classifier | 53
- Default IEEE 802.1ad Classifier | 54
- Default Aliases for CoS Value Bit Patterns Overview | 56
- Defining Aliases for CoS Value Bit Patterns | 61
- Configuring Behavior Aggregate Classifiers | 63
- Applying Behavior Aggregate Classifiers to Logical Interfaces | 66
- Example: Configuring and Applying a Default DSCP Behavior Aggregate Classifier | 71
- Example: Configuring Behavior Aggregate Classifiers | 82
- Understanding DSCP Classification for VPLS | 89
- Example: Configuring DSCP Classification for VPLS | 91
- Configuring Class of Service for MPLS LSPs | 95
- Applying DSCP Classifiers to MPLS Traffic | 99
- Applying MPLS EXP Classifiers to Routing Instances | 105
- Applying MPLS EXP Classifiers for Explicit-Null Labels | 112
- Manage Ingress Oversubscription with Traffic Class Maps | 115



## 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.

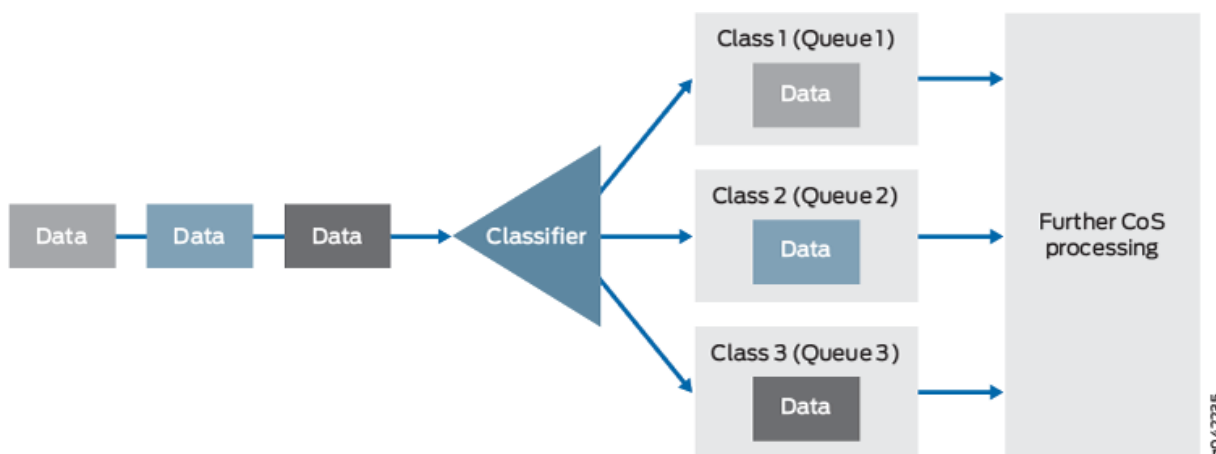
BA classification is useful if the traffic comes from a trusted source and the CoS value in the packet header is trusted. If the traffic is untrusted, multifield classifiers (see ["Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields" on page 132](#)) are used to classify 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 532](#)), 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 45](#) 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 132](#)), 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 532](#).



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

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





**NOTE:** For a specified interface, you can configure both a multifield classifier and a BA classifier without conflicts. Because the classifiers are applied in sequential order 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.

If you apply both a protocol-specific BA classifier and a protocol-independent firewall filter together, the protocol-independent filter is processed before the protocol-specific BA classifier, regardless 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 132](#). 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 305](#).

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 336](#)), is in effect regardless of any custom BA classifier or forwarding class definitions, until you configure a custom scheduler (see ["Configuring Schedulers" on page 338](#)).

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 51](#)) 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 553](#)).

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 53](#).



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

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

### Change History Table

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

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

### RELATED DOCUMENTATION

[Default IP Precedence Classifier | 48](#)

[Default DSCP and DSCP IPv6 Classifiers | 49](#)

[Default MPLS EXP Classifier | 51](#)

[Default IEEE 802.1p Classifier | 53](#)

[Default IEEE 802.1ad Classifier | 54](#)

[Configuring Behavior Aggregate Classifiers | 63](#)

[Applying Behavior Aggregate Classifiers to Logical Interfaces | 66](#)

[Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields | 132](#)

[Rewriting Packet Headers to Ensure Forwarding Behavior | 532](#)



## 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 2 on page 48](#).

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

IP Precedence Bits	Forwarding Class	Loss Priority
000	best-effort	low
001	best-effort	high
010	best-effort	low
011	best-effort	high
100	best-effort	low
101	best-effort	high
110	network-control	low
111	network-control	high

The other default IP precedence classifier (called `ipprec-default`) overrides the `ipprec-compatibility` classifier when you explicitly associate it with a logical interface. To 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 3 on page 49](#) shows the forwarding class and PLP that are assigned to the IP precedence bits when you apply the default IP precedence classifier.



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

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

**RELATED DOCUMENTATION**

| [Applying Behavior Aggregate Classifiers to Logical Interfaces](#) | 66

**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 4 on page 50 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 4: Default DSCP and DSCP IPv6 Classifiers**

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

## RELATED DOCUMENTATION

[Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 44](#)

[Default Aliases for CoS Value Bit Patterns Overview | 56](#)

[Change the Default Queuing and Marking of Host Outbound Traffic | 318](#)

*classifiers (Logical Interface)*



# 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 5 on page 51](#) lists the default MPLS classifier mapping of EXP bits to forwarding classes and loss priorities..

**Table 5: Default MPLS EXP Classification**

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



**Table 5: Default MPLS EXP Classification (Continued)**

MPLS EXP Bits	Forwarding Class	Loss Priority
110	network-control	low
111	network-control	high

Starting with Junos OS Release 21.1, PTX routers support two forwarding classes and four loss priorities for MPLS EXP default classification, as [Table 6 on page 52](#) shows.

**Table 6: Default MPLS EXP Classification for PTX Routers**

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

## RELATED DOCUMENTATION

*Configuring Class of Service for MPLS LSPs*

[Default Aliases for CoS Value Bit Patterns Overview](#) | 56



## Default IEEE 802.1p Classifier

Table 7 on page 53 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 7: Default IEEE 802.1p Classifier**

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



**Table 7: Default IEEE 802.1p Classifier (Continued)**

IEEE 802.1p CoS Value	Forwarding Class	PLP
111	network-control	high

## RELATED DOCUMENTATION

[Applying Behavior Aggregate Classifiers to Logical Interfaces](#) | 66

[Default IEEE 802.1ad Classifier](#) | 54

## Default IEEE 802.1ad Classifier

[Table 8 on page 54](#) 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 8: Default IEEE 802.1ad Classifier**

IEEE 802.1ad CoS Value	Forwarding Class	PLP
0000	best- effort	low
0001	best-effort	high
0010	best- effort	low



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

IEEE 802.1ad CoS Value	Forwarding Class	PLP
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](#) | 758



## 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 9 on page 56 shows the default mappings between the CoS values and standard aliases.

**Table 9: Default CoS Value Aliases**

Default CoS Value Alias	CoS Value
<b>DSCP and DSCP IPv6 CoS Aliases and CoS Values</b>	
ef	101110
af11	001010
af12	001100
af13	001110
af21	010010
af22	010100
af23	010110



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

Default CoS Value Alias	CoS Value
af31	011010
af32	011100
af33	011110
af41	100010
af42	100100
af43	100110
be	000000
cs1	001000
cs2	010000
cs3	011000
cs4	100000
cs5	101000
nc1/cs6	110000
nc2/cs7	111000

---

**MPLS EXP CoS Aliases and CoS Values**


---



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

Default CoS Value Alias	CoS Value
be	000
be1	001
ef	010
ef1	011
af11	100
af12	101
nc1/cs6	110
nc2/cs7	111
<b>IEEE 802.1 CoS Aliases and CoS Values</b>	
be	000
be1	001
ef	010
ef1	011
af11	100
af12	101



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

Default CoS Value Alias	CoS Value
nc1/cs6	110
nc2/cs7	111
<b>IEEE 802.1ad CoS Aliases and CoS Values</b>	
be	0000
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



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

Default CoS Value Alias	CoS Value
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 | 61](#)
[Default IP Precedence Classifier | 48](#)



[Default DSCP and DSCP IPv6 Classifiers | 49](#)

[Default MPLS EXP Classifier | 51](#)

[Default IEEE 802.1p Classifier | 53](#)

[Default IEEE 802.1ad Classifier | 54](#)

*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 | 56](#)

[Applying Behavior Aggregate Classifiers to Logical Interfaces | 66](#)

## 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 10](#).

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 multifield classifier. This setting can then be used by the appropriate drop profile map and rewrite rule. For more information, see ["Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows" on page 473](#).

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 275](#).

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 | 44](#)

[Applying Behavior Aggregate Classifiers to Logical Interfaces | 66](#)

[Enabling Tricolor Marking and Limitations of Three-Color Policers | 232](#)

## 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 10 on page 67](#) shows the supported combinations. In this table, the OSE PICs refer to the 10-port 10-Gigabit OSE PICs.

**Table 10: 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 105](#).
- 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](#) | 99

## Example: Configuring and Applying a Default DSCP Behavior Aggregate Classifier

### IN THIS SECTION

- [Requirements](#) | 72
- [Overview](#) | 72
- [Configuration](#) | 76
- [Verification](#) | 79



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

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

### IN THIS SECTION

● [Topology | 75](#)

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
<b>000000</b>	<b>best-effort</b>	<b>low</b>
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.

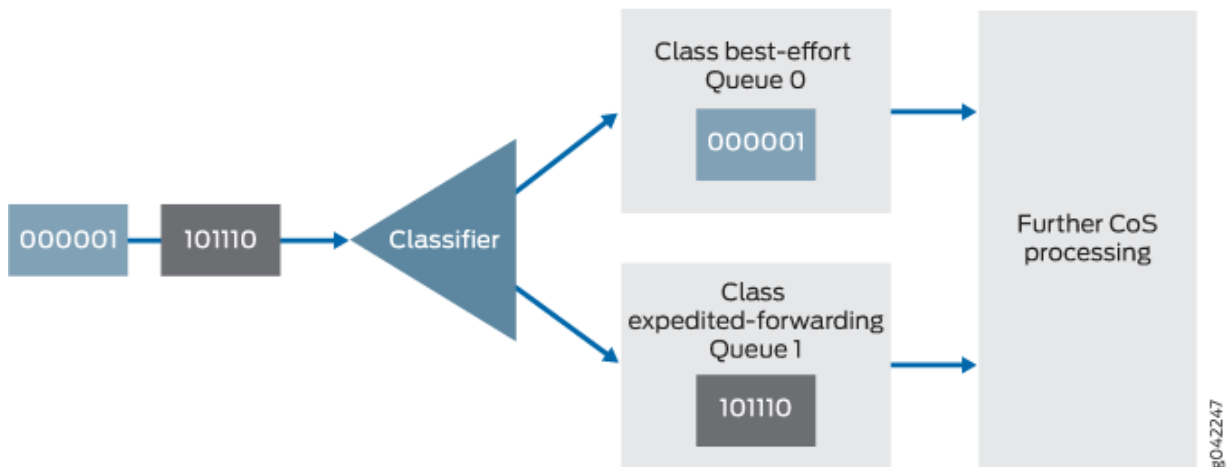
```
user@host> show class-of-service drop-profile

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 75](#). 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.



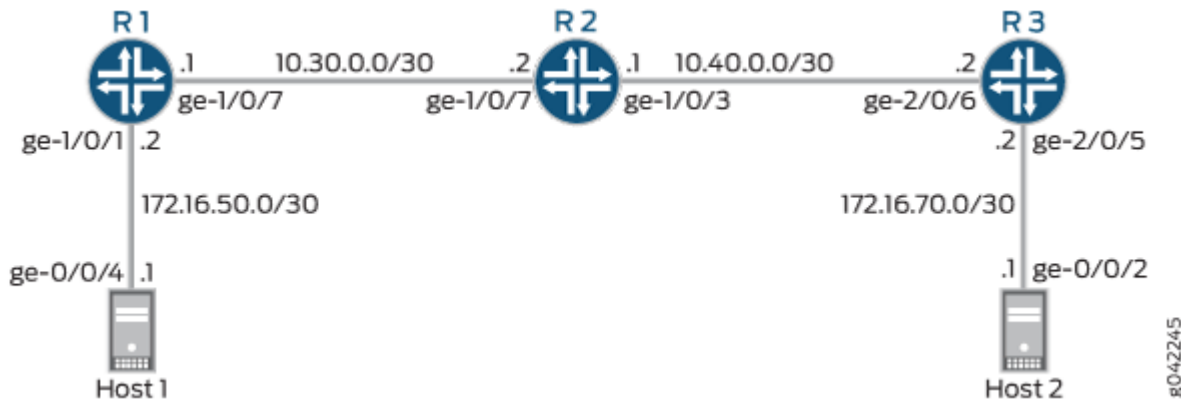
**Video:** [Class of Service Basics, Part 2: Classification Learning Byte](#)

## Topology

[Figure 11 on page 76](#) 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 77 shows the configuration for all of the Juniper Networks devices in [Figure 11](#) on page 76. The section "[Step-by-Step Procedure](#)" on page 77 describes the steps on Device R2.

## Configuration

### IN THIS SECTION

- [Procedure](#) | 77



## Procedure

### CLI Quick Configuration

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

#### Device R1

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

#### Device R2

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

#### Device R3

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

### Step-by-Step Procedure

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

To enable the default DSCP behavior aggregate classifier:

1. Configure the device interfaces.

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



2. Enable the default DSCP classifier on the interfaces.

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

## Results

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

```
user@R2# show interfaces
ge-1/0/3 {
  unit 0 {
    family inet {
      address 10.40.0.1/30;
    }
  }
}
ge-1/0/7 {
  unit 0 {
    family inet {
      address 10.30.0.2/30;
    }
  }
}
```

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



```

        classifiers {
            dscp default;
        }
    }
}

```

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

## Verification

### IN THIS SECTION

- [Verifying Behavior Aggregate Classifiers | 79](#)

Confirm that the configuration is working properly.

### Verifying Behavior Aggregate Classifiers

#### Purpose

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

#### Action

1. Clear the interface statistics on Device R2.

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

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

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

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



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

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



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

### Extended Ping Sent from Device R1

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

PING 172.16.70.1 (172.16.70.1): 56 data bytes
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
--- 172.16.70.1 ping statistics ---
25 packets transmitted, 25 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.430/0.477/0.847/0.079 ms
```

### hping Sent from Host 1

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

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



```

len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=15 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=16 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=17 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=18 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=19 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=20 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=21 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=22 win=0 rtt=0.4 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=23 win=0 rtt=0.5 ms
len=46 ip=172.16.70.1 ttl=61 DF id=0 sport=0 flags=RA seq=24 win=0 rtt=0.4 ms

```

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

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

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

Queue counters:	Queued packets	Transmitted packets	Dropped packets
0	0	0	0
1	0	0	0
<b>2</b>	<b>50</b>	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 | 44](#)

[Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows | 473](#)



## Example: Configuring Behavior Aggregate Classifiers

### IN THIS SECTION

- [Requirements | 82](#)
- [Overview | 82](#)
- [Configuration | 83](#)
- [Verification | 86](#)

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

### 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 11 on page 83](#) shows how the behavior aggregate classifier assigns loss priorities, to incoming packets in the four forwarding classes.



Table 11: Sample ba-classifier Loss Priority Assignments

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

Configuration

IN THIS SECTION

[Procedure | 83](#)

Procedure

CLI Quick Configuration

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

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



```
code-points 110001
set class-of-service interfaces ge-0/0/0 unit 0 classifiers dscp ba-classifier
```

## Step-by-Step Procedure

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

To configure behavior aggregate classifiers for a device:

1. Configure the class of service.

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

2. Configure behavior aggregate classifiers for DiffServ CoS.

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

3. Configure a best-effort forwarding class classifier.

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

4. Configure an expedited forwarding class classifier.

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

5. Configure an assured forwarding class classifier.

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



## 6. Configure a network control forwarding class classifier.

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

## 7. Apply the behavior aggregate classifier to an interface.

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



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

## Results

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

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



```

        unit 0 {
            classifiers {
                dscp ba-classifier;
            }
        }
    }
}

```

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

## Verification

### IN THIS SECTION

- [Verify the DSCP Classifier | 86](#)
- [Verifying That the Classifier Is Applied to the Interfaces | 88](#)

Confirm that the configuration is working properly.

### Verify the DSCP Classifier

#### Purpose

Make sure that the DSCP classifier is configured as expected.

#### Action

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

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

```
Classifier: ba-classifier, Code point type: dscp, Index: 10755
Code point      Forwarding class      Loss priority
000000          best-effort           low
000001          be-class             high
000010          best-effort           low
000011          best-effort           low
000100          best-effort           low
```



000101	best-effort	low
000110	best-effort	low
000111	best-effort	low
001000	best-effort	low
001001	best-effort	low
001010	assured-forwarding	low
001011	best-effort	low
<b>001100</b>	<b>af-class</b>	<b>high</b>
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
<b>101111</b>	<b>ef-class</b>	<b>high</b>



110000	network-control	low
<b>110001</b>	<b>nc-class</b>	<b>high</b>
110010	best-effort	low
110011	best-effort	low
110100	best-effort	low
110101	best-effort	low
110110	best-effort	low
110111	best-effort	low
111000	network-control	low
111001	best-effort	low
111010	best-effort	low
111011	best-effort	low
111100	best-effort	low
111101	best-effort	low
111110	best-effort	low
111111	best-effort	low

## Meaning

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

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

## Verifying That the Classifier Is Applied to the Interfaces

### Purpose

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



## Action

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

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

```
Physical interface: ge-0/0/0, Index: 144
```

```
Queues supported: 8, Queues in use: 4
```

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

```
Congestion-notification: Disabled
```

```
Logical interface: ge-0/0/0.0, Index: 333
```

Object	Name	Type	Index
Classifier	ba-classifier	dscp	10755

## Meaning

The interface is configured as expected.

## RELATED DOCUMENTATION

[Interfaces User Guide for Security Devices](#)

*Classification Overview*

*Sample Behavior Aggregate Classification*

*Understanding Packet Loss Priorities*

## 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 12 on page 90](#).

**Table 12: Default VPLS Classifiers**

MPLS Label EXP Bits	Forwarding Class/Queue
000	0
001	1
010	2
011	3
100	4
101	5



Table 12: Default VPLS Classifiers (*Continued*)

MPLS Label EXP Bits	Forwarding Class/Queue
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 281.](#)

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 | 44](#)

## Example: Configuring DSCP Classification for VPLS

### IN THIS SECTION

- [Requirements | 92](#)
- [Overview | 92](#)
- [Configuration | 92](#)



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

## 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

### IN THIS SECTION

- [CLI Quick Configuration | 92](#)
- [Configuring the DSCP Classifier for a Virtual Private LAN Service \(VPLS\) | 93](#)
- [Results | 94](#)

### 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 [Junos OS 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](#) | 89

## Configuring Class of Service for MPLS LSPs

### IN THIS SECTION

- [Class of Service for MPLS Overview](#) | 96
- [Configuring the MPLS CoS Values](#) | 96
- [Rewriting IEEE 802.1p Packet Headers with the MPLS CoS Value](#) | 99



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

## Class of Service for MPLS Overview

When IP traffic enters an LSP tunnel, the ingress router marks all packets with a CoS value, which is used to place the traffic into a transmission priority queue. On the router, for SDH/SONET and T3 interfaces, each interface has four transmit 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 router. The routers within the LSP utilize the CoS value set at the ingress router. The CoS value is encoded by means of the CoS bits (also known as the EXP or experimental bits). For more information, see [MPLS Label Allocation](#).

MPLS class of service works in conjunction with the router's general CoS functionality. 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 transmit queues are serviced by configuring weighted round-robin, and to configure congestion avoidance using random early detection (RED)..

## Configuring the MPLS CoS Values

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

- The number of the output queue into which the packet was buffered and the packet loss priority (PLP) bit are written into the MPLS header and are used as the packet's CoS value. This behavior is the default, and no configuration is required. ["Default MPLS EXP Classifier" on page 51](#) explains the default MPLS CoS values, and summarizes how the CoS values are treated.
- You set a fixed CoS value on all packets entering the LSP tunnel. A fixed CoS value means that all packets entering the LSP receive the same class of service.
- You set an MPLS EXP rewrite rule to override the default behavior.

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

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

You can include this statement at the following hierarchy levels:

- [edit protocols mpls]
- [edit protocols mpls label-switched-path *path-name*]
- [edit protocols mpls label-switched-path *path-name* primary *path-name*]
- [edit protocols mpls label-switched-path *path-name* secondary *path-name*]



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

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

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



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

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

The low-order bit of the CoS value is treated as the PLP bit and is used to select the RED drop profile to use on the output queue. If the low-order bit is 0, the non-PLP drop profile is used, and if the low-order bit is 1, the PLP drop profile is used. It is generally expected that RED will more aggressively drop packets that have the PLP bit set. For more information about RED and drop profiles, see ["RED Drop Profiles for Congestion Management" on page 464](#).





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

Table 13 on page 98 summarizes how MPLS CoS values correspond to the transmit queue and PLP bit. Note that in MPLS, the mapping between the CoS bit value and the output queue is hard-coded. You cannot configure the mapping for MPLS; you can configure it only for IPv4 traffic flows, as described in "Understanding How Forwarding Classes Assign Classes to Output Queues" on page 275.

**Table 13: MPLS CoS Values**

MPLS CoS Value	Bits	Transmit Queue	PLP Bit
0	000	0	Not set
1	001	0	Set
2	010	1	Not set
3	011	1	Set
4	100	2	Not set
5	101	2	Set
6	110	3	Not set
7	111	3	Set

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

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

1. Specify the CoS value

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



## Rewriting IEEE 802.1p Packet Headers with the MPLS CoS Value

For Ethernet interfaces installed on a T Series router or an M320 router with a peer connection to an M Series router or a T Series router, you can rewrite both MPLS CoS and IEEE 802.1p values to a configured value (the MPLS CoS values are also known as the EXP or experimental bits). Rewriting these values allows you to pass the configured value to the Layer 2 VLAN path. To rewrite both the MPLS CoS and IEEE 802.1p values, you must include the EXP and IEEE 802.1p rewrite rules in the class of service interface configuration. The EXP rewrite table is applied when you configure the IEEE 802.1p and EXP rewrite rules.

For information about how to configure the EXP and IEEE 802.1p rewrite rules, see ["Rewriting Packet Headers to Ensure Forwarding Behavior" on page 532](#).

## Applying DSCP Classifiers to MPLS Traffic

### IN THIS SECTION

- [Applying a DSCP Classifier to MPLS Packets on a Core-facing Interface | 100](#)
- [Applying a DSCP Classifier to MPLS Traffic for L3VPN/VPLS | 102](#)

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

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

The following example:

1. Configures core-facing interface ge-5/3/1.0 for protocol families IPv4, IPv6, and International Organization for Standardization Open Systems Interconnection (ISO OSI)
2. Configures the DSCP classifier dscp11.
3. Apply the DSCP classifier to the logical interface for the MPLS family.

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

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

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

2. Configure the DSCP classifier.

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

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





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

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

#### 4. Confirm the configuration.

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

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

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

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



```
    loss-priority low code-points [ cs6 cs7 ];
}
```

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

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

5. Save the configuration.

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

## 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 [Junos OS CLI User Guide](#).

The following example:

1. Configures routing instances of type either vrf or vpls.
2. Configures the DSCP classifier.
3. Attaches the classifier to the routing instance.

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

1. Configure routing instances of type 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](#) | 66

Applying MPLS EXP Classifiers to Routing Instances

IN THIS SECTION

- [Configuring and Applying Custom MPLS EXP Classifiers to Routing Instances](#) | 106
- [Applying Global Classifiers and Wildcard Routing Instances](#) | 107
- [Applying Global MPLS EXP Classifiers to Routing Instances](#) | 108
- [Applying Classifiers by Using Wildcard Routing Instances](#) | 110
- [Verifying the Classifiers Associated with Routing Instances](#) | 111

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 14 on page 105](#).

Table 14: 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



**Table 14: Default MPLS EXP Classifier (Continued)**

MPLS EXP Bits	Forwarding Class	Loss Priority
100	assured-forwarding	low
101	assured-forwarding	high
110	network-control	low
111	network-control	high



**NOTE:** At times you might need to maintain the original classifier—for example with bridge domains, where you neither want to configure a custom classifier for the routing instance nor accept the default classifier, 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;
  }
}
```

## Configuring and Applying Custom MPLS EXP Classifiers to Routing Instances

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

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

To apply a custom classifier to a routing instance:



1. Filter traffic based on the IP header.

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

2. Configure the custom MPLS EXP classifier.

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

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

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

4. Commit and confirm your configuration.

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

## 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 [Junos OS CLI User Guide](#).

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

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

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



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

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

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

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

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

4. Confirm your configuration.

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

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

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

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



```

classifiers {
    exp exp-classifier-3;
}
}

```

## 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 [Junos OS CLI User Guide](#).

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

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

```

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

```

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

```

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

```

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

```

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

```



#### 4. Commit and confirm your configuration.

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

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

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

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

### Verifying the Classifiers Associated with Routing Instances

#### IN THIS SECTION

- [Purpose | 112](#)
- [Action | 112](#)



**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
```

**Change History Table**

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

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

**RELATED DOCUMENTATION**

<a href="#">Configuring Behavior Aggregate Classifiers   63</a>
<a href="#">Default MPLS EXP Classifier   51</a>
<a href="#">Applying MPLS EXP Classifiers for Explicit-Null Labels   112</a>

**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 User 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 66](#).

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
```



2. 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.

Change History Table

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

Release	Description
18.1	Starting with Junos OS 18.1R1, PTX Series routers with third-generation FPCs (FPC3) support a new CLI option, <code>[explicit-null-cos inet/inet6]</code> at the <code>[edit forwarding-options]</code> 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 | 63
- Default MPLS EXP Classifier | 51
- Applying MPLS EXP Classifiers to Routing Instances | 105



## Manage Ingress Oversubscription with Traffic Class Maps

### SUMMARY

### IN THIS SECTION

- [Ingress Oversubscription at the Packet Forwarding Engine | 115](#)
- [Configure Traffic Class Maps | 115](#)
- [Example: Configuring Traffic Class Maps | 119](#)

### Ingress Oversubscription at the Packet Forwarding Engine

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

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

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

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



**NOTE:** Use [Feature Explorer](#) to confirm platform and release support for traffic class maps.

### Configure Traffic Class Maps

To configure a traffic class map:



1. Configure the interface. This interface needs to be associated with the configured traffic class maps.

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

2. Create a traffic class map based on CoS code points, and map the code points to a traffic class to decide the input packet priority.
  - To create a DSCP traffic class map and map the code points to a traffic class for IPv4 and IPv6 traffic, include the following statements at the [edit class-of-service] hierarchy level.

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

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

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

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

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



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

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

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

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

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

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

- Associate the DSCP traffic class map with the interface.

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

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

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



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

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

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

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

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

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



#### NOTE:

- If you do not associate the traffic class map with the configured interface, all traffic through this interface is treated with the existing fixed rule in the Packet Forwarding Engine. The existing fixed rule prioritizes network control traffic over best-effort traffic.
- As soon as you associate a traffic class map with an interface, any code points entering that interface and not included in the traffic class map are treated as best effort.
- You can associate either an IPv4 precedence traffic class map or a DSCP traffic class map with an interface. You cannot associate both these traffic class maps with a single interface. The DSCP traffic class map applies to both IPv4 and IPv6 traffic.
- You can associate either an IEEE 802.1 traffic class map or an IEEE 802.1ad traffic class map with an interface. You cannot associate both these traffic class maps with a single interface.



- An aggregated Ethernet interface bundle can have member links from both interfaces that support traffic class maps and interfaces that do not. A configured traffic class map is associated with an aggregated Ethernet bundle in following ways:
  - If an aggregated Ethernet bundle has child links only from interfaces that support traffic class maps, then the traffic class map is associated with all links of the aggregated Ethernet bundle.
  - If an aggregated Ethernet bundle has child links only from interfaces that do not support traffic class maps, then the traffic class map is not associated with the aggregated Ethernet bundle or its links.
  - If an aggregated Ethernet bundle has child links from interfaces that support traffic class maps and also interfaces that do not, the traffic class map associates only with the links from the interfaces that support traffic class maps.

## Example: Configuring Traffic Class Maps

### IN THIS SECTION

- Requirements | 119
- Overview and Topology | 120
- Configuration | 121
- Verification | 127

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



**NOTE:** Use [Feature Explorer](#) to confirm platform and release support for traffic class maps.

### Requirements

This example uses the following hardware and software components:

- One MX Series router
- One CE router

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

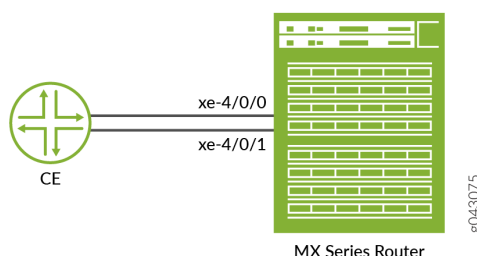


- Connected the CE router to the MX Series router.

## Overview and Topology

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

**Figure 12: Configuring Traffic Class Maps on MX Series router**



As shown in [Figure 12 on page 120](#), the CE router forwards the traffic to interface xe-4/0/0 and xe-4/0/1 on the router.

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

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

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



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

## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 121](#)
- [Configuring Interfaces | 122](#)
- [Configuring Traffic Class Maps for the Code Points and Mapping the Code Points to a Traffic Class | 123](#)
- [Associating Interfaces with Traffic Class Maps | 125](#)
- [Results | 125](#)

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

### *CLI Quick Configuration*

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

```
[edit]
set interfaces xe-4/0/0 unit 0 family inet address 198.51.100.0/24
set interfaces xe-4/0/1 vlan-tagging
set interfaces xe-4/0/1 unit 0 vlan-id 111
set interfaces xe-4/0/1 unit 0 family inet address 198.51.100.1/24
set class-of-service traffic-class-map inet-precedence inetp traffic-class real-time code-points
[ 000 001 ]
set class-of-service traffic-class-map inet-precedence inetp traffic-class network-control code-
points [ 010 011 ]
set class-of-service traffic-class-map inet-precedence inetp traffic-class best-effort code-
points [ 100 101 ]
set class-of-service traffic-class-map exp mpls_exp traffic-class real-time code-points [ 000
001 ]
set class-of-service traffic-class-map exp mpls_exp traffic-class network-control code-points
[ 010 011 ]
set class-of-service traffic-class-map exp mpls_exp traffic-class best-effort code-points [ 100
```



```

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

```

### *Configuring Interfaces*

#### **Step-by-Step Procedure**

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

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

[edit]

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



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

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

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

#### **Step-by-Step Procedure**

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

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

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

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

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



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

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

4. Create a DSCP traffic class map dscp\_v4 and map the code points 100001 100010 100011, 010011 010100 010101, and 101001 101010 101011 to the real-time, network control, and best-effort traffic classes, respectively.

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

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

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



## Associating Interfaces with Traffic Class Maps

### Step-by-Step Procedure

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

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

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

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

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

### Results

```
interfaces {
  xe-4/0/0 {
    unit 0 {
      family inet {
        address 198.51.100.0/24;
      }
    }
  }
  xe-4/0/1 {
    vlan-tagging;
    unit 0 {
      vlan-id 111;
      family inet {
        address 198.51.100.1/24;
      }
    }
  }
}
```



```

class-of-service {
  traffic-class-map {
    inet-precedence inetp {
      traffic-class real-time code-points [ 000 001 ];
      traffic-class network-control code-points [ 010 011 ];
      traffic-class best-effort code-points [ 100 101 ];
    }
    dscp dscp_v4 {
      traffic-class real-time code-points [ 100001 100010 100011 ];
      traffic-class network-control code-points [ 010011 010100 010101 ];
      traffic-class best-effort code-points [ 101001 101010 101011 ];
    }
    exp mpls_exp {
      traffic-class real-time code-points [ 000 001 ];
      traffic-class network-control code-points [ 010 011 ];
      traffic-class best-effort code-points [ 100 101 ];
    }
    ieee-802.1 802.1p {
      traffic-class real-time code-points [ 000 001 ];
      traffic-class network-control code-points [ 010 011 ];
      traffic-class best-effort code-points [ 100 101 ];
    }
    ieee-802.1ad 802.1ad {
      traffic-class real-time code-points [ 0000 0001 1000 1001 ];
      traffic-class network-control code-points [ 0010 0011 1010 1011 ];
      traffic-class best-effort code-points [ 0100 0101 1100 1101 ];
    }
  }
}

interfaces {
  xe-4/0/0 {
    traffic-class-map {
      inet-precedence inetp;
      exp mpls_exp;
      ieee-802.1 802.1p vlan-tag inner;
    }
  }
  xe-4/0/1 {
    traffic-class-map {
      dscp dscp_v4;
      ieee-802.1ad 802.1ad vlan-tag inner;
    }
  }
}

```



```
}
}
```

## Verification

### IN THIS SECTION

- [Verifying Mapping of Code Points to Input Traffic Classes | 127](#)
- [Verifying Mapping of Interfaces to Traffic Class Maps | 129](#)
- [Verifying Traffic Class Information on the Interface | 129](#)

### *Verifying Mapping of Code Points to Input Traffic Classes*

#### Purpose

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

#### Action

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

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

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

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

```
Traffic-class-map: dscp_v4, Code-point type: dscp, Index: 37469
```

Code point	Traffic class
010011	network-control
010100	network-control
010101	network-control



100001	real-time
100010	real-time
100011	real-time
101001	best-effort
101010	best-effort
101011	best-effort

Traffic-class-map: mpls\_exp, Code-point type: exp, Index: 39622

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

Traffic-class-map: 802.1p, Code-point type: ieee-802.1, Index: 13605

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

Traffic-class-map: 802.1ad, Code-point type: ieee-802.1ad, Index: 13677

Code point	Traffic class
0000	real-time
0001	real-time
0010	network-control
0011	network-control
0100	best-effort
0101	best-effort
1000	real-time
1001	real-time
1010	network-control
1011	network-control
1100	best-effort
1101	best-effort



### Meaning

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

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

### *Verifying Mapping of Interfaces to Traffic Class Maps*

### Purpose

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

### Action

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

```

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

```

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

### Meaning

The output shows that:

- Interface xe-4/0/0 is associated with the traffic class maps INET-Precedence, MPLS EXP, and IEEE-802.1.
- Interface xe-4/0/1 is associated with the traffic class maps DSCP and IEEE-802.1AD.

### *Verifying Traffic Class Information on the Interface*

### Purpose

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



## Action

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

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

```
Physical interface: xe-4/0/0, Enabled, Physical link is Up
  Interface index: 162, SNMP ifIndex: 541, Generation: 165
  Link-level type: Ethernet, MTU: 1518, MRU: 0, LAN-PHY mode, Speed: 10Gbps, BPDU Error: None,
  MAC-REWRITE Error: None, Loopback: None, Source filtering: Disabled,
  Flow control: Enabled
```

```
...
```

```
Preclassifier statistics:
```

Traffic Class	Received Packets	Transmitted Packets	Dropped Packets
real-time	3000	3000	0
network-control	2000	2000	0
best-effort	2000	1000	1000

```
Interface transmit statistics: Enabled
```

```
...
```

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

```
Physical interface: xe-4/0/1, Enabled, Physical link is Up
  Interface index: 163, SNMP ifIndex: 525, Generation: 166
  Link-level type: Ethernet, MTU: 1518, MRU: 0, LAN-PHY mode, Speed: 10Gbps, BPDU Error: None,
  MAC-REWRITE Error: None, Loopback: None, Source filtering: Disabled,
  Flow control: Enabled
```

```
...
```

```
Preclassifier statistics:
```

Traffic Class	Received Packets	Transmitted Packets	Dropped Packets
real-time	2000	2000	0
network-control	1000	1000	0
best-effort	1000	600	400

```
Interface transmit statistics: Enabled
```

```
...
```



## Meaning

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

## RELATED DOCUMENTATION

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

---

*show class-of-service traffic-class-map*



# Assigning Service Levels with Multifield Classifiers

## IN THIS CHAPTER

- Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields | 132
- Configuring Multifield Classifiers | 133
- Using Multifield Classifiers to Set Packet Loss Priority | 137
- Example: Configuring and Applying a Firewall Filter for a Multifield Classifier | 139
- Example: Classifying Packets Based on Their Destination Address | 148
- Example: Configuring and Verifying a Complex Multifield Filter | 151

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

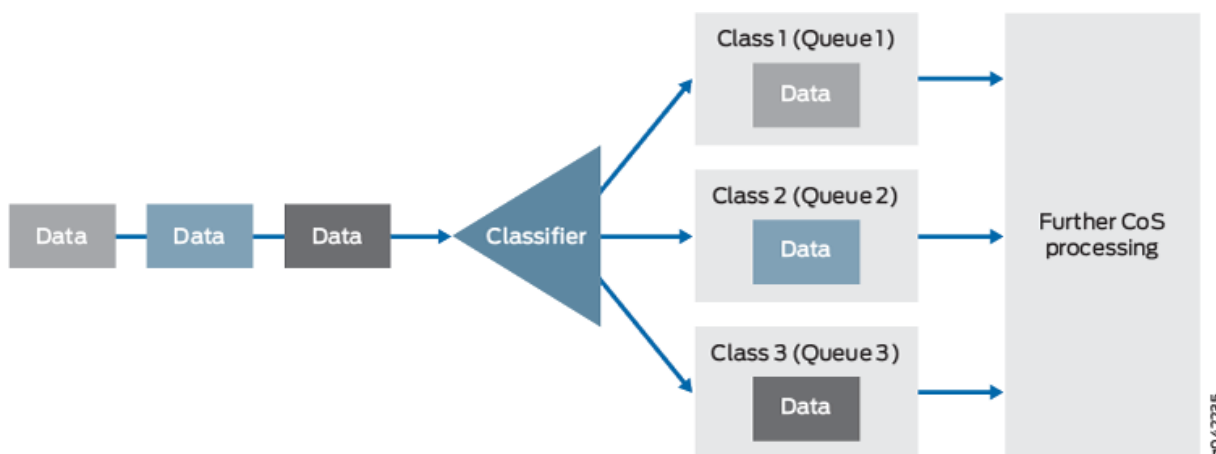
Behavior aggregate (BA) classification (see "[Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic](#)" on page 44), where packets are classified based on their QoS markings, is the most common way to assign service levels because it is straightforward and based on a well-established, 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 133](#) 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](#) | 44

[Configuring Multifield Classifiers](#) | 133

## 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](#) | 132



[Configuring a Simple Filter | 1058](#)

*Guidelines for Applying Standard Firewall Filters*

[Using Multifield Classifiers to Set Packet Loss Priority | 137](#)

## 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 multifield classifier sets the forwarding class and the PLP for each packet entering or exiting the interface with a specific destination address, source address, IP protocol, source port, destination port, or DSCP.

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

To use the classifier in this sample procedure, you must configure the settings for the `expedited-forwarding` forwarding class at the `[edit class-of-service forwarding-classes queue queue-number expedited-forwarding]` hierarchy level. For more information, see ["Understanding How Forwarding Classes Assign Classes to Output Queues" on page 275](#).

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 | 133](#)

[Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields | 132](#)

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

### IN THIS SECTION

- [Requirements | 139](#)
- [Overview | 139](#)
- [Configuration | 141](#)
- [Verification | 145](#)

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

### Requirements

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

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

### Overview

#### IN THIS SECTION

- [Topology | 140](#)



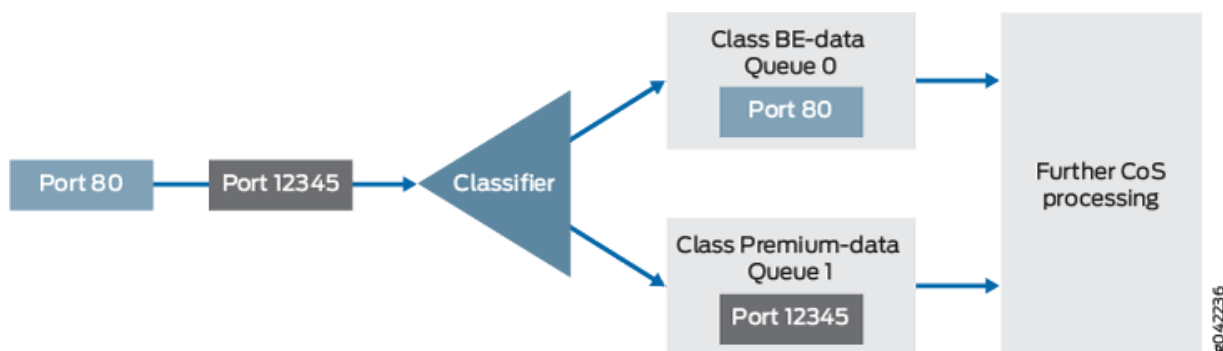
A classifier is a software operation that inspects a packet as it enters the router or switch. The packet header contents are examined, and this examination determines how the packet is treated when the network becomes too busy to handle all of the packets and you want your devices to drop packets intelligently, 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 140](#).

**Figure 14: Multifield Classifier Based on TCP Source Ports**



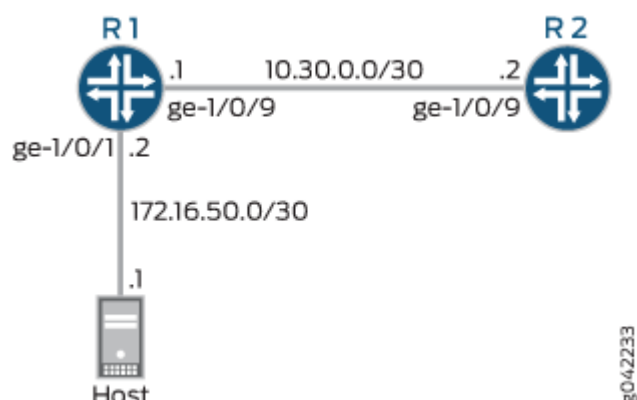
You apply the multifield classifier's firewall filter as an input filter on each customer-facing or host-facing interface that needs the filter. The incoming interface is ge-1/0/1 on Device R1. The classification and queue assignment is verified on the outgoing interface. The outgoing interface is Device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

## Topology

[Figure 15 on page 141](#) shows the sample network.



Figure 15: Multifield Classifier Scenario



"CLI Quick Configuration" on page 141 shows the configuration for all of the Juniper Networks devices in Figure 15 on page 141.

"Step-by-Step Procedure" on page 142 describes the steps on Device R1.

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



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

## Configuration

### IN THIS SECTION

- Procedure | 141

## Procedure

### CLI Quick Configuration

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

#### Device R1

```

set interfaces ge-1/0/1 description to-host
set interfaces ge-1/0/1 unit 0 family inet filter input mf-classifier

```



```

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

```

## Device R2

```

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

```

## Step-by-Step Procedure

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

To configure Device R1:

1. Configure the device interfaces.

```

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

```



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

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

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

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

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

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

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

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

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

6. Apply the firewall filter to the ge-1/0/1 interface as an input filter.

```
[edit interfaces]
user@R1# set ge-1/0/1 unit 0 family inet filter input mf-classifier
```



## Results

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

```
user@R1# show interfaces
ge-1/0/1 {
  description to-host;
  unit 0 {
    family inet {
      filter {
        input mf-classifier;
      }
      address 172.16.50.2/30;
    }
  }
}
ge-1/0/9 {
  description to-R2;
  unit 0 {
    family inet {
      address 10.30.0.1/30;
    }
  }
}
```

```
user@R1# show class-of-service
forwarding-classes {
  class BE-data queue-num 0;
  class Premium-data queue-num 1;
  class Voice queue-num 2;
  class NC queue-num 3;
}
```

```
user@R1# show firewall
family inet {
  filter mf-classifier {
    term BE-data {
```



```
        from {
            protocol tcp;
            port 80;
        }
        then forwarding-class BE-data;
    }
    term Premium-data {
        from {
            protocol tcp;
            port 12345;
        }
        then forwarding-class Premium-data;
    }
    term accept-all-else {
        then accept;
    }
}
}
```

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

## Verification

### IN THIS SECTION

- [Checking the CoS Settings | 145](#)
- [Sending TCP Traffic into the Network and Monitoring the Queue Placement | 146](#)

Confirm that the configuration is working properly.

### Checking the CoS Settings

#### Purpose

Confirm that the forwarding classes are configured correctly.



Action

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

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

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

Meaning

The output shows the configured custom classifier settings.

Sending TCP Traffic into the Network and Monitoring the Queue Placement

Purpose

Make sure that the traffic of interest is sent out the expected queue.

Action

1. Clear the interface statistics on Device R1's outgoing interface.

```
user@R1> clear interfaces statistics ge-1/0/9
```

2. Use a traffic generator to send 50 TCP port 80 packets to Device R2 or to some other downstream device.
3. On Device R1, check the queue counters.



Notice that you check the queue counters on the downstream output interface, not on the incoming interface.

```
user@R1> show interfaces extensive ge-1/0/9 | find "Queue counters"
```

Queue counters:	Queued packets	Transmitted packets	Dropped packets
0	50	50	0
1	0	57	0
2	0	0	0
3	0	0	0

4. Use a traffic generator to send 50 TCP port 12345 packets to Device R2 or to some other downstream device.

```
[root@host]# hping 172.16.60.1 -c 50 -s 12345 -k
```

5. On Device R1, check the queue counters.

```
user@R1> show interfaces extensive ge-1/0/9 | find "Queue counters"
```

Queue counters:	Queued packets	Transmitted packets	Dropped packets
0	50	50	0
1	50	57	0
2	0	0	0
3	0	0	0

## Meaning

The output shows that the packets are classified correctly. When port 80 is used in the TCP packets, queue 0 is incremented. When port 12345 is used, queue 1 is incremented.

## RELATED DOCUMENTATION

| *Example: Configuring a Two-Rate Three-Color Policer*



## Example: Classifying Packets Based on Their Destination Address

### IN THIS SECTION

- [Requirements | 148](#)
- [Overview | 148](#)
- [Configuration | 148](#)

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

### Requirements

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

### Overview

In this example you configure a multifold 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

#### IN THIS SECTION

- [CLI Quick Configuration | 149](#)
- [Configuring Firewall Filter | 149](#)
- [Results | 150](#)



## CLI Quick Configuration

To quickly configure the multifold 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 multifold classifier (firewall filter):

1. Create and configure the multifold 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 | 132](#)  
[Configuring Multifield Classifiers | 133](#)

## Example: Configuring and Verifying a Complex Multifield Filter

### IN THIS SECTION

- [Configuring a Complex Multifield Filter | 151](#)
- [Verifying a Complex Multifield Filter | 153](#)

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

### Configuring a Complex Multifield Filter

To configure the multifield filter, perform the following actions:

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

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

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

#### Classifying SIP Signaling Messages

```
firewall {
  family inet {
```



```

filter classify {
    interface-specific;
    term sip {
        from {
            protocol [ udp tcp ];
            port 5060;
        }
        then {
            forwarding-class network-control;
            accept;
        }
    }
}

```

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

#### Classifying VoIP Channels That Use UDP

```

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

```

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

#### Configuring the Policer

```

policer be-policer {
    if-exceeding {
        bandwidth-limit 1m;
        burst-size-limit 15k;
    }
}

```



```

    then loss-priority high;
}

```

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

### Policing All Remaining Traffic

```

term be {
    then policer be-policer;
}

```

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

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

### Applying the Classifier

```

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

```

## Verifying a Complex Multifield Filter

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



## Verifying Default Classification

```

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

Logical interface: fe-0/0/2.0, Index: 68
Shaping rate: 32000

```

Object	Name	Type	Index
Scheduler-map	<default>		27
Rewrite	exp-default	exp	21
Classifier	exp-default	exp	5
Classifier	ipprec-compatibility	ip	8

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

## Displaying Default Classifier Mappings

```

user@host> show class-of-service classifier name ipprec-compatibility
Classifier: ipprec-compatibility, Code point type: inet-precedence, Index: 12

```

Code point	Forwarding class	Loss priority
000	best-effort	low
001	best-effort	high
010	best-effort	low
011	best-effort	high
100	best-effort	low
101	best-effort	high
110	network-control	low
111	network-control	high

After your configuration is committed, verify that your multifold 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 multifold 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.



# Controlling Network Access with Traffic Policing

## IN THIS CHAPTER

- [Controlling Network Access Using Traffic Policing Overview | 156](#)
- [Effect of Two-Color Policers on Shaping Rate Changes | 162](#)
- [Configuring Policers Based on Logical Interface Bandwidth | 163](#)
- [Example: Limiting Inbound Traffic at Your Network Border by Configuring an Ingress Single-Rate Two-Color Policer | 167](#)
- [Example: Performing CoS at an Egress Network Boundary by Configuring an Egress Single-Rate Two-Color Policer | 179](#)
- [Example: Limiting Inbound Traffic Within Your Network by Configuring an Ingress Single-Rate Two-Color Policer and Configuring Multifield Classifiers | 192](#)
- [Example: Limiting Outbound Traffic Within Your Network by Configuring an Egress Single-Rate Two-Color Policer and Configuring Multifield Classifiers | 209](#)
- [Overview of Tricolor Marking Architecture | 229](#)
- [Enabling Tricolor Marking and Limitations of Three-Color Policers | 232](#)
- [Configuring and Applying Tricolor Marking Policers | 234](#)
- [Configuring Single-Rate Tricolor Marking | 241](#)
- [Configuring Two-Rate Tricolor Marking | 245](#)
- [Example: Configuring and Verifying Two-Rate Tricolor Marking | 249](#)
- [Applying Firewall Filter Tricolor Marking Policers to Interfaces | 260](#)
- [Policer Overhead to Account for Rate Shaping in the Traffic Manager | 262](#)

## Controlling Network Access Using Traffic Policing Overview

### IN THIS SECTION

- [Congestion Management for IP Traffic Flows | 157](#)



- [Traffic Limits | 157](#)
- [Traffic Color Marking | 159](#)
- [Forwarding Classes and PLP Levels | 160](#)
- [Policer Application to Traffic | 161](#)

## 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*

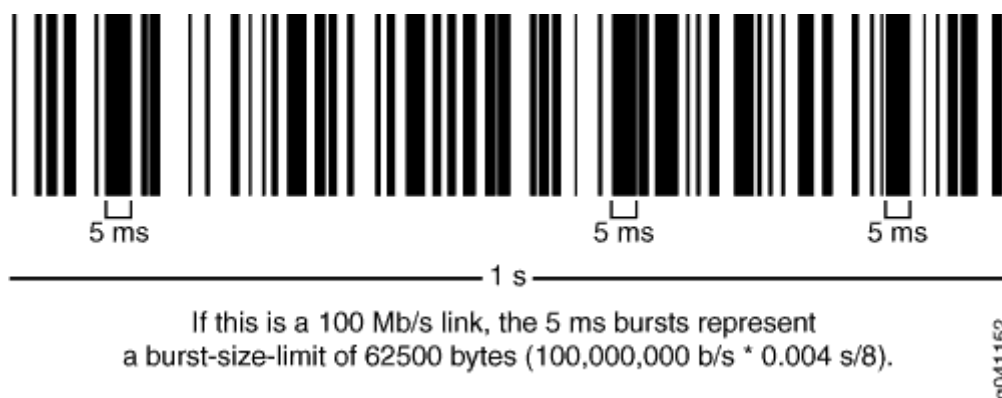


*bucket algorithm* in that you can allow a specified traffic burst before starting to discard packets or apply a penalty such as packet output-queuing priority or packet-drop priority.

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 160](#) 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
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](#) | 18

## 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](#) | 163

## 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*

*logical-bandwidth-policer*

*shaping-rate (Applying to an Interface)*

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

### IN THIS SECTION

- Requirements | 168
- Overview | 168
- Configuration | 171



## ● Verification | 177

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](http://www.juniper.net/books).

## Requirements

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

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

## Overview

### IN THIS SECTION

#### ● Topology | 170

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.

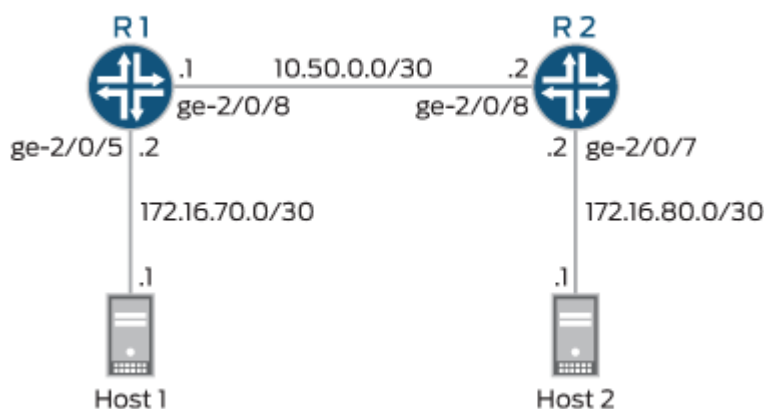


**NOTE:** You need to leave some additional bandwidth available that is not rate limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

## Topology

This example uses the topology in [Figure 17 on page 170](#).

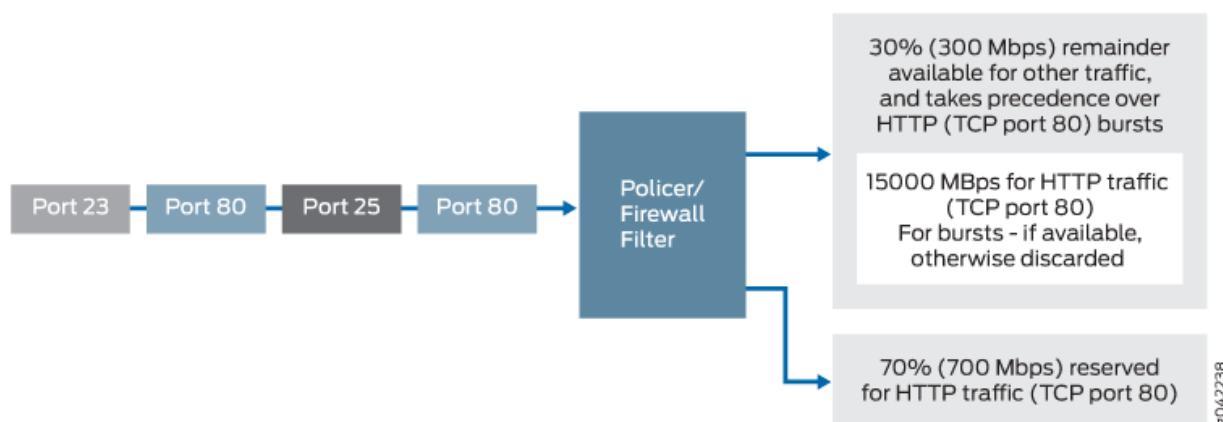
**Figure 17: Single-Rate Two-Color Policer Scenario**



[Figure 18 on page 171](#) shows the policing behavior.



Figure 18: Traffic Limiting in a Single-Rate Two-Color Policer Scenario



## Configuration

### IN THIS SECTION

- Procedure | 171

## Procedure

### CLI Quick Configuration

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

#### Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description 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 [Junos OS CLI User Guide](#).

To configure Device R1:

1. Configure the device interfaces.

```

[edit interfaces]
user@R1# set ge-2/0/5 description to-Host
user@R1# set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set ge-2/0/8 description to-R2
user@R1# set ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set lo0 unit 0 description 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

### IN THIS SECTION

- [Clearing the Counters | 177](#)
- [Sending TCP Traffic into the Network and Monitoring the Discards | 178](#)

Confirm that the configuration is working properly.

### Clearing the Counters

#### Purpose

Confirm that the firewall counters are cleared.



## Action

On Device R1, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@R1> clear firewall all
```

## Sending TCP Traffic into the Network and Monitoring the Discards

### Purpose

Make sure that the traffic of interest that is sent is rate-limited on the input interface (ge-2/0/5).

### Action

1. Use a traffic generator to send 10 TCP packets with a source port of 80.

The `-s` flag sets the source port. The `-k` flag causes the source port to remain steady at 80 instead of incrementing. The `-c` flag sets the number of packets to 10. The `-d` flag sets the packet size.

The destination IP address of 172.16.80.1 belongs to Device Host 2 that is connected to Device R2. The user on Device Host 2 has requested a webpage from Device Host 1 (the webserver emulated by the traffic generator on Device Host 1). The packets that being rate-limited are sent from Device Host 1 in response to the request from Device Host 2.



**NOTE:** In this example the policer numbers are reduced to a bandwidth limit of 8 Kbps and a burst size limit of 1500 KBps to ensure that some packets are dropped during this test.

```
[root@host]# hping 172.16.80.1 -c 10 -s 80 -k -d 300
```

```
[User@Host]# hping 172.16.80.1 -c 10 -s 80 -k -d 350
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 350 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.5 ms
.
.
.
--- 172.16.80.1 hping statistic ---
```



```
10 packets transmitted, 6 packets received, 40% packet loss
round-trip min/avg/max = 0.5/3000.8/7001.3 ms
```

2. On Device R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall
```

```
User@R1# run show firewall
```

```
Filter: __default_bpdu_filter__
```

```
Filter: mf-classifier
```

```
Policers:
```

Name	Bytes	Packets
discard-t1	1560	4

## Meaning

In Steps 1 and 2 the output from both devices shows that 4 packets were discarded. This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 Kbps burst option for red out-of-contract HTTP port 80 traffic was exceeded.

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

### IN THIS SECTION

- Requirements | 180
- Overview | 180
- Configuration | 183
- Verification | 189



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](http://www.juniper.net/books).

## Requirements

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

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

## Overview

### IN THIS SECTION

- [Topology | 182](#)

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.

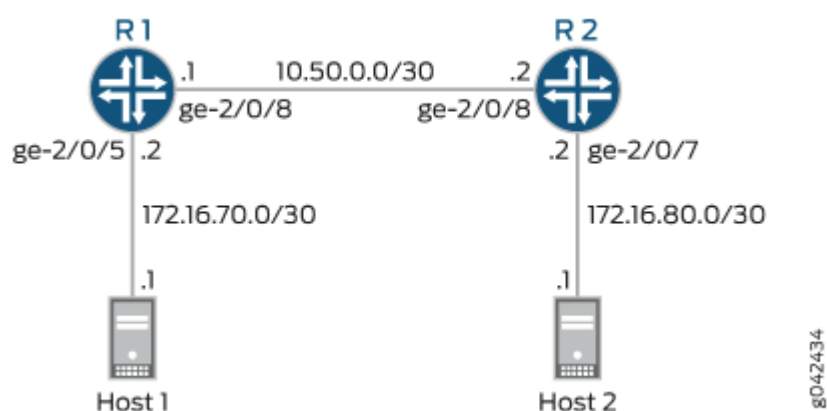


**NOTE:** You need to leave some additional bandwidth available that is not rate-limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

## Topology

This example uses the topology in [Figure 19 on page 182](#).

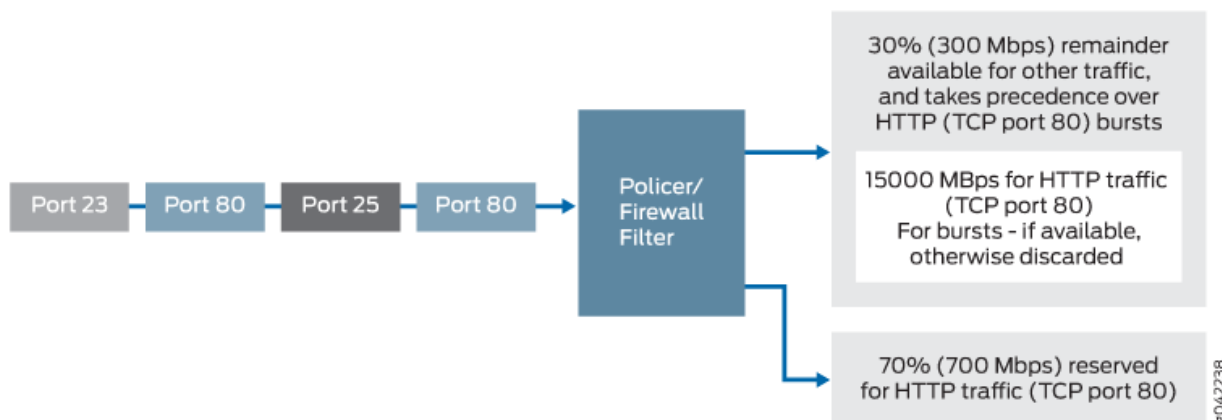
**Figure 19: Single-Rate Two-Color Policer Scenario**



[Figure 20 on page 183](#) shows the policing behavior.



Figure 20: Traffic Limiting in a Single-Rate Two-Color Policer Scenario



## Configuration

### IN THIS SECTION

- Procedure | 183

## Procedure

### CLI Quick Configuration

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

#### Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces ge-2/0/8 unit 0 family inet filter output mf-classifier
set interfaces lo0 unit 0 description 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 [Junos OS CLI User Guide](#).

To configure Device R1:

1. Configure the device interfaces.

```

[edit interfaces]
user@R1#set ge-2/0/5 description to-Host
user@R1#set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1#set ge-2/0/8 description to-R2
user@R1#set ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set lo0 unit 0 description 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 looback-interface;
        family inet {
            address 192.168.14.1/32;
        }
    }
}
}

```

```

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

```

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

## Verification

### IN THIS SECTION

- [Clearing the Counters | 190](#)
- [Sending TCP Traffic into the Network and Monitoring the Discards | 190](#)

Confirm that the configuration is working properly.



## Clearing the Counters

### Purpose

Confirm that the firewall counters are cleared.

### Action

On Device R1, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@R1> clear firewall all
```

## Sending TCP Traffic into the Network and Monitoring the Discards

### Purpose

Make sure that the traffic of interest that is sent is rate-limited on the output interface (ge-2/0/8).

### Action

1. Use a traffic generator to send 20 TCP packets with a source port of 80.

The `-s` flag sets the source port. The `-k` flag causes the source port to remain steady at 80 instead of incrementing. The `-c` flag sets the number of packets to 10. The `-d` flag sets the packet size.

The destination IP address of 172.16.80.1 represents a user that is downstream of Device R2. The user has requested a webpage from the host (the webserver emulated by the traffic generator), and the packets are sent in response to the request.



**NOTE:** In this example the policer numbers are reduced to a bandwidth limit of 8 Kbps and a burst size limit of 1500 KBps to ensure that some packets are dropped.

```
[root@host]# hping 172.16.80.1 -s 80 -k -d 375 -c 20
```

```
[root@tp-lnx03 rtwright]# hping 172.16.80.1 -s 80 -k -d 375 -c 20
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 375 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=4000.8 ms
.
```



```
.
.
--- 172.16.80.1 hping statistic ---
20 packets transmitted, 12 packets received, 40% packet loss
```

2. On Device R1, check the firewall counters by using the show firewall command.

```
user@R1> show firewall

user@sugar# run show firewall

Filter: mf-classifier
Policers:
Name                               Bytes      Packets
discard-t1                         3320        8
```

Meaning

In Steps 1 and 2 the output from both devices shows that 8 packets were discarded. This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 KBps burst option for red out-of-contract HTTP port 80 traffic was exceeded.

RELATED DOCUMENTATION

<a href="#">Routing Policies, Firewall Filters, and Traffic Policers User Guide</a>
<i>Example: Configuring a Two-Rate Three-Color Policer</i>



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

### IN THIS SECTION

- Requirements | 192
- Overview | 192
- Configuration | 196
- Verification | 204

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](http://www.juniper.net/books).

### Requirements

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

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

### Overview

#### IN THIS SECTION

- Topology | 194



## 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 21 on page 195](#).



Figure 21: Single-Rate Two-Color Policer Scenario

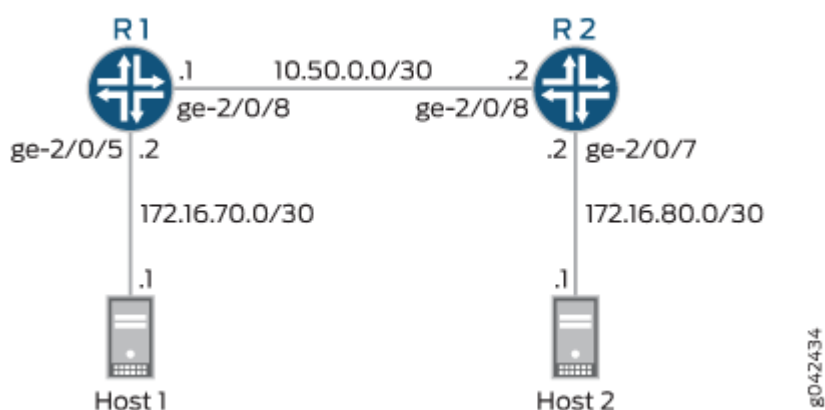
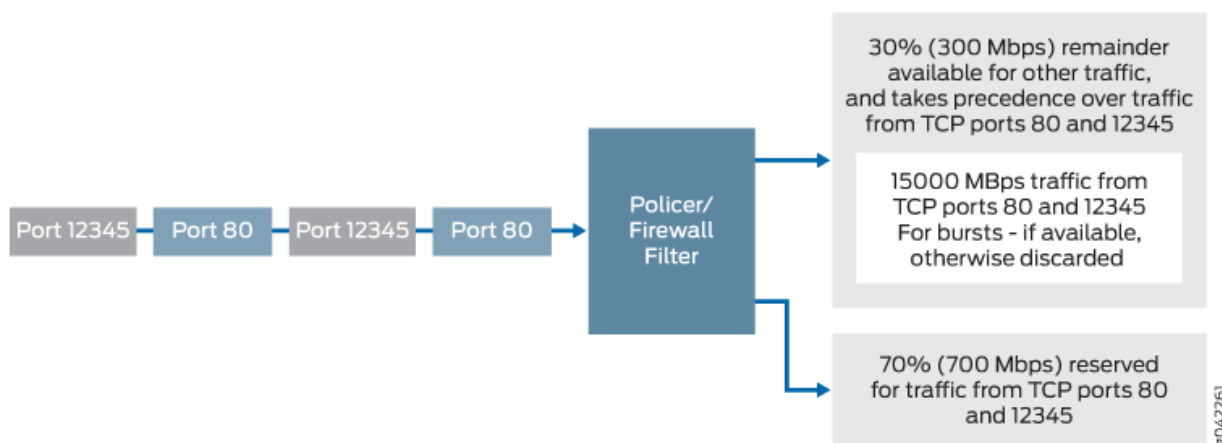


Figure 22 on page 195 shows the policing behavior.

Figure 22: 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



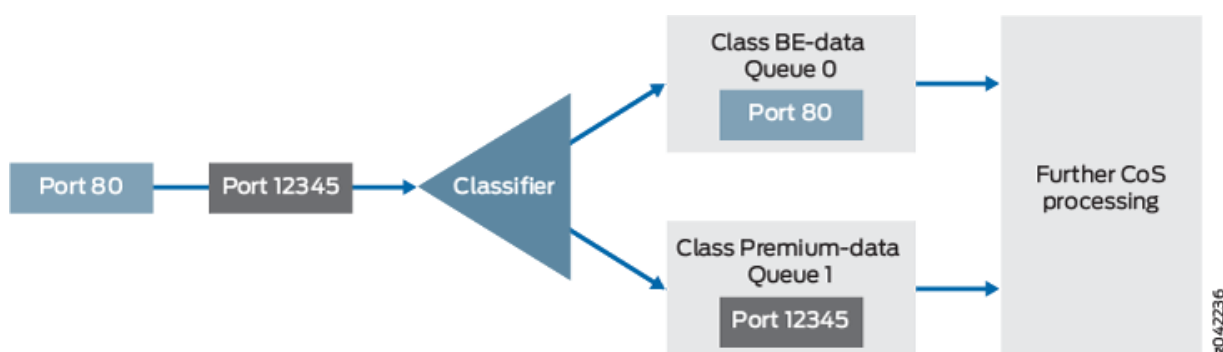
80 are classified into the BE-data forwarding class and queue number 0, and TCP packets with a source port 12345 are classified into the Premium-data forwarding class and queue number 1. Traffic from both port numbers is monitored by the policer first. If the traffic makes it through the policer, it 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 23 on page 196](#).

**Figure 23: Multifield Classifier Based on TCP Source Ports**



You apply the multifield classifier's firewall filter as an input filter on each customer-facing or host-facing interface that needs the filter. In this example, the incoming interface `ge-2/0/5` on Device R1 is used. You monitor the behavior of the queues on the interfaces that the traffic is transmitted over. In this example, to determine how the queues are being serviced, you examine the traffic statistics on interface `ge-2/0/8` by using the `extensive` option in the `show interfaces` command.

## Configuration

### IN THIS SECTION

- Procedure | 197



## Procedure

### CLI Quick Configuration

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

#### Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description 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 looback-interface
set interfaces lo0 unit 0 family inet address 192.168.14.1/32
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0

```

## Step-by-Step Procedure

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

To configure Device R1:

1. Configure the device interfaces.

```

[edit interfaces]
user@R1#set ge-2/0/5 description to-Host
user@R1#set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1#set ge-2/0/8 description to-R2
user@R1#set ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set lo0 unit 0 description looback-interface
user@R1#set lo0 unit 0 family inet address 192.168.13.1/32

```

2. Configure the policer to rate-limit to a bandwidth of 700 Mbps and a burst size of 15 KBps.

```

[edit firewall policer discard]
user@R1# set if-exceeding bandwidth-limit 700m
user@R1# set if-exceeding burst-size-limit 15k

```

3. Configure the policer to discard packets in the red traffic flow.

```

[edit firewall policer discard]
user@R1# set then discard

```



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

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

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

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

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

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

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

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

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

8. Apply the firewall filter to the ge-2/0/5 interface as an input filter.

```
[edit interfaces]
user@R1# set ge-2/0/5 unit 0 family inet filter input mf-classifier
```



## 9. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

## Step-by-Step Procedure

To configure Device R2:

### 1. Configure the device interfaces.

```
[edit interfaces]
user@R2# set ge-2/0/7 description to-Host
user@R2# set ge-2/0/7 unit 0 family inet address 172.16.80.2/30
user@R2# set ge-2/0/8 description to-R1
user@R2# set ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@R2# set lo0 unit 0 description 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;
            }
        }
    }
}

```







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

IN THIS SECTION

●

Checking the CoS Settings | 204

●

Clearing the Counters | 205

●

Sending Traffic into the Network from TCP HTTP Port 80 and Monitoring the Results | 205

●

Sending Traffic into the Network from TCP Port 12345 and Monitoring the Results | 207

Confirm that the configuration is working properly.

Checking the CoS Settings

Purpose

Confirm that the forwarding classes are configured correctly.

Action

From Device R1, run the `show class-of-service forwarding-class` command.

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

Forwarding class		ID	Queue	Restricted queue	Fabric priority
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

- 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	<b>1360</b>	<b>4</b>
discard-Premium-data	0	0

Notice that in the hping output that there was 20% packet loss (4 packets out of 20) and the same number of packets were dropped by the policer as shown in the output of the show firewall command. Also notice that the drops are associated with the queue BE-data as specified in the mf-classifier in the firewall configuration.

3. On Device R1, check the queue counters by using the show interfaces extensive ge-2/0/8| find "Queue counters" command.

```
user@R1> show interfaces extensive ge-2/0/8| find "Queue counters"
```

Queue counters:	Queued packets	Transmitted packets	Dropped packets
0	16	16	0
1	0	0	0
2	0	0	0
3	4	4	0

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



Notice that 16 packets were transmitted out interface 2/0/8 using the queue BE-data as specified in the mf-classifier in the firewall configuration. The remaining 4 packets, were dropped by the policer, as shown above. The 4 packets sent to queue 3 are network control traffic. They are possibly routing protocol updates.

## Meaning

The output from both devices shows that 4 packets were discarded. This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 Kbps burst option for red out-of-contract HTTP port 80 traffic was exceeded. In Steps 2 and 3, you can see that the correct queues were used to transmit the remaining traffic out interface 2/0/8.

## Sending Traffic into the Network from TCP Port 12345 and Monitoring the Results

### Purpose

Send traffic that can be monitored at the policer and custom queue level.

### Action

1. Clear the counters again as shown in section ["Clearing the Counters" on page 205](#).
2. Use a traffic generator to send 20 TCP packets with a source port of 12345 into the network.

The -s flag sets the source port. The -k flag causes the source port to remain steady at 12345 instead of incrementing. The -c flag sets the number of packets to 20. The -d flag sets the packet size.

```
[User@host]# hping 172.16.80.1 -c 20 -s 12345 -k -d 300

[root@tp-host]# hping 172.16.80.1 -s 12345 -k -c 20 -d 300
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 300 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.4 ms
.
.
.
--- 172.16.80.1 hping statistic ---
20 packets transmitted, 16 packets received, 20% packet loss
round-trip min/avg/max = 0.4/9126.3/18002.4 ms
```



3. On Device R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall
```

```
Filter: mf-classifier
```

```
Policers:
```

Name	Bytes	Packets
discard-BE-data	0	0
discard-Premium-data	1360	4

Notice that in the `hping` output that there was 20% packet loss (4 packets out of 20) and the same number of packets were dropped by the policer as shown in the output of the `show firewall` command. Also notice that the drops are associated with the queue Premium-data as specified in the `mf-classifier` in the firewall configuration.

4. On Device R1, check the queue counters by using the `show interfaces extensive ge-2/0/8 | find "Queue counters"` command.

```
user@R1> show interfaces extensive ge-2/0/8 | find "Queue counters"
```

Queue counters:	Queued packets	Transmitted packets	Dropped packets
0	0	0	0
1	16	16	0
2	0	0	0
3	19	19	0

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

Notice that 16 packets were transmitted out interface 2/0/8 using the Premium-data queues as specified in the `mf-classifier` firewall configuration. The remaining 4 packets were dropped by the policer, as shown above. The 19 packets sent to queue 3 are network control traffic. They are possibly routing protocol updates.

## Meaning

The output from both devices shows that 4 packets were discarded. This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 KBps burst option for red out-of-



contract HTTP port 80 traffic was exceeded. In Steps 3 and 4, you can see that the correct queues were used to transmit the remaining traffic out interface 2/0/8.

## RELATED DOCUMENTATION

[Routing Policies, Firewall Filters, and Traffic Policers User Guide](#)

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

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

### IN THIS SECTION

- [Requirements | 210](#)
- [Overview | 210](#)
- [Configuration | 215](#)
- [Verification | 224](#)

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 later.

A thorough explanation of the token bucket concept and its underlying algorithms is beyond the scope of this document. For more information about traffic policing, and CoS in general, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at [www.juniper.net/books](http://www.juniper.net/books).



## Requirements

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

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

## Overview

### IN THIS SECTION

- [Topology | 213](#)

### *Policing*

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

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

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

Or

Burst size = interface mtu x 10

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



**NOTE:** There is a finite buffer space for an interface. In general, the estimated total buffer depth for an interface is about 125 ms.



For a traffic flow that conforms to the configured limits (categorized as green traffic), packets are implicitly marked with a packet loss priority (PLP) level of `low` and are allowed to pass through the interface unrestricted.

For a traffic flow that exceeds the configured limits (categorized as red traffic), packets are handled according to the traffic-policing actions configured for the policer. This example discards packets that burst over the 15 KBps limit.

To rate-limit Layer 3 traffic, you can apply a two-color policer in the following ways:

- Directly to a logical interface, at a specific protocol level.
- As the action of a standard stateless firewall filter that is applied to a logical interface, at a specific protocol level. This is the technique used in this example.

To rate-limit Layer 2 traffic, you can apply a two-color policer as a logical interface policer only. You cannot apply a two-color policer to Layer 2 traffic through a firewall filter.

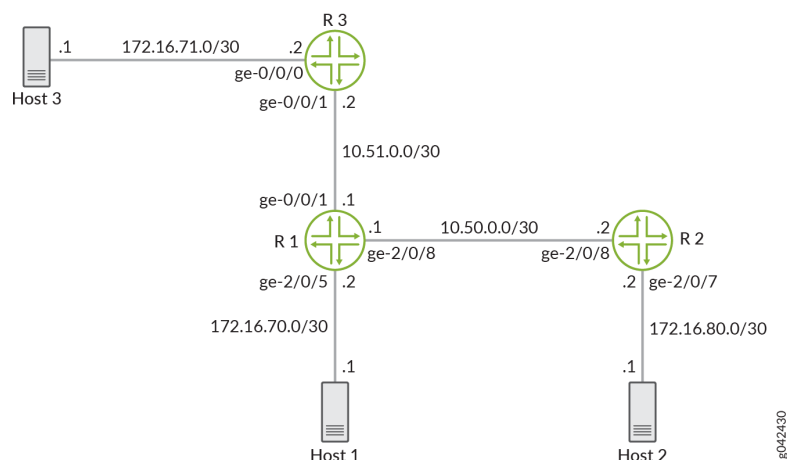


**CAUTION:** You can choose either bandwidth-limit or bandwidth percent within the policer, as they are mutually exclusive. You cannot configure a policer to use bandwidth percent for aggregate, tunnel, and software interfaces.

In this example, as illustrated in [Figure 24 on page 212](#), Host1 connected to device R1 and Host3 connected to device R3 are traffic generators emulating webserver. Both Host1 and Host3 are sending traffic to Host2 behind device R2. Devices R1, R2, and R3 are owned by a service provider. Host1 is accessed by users on Host2 behind R2. Host1 and Host2 are owned by the same customer and their traffic must be managed. Host1 will be sending traffic with a source TCP HTTP port of 80 to the users. A single-rate two-color policer is configured and applied to the interface on R1 that connects to R2. The policer enforces the contract agreed upon by the webserver owner and the service provider for the bandwidth available to the web traffic flowing between R1 and R2.



Figure 24: Single-Rate Two-Color Policer Scenario



This example applies an egress policer between R1 and R2 because this is the point where the traffic from both customer sites shares the same link. This makes it easier to enforce the required policing parameters. Trying to rate-limit the combined customer traffic on the link between R1 and R2 by applying the policers as ingress policers on interfaces `ge-0/0/0` on R3 and `ge-2/0/5` on R1 would be complicated because using the contracted rate of 700 Mbps (70 percent) of the available bandwidth with an allowable burst rate of 10 x the MTU size of the gigabit Ethernet interface between Host3 and R3 and the Host1 and R1 would result in allowing a maximum throughput of 1400 Mbps over the link between R1 and R2.

Therefore, the rate-limiting applied to the host connections between the hosts and R3 and R1 would have to be reduced below 700 Mbps. The calculation of what to reduce the rate-limit number to would be a problem because just reducing each host to 350 Mbps would mean that if one host was transmitting traffic while the other host was not transmitting, the maximum throughput on the link between R1 and R2 would be only one half of the contracted rate (350 Mbps instead of 700 Mbps). This is why this example is useful to show the amount of thought that must go into applying CoS in a network to achieve the desired goals.

According to the contractual bandwidth availability, the egress policer on R1 will limit the HTTP port 80 traffic originating from Host1 to using 700 Mbps (70 percent) of the available bandwidth with an allowable burst rate of 10 x the MTU size of the gigabit Ethernet interface between R1 and R2.

Additional traffic from TCP source port 12345 is used in this example to further illustrate how traffic is allocated to the outbound queues.





**NOTE:** In a real-world scenario you would probably also rate-limit traffic for a variety of other ports such as FTP, SFTP, SSH, TELNET, SMTP, IMAP, and POP3 because they are often included as additional services with web hosting services.

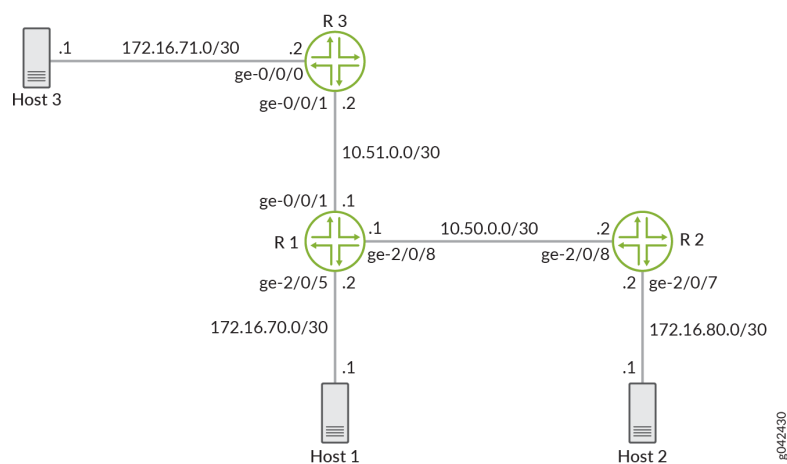


**NOTE:** You must leave some additional bandwidth available that is not rate-limited for network control protocols such as routing protocols, DNS, and any other protocols required to keep network connectivity operational. This is why the firewall filter has a final accept condition on it.

## Topology

This example uses the topology in [Figure 25 on page 213](#).

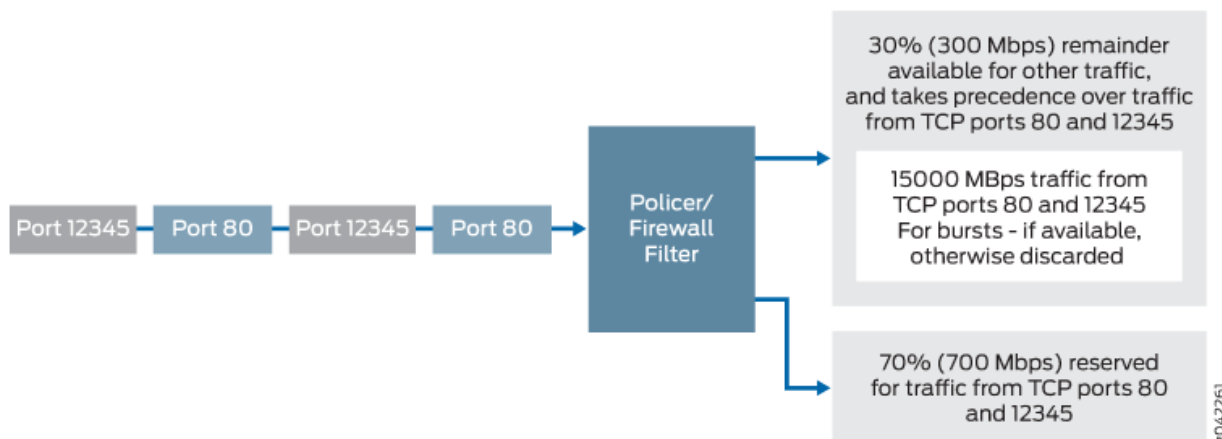
**Figure 25: Single-Rate Two-Color Policer Scenario**



[Figure 26 on page 214](#) shows the policing behavior.



Figure 26: 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.

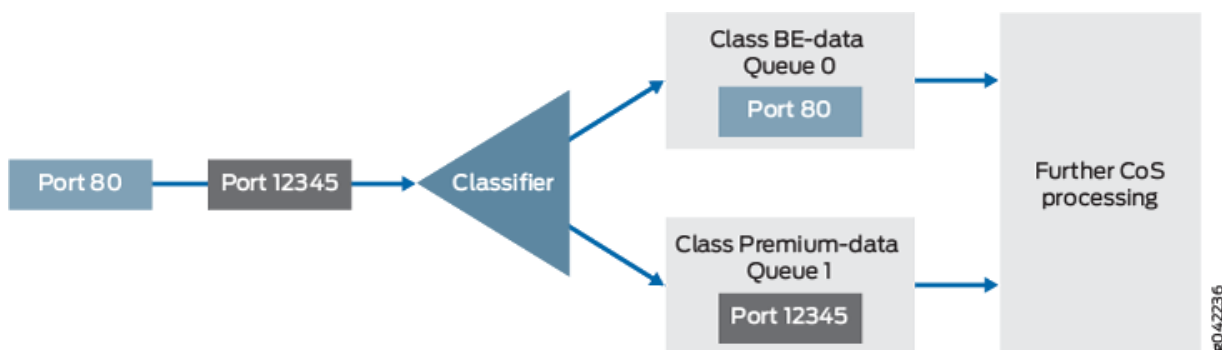
Multifield classifiers are typically used at the network edge as packets enter an autonomous system (AS). However, as explained previously in the policing section, in this example the multifield classifier is configured within the AS of the service provider.

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

The classifier operation is shown in [Figure 27 on page 215](#).



Figure 27: Multifield Classifier Based on TCP Source Ports



You monitor the behavior of the queues on the interfaces that the traffic is transmitted over. In this example, to determine how the queues are being serviced, you examine the traffic statistics on interface ge-2/0/8 on R1 by using the extensive option in the `show interfaces` command.

## Configuration

### IN THIS SECTION

- [Procedure | 215](#)

## Procedure

### CLI Quick Configuration

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

### Device R1

```
set interfaces ge-0/0/1 description to-R3
set interfaces ge-0/0/1 unit 0 family inet address 10.51.0.1/30
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces ge-2/0/8 unit 0 family inet filter output mf-classifier
set interfaces lo0 unit 0 description loopback-interface
```



```

set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set firewall policer discard if-exceeding bandwidth-limit 700m
set firewall policer discard if-exceeding burst-size-limit 15k
set firewall policer discard then discard
set class-of-service forwarding-classes class BE-data queue-num 0
set class-of-service forwarding-classes class Premium-data queue-num 1
set class-of-service forwarding-classes class Voice queue-num 2
set class-of-service forwarding-classes class NC queue-num 3
set firewall family inet filter mf-classifier term BE-data from protocol tcp
set firewall family inet filter mf-classifier term BE-data from port http
set firewall family inet filter mf-classifier term BE-data then forwarding-class BE-data
set firewall family inet filter mf-classifier term BE-data then policer discard
set firewall family inet filter mf-classifier term Premium-data from protocol tcp
set firewall family inet filter mf-classifier term Premium-data from port 12345
set firewall family inet filter mf-classifier term Premium-data then forwarding-class Premium-
data
set firewall family inet filter mf-classifier term Premium-data then policer discard
set firewall family inet filter mf-classifier term accept then accept
set protocols ospf area 0.0.0.0 interface ge-2/0/5.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0

```

## Device R2

```

set interfaces ge-2/0/7 description to-Host
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.14.1/32
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0

```

## Device R3

```

set interfaces ge-0/0/0 description to-Host
set interfaces ge-0/0/0 unit 0 family inet address 172.16.71.1/30
set interfaces ge-0/0/1 description to-R1
set interfaces ge-0/0/1 unit 0 family inet address 10.51.0.2/30

```



```

set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.15.1/32
set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 passive
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-0/0/1.0

```

## Step-by-Step Procedure

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

To configure R1:

1. Configure the device interfaces.

```

[edit interfaces]
user@R1# set ge-0/0/1 description to-R3
user@R1# set ge-0/0/1 unit 0 family inet address 10.51.0.1/30
user@R1# set ge-2/0/5 description to-Host
user@R1# set ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set ge-2/0/8 description to-R2
user@R1# set ge-2/0/8 unit 0 family inet address 10.50.0.1/30

```

2. Configure the policer to rate-limit to a bandwidth of 700 Mbps and a burst size of 15 KBps.

```

[edit firewall policer discard]
user@R1# set if-exceeding bandwidth-limit 700m
user@R1# set if-exceeding burst-size-limit 15k

```

3. Configure the policer to discard packets in the red traffic flow.

```

[edit firewall policer discard]
user@R1# set then discard

```

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

```

[edit class-of-service forwarding-classes]
user@R1# set class BE-data queue-num 0

```



```

user@R1# set class Premium-data queue-num 1
user@R1# set class Voice queue-num 2
user@R1# set class NC queue-num 3

```

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

```

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

```

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

```

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

```

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

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

```

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

```

8. Apply the firewall filter to interface ge-2/0/8 as an output filter.

```

[edit interfaces]
user@R1# set ge-2/0/8 unit 0 family inet filter output mf-classifier

```



## 9. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-0/0/1.0
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

### Step-by-Step Procedure

To configure R2:

#### 1. Configure the device interfaces.

```
[edit]
user@R2# set interfaces ge-2/0/7 description to-Host
user@R2# set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
user@R2# set interfaces ge-2/0/8 description to-R1
user@R2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@R2# set interfaces lo0 unit 0 description loopback-interface
user@R2# set interfaces lo0 unit 0 family inet address 192.168.14.1/32
```

Configure OSPF.

```
[edit protocols ospf]
user@R2# set area 0.0.0.0 interface ge-2/0/7.0 passive
user@R2# set area 0.0.0.0 interface lo0.0 passive
user@R2# set area 0.0.0.0 interface ge-2/0/8.0
```

### Step-by-Step Procedure

To configure R3:

#### 1. Configure the interfaces.

```
[edit]
user@R3# set interfaces ge-0/0/0 description to-Host
user@R3# set interfaces ge-0/0/0 unit 0 family inet address 172.16.71.1/30
user@R3# set interfaces ge-0/0/1 description to-R1
```



```

user@R3# set interfaces ge-0/0/1 unit 0 family inet address 10.51.0.2/30
user@R3# set interfaces lo0 unit 0 description loopback-interface
user@R3# set interfaces lo0 unit 0 family inet address 192.168.15.1/32

```

## 2. Configure OSPF

```

[edit protocols ospf]
user@R3# set protocols ospf area 0.0.0.0 interface ge-0/0/0.0 passive
user@R3# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@R3# set protocols ospf area 0.0.0.0 interface ge-0/0/1.0

```

## Results

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

```

user@R1# show interfaces
ge-0/0/1 {
  description to-R3;
  unit 0 {
    family inet {
      address 10.51.0.1/30;
    }
  }
}
}
}
ge-2/0/5 {
  description to-Host;
  unit 0 {
    family inet {
      address 172.16.70.2/30;
    }
  }
}
ge-2/0/8 {
  description to-R2;
  unit 0 {
    family inet {
      filter {

```



```

        output mf-classifier;
    }
    address 10.50.0.1/30;
}
}
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
}

```

```

user@R1# show class-of-service
forwarding-classes {
    class BE-data queue-num 0;
    class Premium-data queue-num 1;
    class Voice queue-num 2;
    class NC queue-num 3;
}

```

```

user@R1# show firewall
family inet {
    filter mf-classifier {
        term BE-data {
            from {
                protocol tcp;
                port http;
            }
            then {
                policer discard;
                forwarding-class BE-data;
            }
        }
        term Premium-data {
            from {
                protocol tcp;
                port 12345;
            }
        }
    }
}

```







```

}
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.50.0.2/30;
        }
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.14.1/32;
        }
    }
}
}

```

```

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

```

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

```

user@R3# show interfaces
ge-0/0/0 {
    description to-Host;
    unit 0 {
        family inet {
            address 172.16.71.2/30;
        }
    }
}
ge-0/0/1 {

```



```

description to-R1;
unit 0 {
    family inet {
        address 10.51.0.2/30;
    }
}
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.15.1/32;
        }
    }
}
}

```

```

user@R3# show protocols ospf
area 0.0.0.0 {
    interface ge-0/0/0.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-0/0/1.0;
}

```

If you are done configuring R3, enter `commit` from configuration mode.

## Verification

### IN THIS SECTION

- [Checking the CoS Settings | 225](#)
- [Clearing the Counters | 225](#)
- [Sending Traffic into the Network from TCP HTTP Port 80 and Monitoring the Results | 226](#)
- [Sending Traffic into the Network from TCP Port 12345 and Monitoring the Results | 228](#)



Confirm that the configuration is working properly.

Checking the CoS Settings

Purpose

Confirm that the forwarding classes are configured correctly.

Action

From R1, run the `show class-of-service forwarding-class` command.

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

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

Meaning

The output shows the configured custom classifier settings.

Clearing the Counters

Purpose

Confirm that the firewall and interface counters are cleared.



## Action

- On R1, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@R1> clear firewall all
```

- On R1, run the `clear interface statistics ge-2/0/5` command to reset the interface counters to 0.

```
user@R1> clear interface statistics ge-2/0/8
```

## Sending Traffic into the Network from TCP HTTP Port 80 and Monitoring the Results

### Purpose

Send traffic that can be monitored at the policer and custom queue level.

### Action

- Use a traffic generator to send 20 TCP packets with a source port of 80 into the network.

The `-s` flag sets the source port. The `-k` flag causes the source port to remain steady at 80 instead of incrementing. The `-c` flag sets the number of packets to 20. The `-d` flag sets the packet size.



**NOTE:** In this example the policer numbers are reduced to a bandwidth limit of 8 Kbps and a burst size limit of 1500 KBps to ensure that some packets are dropped.

```
[User@host]# hping 172.16.80.1 -c 20 -s 80 -k -d 300
```

```
[User@Host]# hping 172.16.80.1 -s 80 -k -c 20 -d 375
```

```
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 375 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=1001.0 ms
```

```
.
```

```
.
```

```
.
```

```
--- 172.16.80.1 hping statistic ---
```

```
20 packets transmitted, 14 packets received, 30% packet loss
```

```
round-trip min/avg/max = 1001.0/10287.1/19002.1 ms
```



2. On R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall
```

```
Filter: mf-classifier
```

```
Policers:
```

Name	Bytes	Packets
discard-BE-data	2490	6
discard-Premium-data	0	0

Notice that in the `hping` output that there was 30% packet loss (6 packets out of 20) and the same number of packets was dropped by the policer as shown in the output of the `show firewall` command. Also notice that the drops are associated with the queue BE-data as specified in the `mf-classifier` in the firewall configuration.

3. On R1, check the queue counters by using the `show interfaces extensive ge-2/0/8 | find "Queue counters"` command.

```
user@R1> show interfaces extensive ge-2/0/8 | find "Queue counters"
```

Queue counters:	Queued packets	Transmitted packets	Dropped packets
0	14	14	0
1	0	0	0
2	0	0	0
3	16	16	0

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

Notice that 14 packets were transmitted out interface 2/0/8 using the queue BE-data as specified in the `mf-classifier` in the firewall configuration. The remaining 6 packets were dropped by the policer, as shown above. The 16 packets sent to queue 3 are network control traffic. They are possibly routing protocol updates.

## Meaning

The output from both devices shows that 6 packets were discarded. This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 Kbps burst option for red (out-of-



contract HTTP port 80) traffic was exceeded. In Steps 2 and 3, you can see that the correct queues were used to transmit the remaining traffic out interface 2/0/8.

## Sending Traffic into the Network from TCP Port 12345 and Monitoring the Results

### Purpose

Send traffic that can be monitored at the policer and custom queue level.

### Action

1. Clear the counters again as shown in section ["Clearing the Counters" on page 225](#).
2. Use a traffic generator to send 20 TCP packets with a source port of 12345 into the network.

The `-s` flag sets the source port. The `-k` flag causes the source port to remain steady at 12345 instead of incrementing. The `-c` flag sets the number of packets to 20. The `-d` flag sets the packet size.

```
[User@host]# hping 172.16.80.1 -c 20 -s 12345 -k -d 300
[Host@User]# hping 172.16.80.1 -s 12345 -k -c 20 -d 375
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 375 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=1000.4 ms
.
.
.

--- 172.16.80.1 hping statistic ---
20 packets transmitted, 13 packets received, 35% packet loss
round-trip min/avg/max = 1000.4/10924.5/19002.2 ms
```

3. On R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall
Filter: mf-classifier
Policers:

```

Name	Bytes	Packets
discard-BE-data	0	0
discard-Premium-data	2905	7

Notice that in the `hping` output that there was 35% packet loss (7 packets out of 20) and the same number of packets were dropped by the policer as shown in the output of the `show firewall` command.



Also notice that the drops are associated with the queue Premium-data as specified in the mf-classifier in the firewall configuration.

4. On R1, check the queue counters by using the `show interfaces extensive ge-2/0/8| find "Queue counters"` command.

```
user@R1> show interfaces extensive ge-2/0/8| find "Queue counters"
```

Queue counters:	Queued packets	Transmitted packets	Dropped packets
0	0	0	0
1	13	13	0
2	0	0	0
3	16	16	0

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

Notice that 13 packets were transmitted out interface 2/0/8 using the Premium-data queues specified in the mf-classifier in the firewall configuration. The remaining 7 packets were dropped by the policer, as shown above. The 16 packets sent to queue 3 are network control traffic. They are possibly routing protocol updates.

## Meaning

The output from both devices shows that 7 packets were discarded. This means that there was at least 8 Kbps of green (in-contract HTTP port 80) traffic and that the 1500 Kbps burst option for red (out-of-contract HTTP port 80) traffic was exceeded. In Steps 3 and 4, you can see that the correct queues were used to transmit the remaining traffic out interface 2/0/8.

## RELATED DOCUMENTATION

[Routing Policies, Firewall Filters, and Traffic Policers User Guide](#)

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

## Overview of Tricolor Marking Architecture

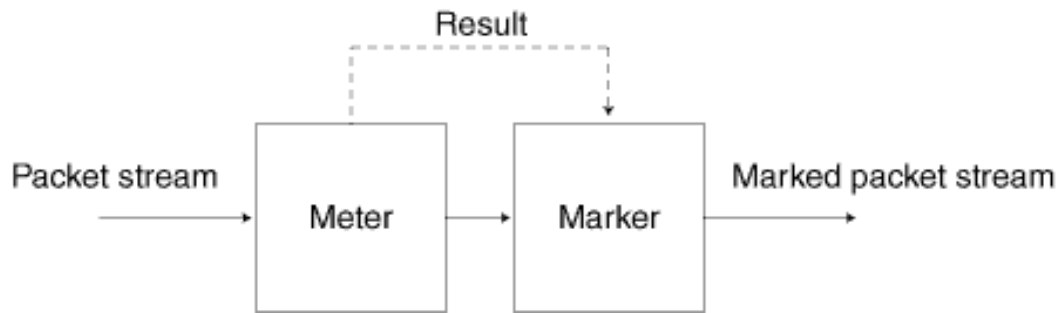
Policers provide two functions: metering and marking.



The policer meters each packet and passes the packet and the metering result to the marker, as shown in [Figure 28 on page 231](#).



Figure 28: Flow of Tricolor Marking Policer Operation



g017049



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 245](#).

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 63](#) and ["Using Multifield Classifiers to Set Packet Loss Priority" on page 137](#).

## RELATED DOCUMENTATION

[Enabling Tricolor Marking and Limitations of Three-Color Policers | 232](#)

[Configuring and Applying Tricolor Marking Policers | 234](#)

## 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 233](#) 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 | 229](#)

[Configuring and Applying Tricolor Marking Policers | 234](#)

## Configuring and Applying Tricolor Marking Policers

### IN THIS SECTION

- [Defining a Tricolor Marking Policer | 235](#)
- [Applying Tricolor Marking Policers to Firewall Filters | 238](#)
- [Applying Firewall Filter Tricolor Marking Policers to Interfaces | 239](#)
- [Example: Configuring and Applying a Single-Rate Tricolor Marking Policer | 239](#)

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

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 232](#)):

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 17 on page 235](#) describes all the configurable TCM statements.

**Table 17: 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.	–
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



Table 17: Tricolor Marking Policer Statements (*Continued*)

Statement	Meaning	Configurable Values
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 [Junos OS 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*

[Overview of Tricolor Marking Architecture | 229](#)

## Configuring Single-Rate Tricolor Marking

### IN THIS SECTION

- [Configuring Color-Blind Mode for Single-Rate Tricolor Marking | 242](#)
- [Configuring Color-Aware Mode for Single-Rate Tricolor Marking | 243](#)



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

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 18 on page 242](#).

**Table 18: 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 19 on page 243](#).

**Table 19: Color-Aware Mode TCM PLP Mapping**

Incoming PLP	Packet Metered Against	Possible Cases	Outgoing PLP
low	CBS and EBS	Packet does not exceed the CBS.	low
		Packet exceeds the CBS but not the EBS.	medium-high
		Packet exceeds the EBS.	high
medium-low	EBS only	Packet does not exceed the CBS.	medium-low
		Packet does not exceed the EBS.	medium-low
		Packet exceeds the EBS.	high
medium-high	EBS only	Packet does not exceed the CBS.	medium-high
		Packet does not exceed the EBS.	medium-high
		Packet exceeds the EBS.	high
high	Not metered by the policer.	All cases.	high

The following sections describe single-rate color-aware PLP mapping in more detail.



### Effect on Low PLP of Single-Rate Policer

Packets belonging to the green class have already been marked by a classifier with low PLP. The marking policer can leave the packet's PLP unchanged or increase the PLP to medium-high or high. Therefore, these packets are metered against both the CBS and the EBS.

For example, if a BA or multifield classifier marks a packet with low PLP 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 | 234](#)

[Configuring Two-Rate Tricolor Marking | 245](#)

## Configuring Two-Rate Tricolor Marking

### IN THIS SECTION

- [Configuring Color-Blind Mode for Two-Rate Tricolor Marking | 245](#)
- [Configuring Color-Aware Mode for Two-Rate Tricolor Marking | 246](#)

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

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 20 on page 246](#).

**Table 20: 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 21 on page 247](#).



**Table 21: 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 | 234](#)

[Configuring Single-Rate Tricolor Marking | 241](#)

## Example: Configuring and Verifying Two-Rate Tricolor Marking

### IN THIS SECTION

- [Requirements | 249](#)
- [Overview | 249](#)
- [Configuration | 250](#)
- [Verification | 259](#)

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

### Requirements

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

### Overview

#### IN THIS SECTION

- [Topology | 250](#)



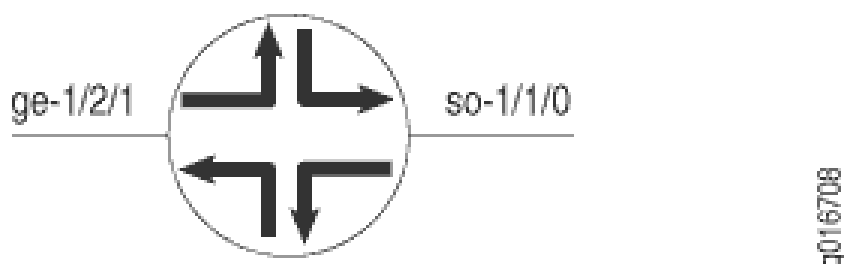
This example configures a two-rate tricolor marking policer on an input Gigabit Ethernet interface and shows commands to verify its operation.

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 29 on page 250 shows the sample topology.

Figure 29: Tricolor Marking Sample Topology



## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 250](#)
- [Example: Applying a Policer to an Input Interface | 252](#)
- [Example: Applying Profiles to an Output Interface | 254](#)
- [Example: Marking Packets with Medium-Low Loss Priority | 256](#)
- [Results | 258](#)

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

### CLI Quick Configuration

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



## Applying a Policer to an Input Interface

```
set interfaces 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 [Junos OS 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 [Junos OS 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

### IN THIS SECTION

- [Verifying Two-Rate Tricolor Marking Operation | 259](#)

Confirm that the configuration is working properly.

### Verifying Two-Rate Tricolor Marking Operation

#### Purpose

#### 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 | 245](#)

*Guidelines for Configuring Firewall Filters*

## Applying Firewall Filter Tricolor Marking Policers to Interfaces

### IN THIS SECTION

- [Example: Applying a Single-Rate Tricolor Marking Policers to an Interface | 261](#)

To apply a tricolor marking policer to an interface, you must reference the filter name in the interface configuration. To do this, include the `filter` statement:

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

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*]

The filter name that you reference should have an attached tricolor marking policer, as shown in ["Configuring and Applying Tricolor Marking Policers" on page 234](#).



## Example: Applying a Single-Rate Tricolor Marking Policer to an Interface

Apply the `trtcm1-cb` policer to an interface:

```

firewall {
    three-color-policer srtcm1 {    # Configure the srtcm1-cb policer.
        single-rate {
            color-blind;
            committed-information-rate 1048576;
            committed-burst-size 65536;
            excess-burst-size 131072;
        }
    }
}
filter fil {    # Configure the fil firewall filter, applying the srtcm1-cb policer.
    term default {
        then {
            three-color-policer {
                single-rate srtcm1-cb;    # The TCM policer must be single-rate.
            }
        }
    }
}
interfaces {    # Configure the interface, which attaches the fil firewall filter.
    ge-1/0/0 {
        unit 0 {
            family inet {
                filter {
                    input fil;
                }
            }
        }
    }
}

```

### RELATED DOCUMENTATION

| [Configuring and Applying Tricolor Marking Policers](#) | 234



## Policer Overhead to Account for Rate Shaping in the Traffic Manager

### IN THIS SECTION

- [Policer Overhead to Account for Rate Shaping Overview | 262](#)
- [Example: Configuring Policer Overhead to Account for Rate Shaping | 262](#)

### Policer Overhead to Account for Rate Shaping Overview

If you configure ingress or egress traffic-shaping overhead values for an interface, the traffic manager cannot apply these values to any rate-limiting also applied to the interface. To enable the router to account for the additional Ethernet frame length when policing actions are being determined, you must configure the ingress or egress overhead values for policers separately.



**NOTE:** When a policer overhead value is changed, the PIC or DPC goes offline and then comes back online.

For Gigabit Ethernet Intelligent Queuing 2 (IQ2) and Enhanced IQ2 (IQ2E) PICs or interfaces on Dense Port Concentrators (DPCs) in MX Series routers, you can control the rate of traffic that passes through all interfaces on the PIC or DPC by configuring a *policer overhead*. You can configure a policer ingress overhead and a policer egress overhead, each with values from 0 through 255 bytes. The policer overhead values are added to the length of the final Ethernet frame when determining ingress and egress policer actions.

### SEE ALSO

[\*egress-policer-overhead\*](#)

[\*ingress-policer-overhead\*](#)

### Example: Configuring Policer Overhead to Account for Rate Shaping

#### IN THIS SECTION

- [Requirements | 263](#)
- [Overview | 263](#)



●	<a href="#">Configuration   264</a>
●	<a href="#">Verification   272</a>

This example shows how to configure overhead values for policers when rate-shaping overhead is configured.

## Requirements

Before you begin, make sure that interface for which you are applying ingress or egress policer overhead is hosted on one of the following:

- Gigabit Ethernet IQ2 PIC
- IQ2E PIC
- DPCs in MX Series routers

## Overview

### IN THIS SECTION

●	<a href="#">Topology   263</a>
---	--------------------------------

This example shows how to configure policer overhead values for all physical interfaces on a supported PIC or MPC so that the rate shaping value configured on a logical interface is accounted for in any policing on that logical interface.

## *Topology*

The router hosts a Gigabit Ethernet IQ2 PIC, installed in PIC location 3 of the Flexible PIC Concentrator (FPC) in slot number 1. The physical interface on port 1 on that PIC is configured to receive traffic on logical interface 0 and send it back out on logical interface 1. Class-of-service scheduling includes 100 Mbps of traffic rate-shaping overhead for the output traffic. A policer egress overhead of 100 bytes is configured on the entire PIC so that, for any policers applied to the output traffic, 100 bytes are added to the final Ethernet frame length when determining ingress and egress policer actions.





**NOTE:** Traffic rate-shaping and corresponding policer overhead are configured separately:

- You configure rate shaping at the [edit class-of-service interfaces *interface-name* unit *unit-number*] hierarchy level.
- You configure policer overhead at the [edit chassis fpc *slot-number* pic *pic-number*] hierarchy level.

When a policer overhead value is changed, the PIC or DPC goes offline and then comes back online.

## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 264](#)
- [Configuring the Logical Interfaces | 265](#)
- [Configuring Traffic Rate-Shaping on the Logical Interface That Carries Output Traffic | 267](#)
- [Configuring Policer Overhead on the PIC or DPC That Hosts the Rate-Shaped Logical Interface | 269](#)
- [Applying a Policer to the Logical Interface That Carries Input Traffic | 270](#)

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

To configure this example, perform the following tasks:

### *CLI Quick Configuration*

To quickly configure this example, copy the following configuration commands into a text file, remove any line breaks, and then paste the commands into the CLI at the [edit] hierarchy level.

```
set interfaces ge-1/3/1 per-unit-scheduler
set interfaces ge-1/3/1 vlan-tagging
set interfaces ge-1/3/1 unit 0 vlan-id 100
set interfaces ge-1/3/1 unit 0 family inet address 10.10.10.1/30
set interfaces ge-1/3/1 unit 1 vlan-id 101
set interfaces ge-1/3/1 unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac
```



```

00:00:11:22:33:44
set class-of-service schedulers be transmit-rate percent 5
set class-of-service schedulers ef transmit-rate percent 30
set class-of-service schedulers af transmit-rate percent 30
set class-of-service schedulers nc transmit-rate percent 35
set class-of-service scheduler-maps my-map forwarding-class best-effort scheduler be
set class-of-service scheduler-maps my-map forwarding-class expedited-forwarding scheduler ef
set class-of-service scheduler-maps my-map forwarding-class network-control scheduler nc
set class-of-service scheduler-maps my-map forwarding-class assured-forwarding scheduler af
set class-of-service interfaces ge-1/3/1 unit 1 scheduler-map my-map
set class-of-service interfaces ge-1/3/1 unit 1 shaping-rate 100m
set firewall policer 500Kbps logical-interface-policer
set firewall policer 500Kbps if-exceeding bandwidth-limit 500k
set firewall policer 500Kbps if-exceeding burst-size-limit 625k
set firewall policer 500Kbps then discard
set chassis fpc 1 pic 3 ingress-policer-overhead 100
set chassis fpc 1 pic 3 egress-policer-overhead 100
set interfaces ge-1/3/1 unit 0 family inet policer input 500Kbps

```

### *Configuring the Logical Interfaces*

#### **Step-by-Step Procedure**

To configure the logical interfaces:

1. Enable configuration of the interface

```

[edit]
user@host# edit interfaces ge-1/3/1

```

2. Enable multiple queues for each logical interface (so that you can associate an output scheduler with each logical interface).

```

[edit interfaces ge-1/3/1]
user@host# set per-unit scheduler
user@host# set vlan-tagging

```





**NOTE:** For Gigabit Ethernet IQ2 PICs only, use the `shared-scheduler` statement to enable shared schedulers and shapers on a physical interface.

### 3. Configure logical interface `ge-1/3/1.0`.

```
[edit interfaces ge-1/3/1]
user@host# set unit 0 vlan-id 100
user@host# set unit 0 family inet address 10.10.10.1/30
```

### 4. Configure logical interface `ge-1/3/1.1`.

```
[edit interfaces ge-1/3/1]
user@host# set unit 1 vlan-id 101
user@host# set unit 1 family inet address 20.20.20.1/30 arp 20.20.20.2 mac 00:00:11:22:33:44
```

## Results

Confirm the configuration of the interfaces by entering the `show interfaces configuration mode` command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show interfaces
ge-1/3/1 {
    per-unit-scheduler;
    vlan-tagging;
    unit 0 {
        vlan-id 100;
        family inet {
            address 10.10.10.1/30;
        }
    }
    unit 1 {
        vlan-id 101;
        family inet {
            address 20.20.20.1/30 {
                arp 20.20.20.2 mac 00:00:11:22:33:44;
            }
        }
    }
}
```



```

    }
  }
}

```

### *Configuring Traffic Rate-Shaping on the Logical Interface That Carries Output Traffic*

#### Step-by-Step Procedure

To configure traffic rate-shaping on the logical interface that carries output traffic:

1. Enable configuration of class-of-service features.

```

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

```

2. Configure packet scheduling on logical interface ge-1/3/1.0.

- Configure schedulers that specify the percentage of transmission capacity.

```

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

[edit class-of-service schedulers]
user@host# set be transmit-rate percent 5
user@host# set ef transmit-rate percent 30
user@host# set af transmit-rate percent 30
user@host# set nc transmit-rate percent 35

```

A percentage of zero drops all packets in the queue. When the `rate-limit` option is specified, the transmission rate is limited to the rate-controlled amount. In contrast with the `exact` option, a scheduler with the `rate-limit` option shares unused bandwidth above the rate-controlled amount.

- Configure a scheduler map to associate each scheduler with a forwarding class.

```

[edit class-of-service]
user@host# edit scheduler-maps my-map

[edit class-of-service scheduler-maps my-map]
user@host# set forwarding-class best-effort scheduler be
user@host# set forwarding-class expedited-forwarding scheduler ef

```



```

user@host# set forwarding-class network-control scheduler nc
user@host# set forwarding-class assured-forwarding scheduler af

```

- Associate the scheduler map with logical interface ge-1/3/1.0.

```

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

[edit class-of-service interfaces ge-1/3/1 unit 1]
user@host# set scheduler-map my-map

```

### 3. Configure 100 Mbps of traffic rate-shaping overhead on logical interface ge-1/3/1.1.

```

[edit class-of-service interfaces ge-1/3/1 unit 1]
user@host# set shaping-rate 100

```

Alternatively, you can configure a shaping rate for a logical interface and oversubscribe the physical interface by including the `shaping-rate` statement at the `[edit class-of-service traffic-control-profiles]` hierarchy level. With this configuration approach, you can independently control the delay-buffer rate.

## Results

Confirm the configuration of the class-of-service features (including the 100 Mbp of shaping of the egress traffic) by entering the `show class-of-service` configuration mode command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```

[edit]
user@host# show class-of-service
interfaces {
  ge-1/3/1 {
    unit 1 {
      scheduler-map my-map;
      shaping-rate 100m;
    }
  }
}
scheduler-maps {
  my-map {

```



```

        forwarding-class best-effort scheduler be;
        forwarding-class expedited-forwarding scheduler ef;
        forwarding-class network-control scheduler nc;
        forwarding-class assured-forwarding scheduler af;
    }
}
schedulers {
    be {
        transmit-rate percent 5;
    }
    ef {
        transmit-rate percent 30;
    }
    af {
        transmit-rate percent 30;
    }
    nc {
        transmit-rate percent 35;
    }
}

```

### *Configuring Policer Overhead on the PIC or DPC That Hosts the Rate-Shaped Logical Interface*

#### **Step-by-Step Procedure**

To configure policer overhead on the PIC or MPC that hosts the rate-shaped logical interface:

1. Enable configuration of the supported PIC or MPC.

```

[edit]
user@host# set chassis fpc 1 pic 3

```

2. Configure 100 bytes of policer overhead on the supported PIC or MPC.

```

[edit chassis fpc 1 pic 3]
user@host# set ingress-policer-overhead 100
user@host# set egress-policer-overhead 100

```





**NOTE:** These values are added to the length of the final Ethernet frame when determining ingress and egress policer actions for all physical interfaces on the PIC or MPC.

You can specify policer overhead with values from 0 through 255 bytes.

## Results

Confirm the configuration of the policer overhead on the physical interface to account for rate-shaping by entering the `show chassis configuration mode` command. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show chassis
chassis {
  fpc 1 {
    pic 3 {
      egress-policer-overhead 100;
      ingress-policer-overhead 100;
    }
  }
}
```

## *Applying a Policer to the Logical Interface That Carries Input Traffic*

### Step-by-Step Procedure

To apply a policer to the logical interface that carries input traffic:

1. Configure the logical interface (aggregate) policer.

```
[edit]
user@host# edit firewall policer 500Kbps

[edit firewall policer 500Kbps]
user@host# set logical-interface-policer
user@host# set if-exceeding bandwidth-limit 500k
user@host# set if-exceeding burst-size-limit 625k
user@host# set then discard
```



2. Apply the policer to Layer 3 input on the IPv4 logical interface.

```
[edit]
user@host# set interfaces ge-1/3/1 unit 0 family inet policer input 500Kbps
```



**NOTE:** The 100 Mbps policer overhead is added to the length of the final Ethernet frame when determining ingress and egress policer actions,

## Results

Confirm the configuration of the policer with rate-shaping overhead by entering the `show firewall` and `show interfaces configuration mode` commands. If the command output does not display the intended configuration, repeat the instructions in this procedure to correct the configuration.

```
[edit]
user@host# show firewall
policer 500Kbps {
    logical-interface-policer;
    if-exceeding {
        bandwidth-limit 500k;
        burst-size-limit 625k;
    }
    then discard;
}

[edit]
user@host# show interfaces
ge-1/3/1 {
    per-unit-scheduler;
    vlan-tagging;
    unit 0 {
        vlan-id 100;
        layer2-policer {
            input-policer 500Kbps;
        }
        family inet {
            address 10.10.10.1/30;
        }
    }
}
```



```

unit 0 {
    vlan-id 101;
    family inet {
        address 20.20.20.1/30 {
            arp 20.20.20.2 mac 00:00:11:22:33:44;
        }
    }
}

```

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

## Verification

### IN THIS SECTION

- [Displaying Traffic Statistics and Policers for the Logical Interface | 272](#)
- [Displaying Statistics for the Policer | 273](#)

Confirm that the configuration is working properly.

### *Displaying Traffic Statistics and Policers for the Logical Interface*

#### Purpose

Verify the traffic flow through the logical interface and that the policer is evaluated when packets are received on the logical interface.

#### Action

Use the `show interfaces operational mode` command for logical interface `ge-1/3/1.0`, and include the `detail` or `extensive` option. The command output section for **Traffic statistics** lists the number of bytes and packets received and transmitted on the logical interface, and the **Protocol inet** section contains a **Policer** field that would list the policer `500Kbps` as an input or output policer as follows:

- **Input:** `500Kbps-ge-1/3/1.0-log_int-i`
- **Output:** `500Kbps-ge-1/3/1.0-log_int-o`



The **log\_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log\_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to Input traffic only.

### *Displaying Statistics for the Policer*

#### **Purpose**

Verify the number of packets evaluated by the policer.

#### **Action**

Use the `show policer operational mode` command and optionally specify the name of the policer. The command output displays the number of packets evaluated by each configured policer (or the specified policer), in each direction. For the policer `500Kbps`, the input and output policer names are displayed as follows:

- `500Kbps-ge-1/3/1.0-log_int-i`
- `500Kbps-ge-1/3/1.0-log_int-o`

The **log\_int-i** suffix denotes a logical interface policer applied to input traffic, while the **log\_int-o** suffix denotes a logical interface policer applied to output traffic. In this example, the logical interface policer is applied to input traffic only.

#### **SEE ALSO**

*egress-policer-overhead*

*ingress-policer-overhead*

#### **RELATED DOCUMENTATION**

*Two-Color Policer Configuration Overview*

*Guidelines for Applying Traffic Policers*

*Configuring a Policer Overhead*

[CLI Explorer](#)



# Defining Forwarding Behavior with Forwarding Classes

## IN THIS CHAPTER

- Understanding How Forwarding Classes Assign Classes to Output Queues | 275
- Default Forwarding Classes | 277
- Configuring a Custom Forwarding Class for Each Queue | 281
- Configuring Up to 16 Custom Forwarding Classes | 283
- Classifying Packets by Egress Interface | 291
- Forwarding Policy Options Overview | 294
- Configuring CoS-Based Forwarding | 296
- Example: Configuring CoS-Based Forwarding | 299
- Example: Configuring CoS-Based Forwarding for Different Traffic Types | 303
- Example: Configuring CoS-Based Forwarding for IPv6 | 304
- Applying Forwarding Classes to Interfaces | 305
- Understanding Queuing and Marking of Host Outbound Traffic | 306
- Forwarding Classes and Fabric Priority Queues | 308
- Default Routing Engine Protocol Queue Assignments | 309
- Assigning Forwarding Class and DSCP Value for Routing Engine-Generated Traffic | 313
- Example: Writing Different DSCP and EXP Values in MPLS-Tagged IP Packets | 315
- Change the Default Queuing and Marking of Host Outbound Traffic | 318
- Example: Configure Different Queuing and Marking Defaults for Outbound Routing Engine and Distributed Protocol Handler Traffic | 320
- Overriding the Input Classification | 330



## Understanding How Forwarding Classes Assign Classes to Output Queues

### IN THIS SECTION

- [Output Queue Assignments Based on Forwarding Class | 275](#)
- [Devices That Support Up to Four Forwarding Classes | 275](#)
- [Devices That Support Up to 16 Forwarding Classes | 276](#)
- [Default and Configurable Packet Loss Priority Values | 276](#)
- [Configuration Statements Used to Configure and Apply Forwarding Classes | 277](#)

This topic covers the following information:

### Output Queue Assignments Based on Forwarding Class

It is helpful to think of forwarding classes as output queues. In effect, the end result of classification is the identification of an output queue for a particular packet.

CoS packet classification assigns an incoming packet to an output queue based on the packet's forwarding class. Each packet is associated with one of the following default forwarding classes:

- Expedited forwarding (EF)—Provides a low-loss, low-latency, low- *jitter*, assured bandwidth, end-to-end service.
- Assured forwarding (AF)—Provides a group of values you can define and includes four subclasses: AF1, AF2, AF3, and AF4, each with three drop probabilities: low, medium, and high.
- Best effort (BE)—Provides no service profile. For the best effort forwarding class, loss priority is typically not carried in a class-of-service (CoS) value and random early detection (RED) drop profiles are more aggressive.
- Network control (NC)—This class is typically high priority because it supports protocol control.

### Devices That Support Up to 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);
}
interfaces {
  interface-name {
    unit logical-unit-number {
      forwarding-class class-name;
    }
  }
}
restricted-queues {
  forwarding-class class-name queue queue-number;
}
```

### RELATED DOCUMENTATION

<a href="#">Default Forwarding Classes   277</a>
<a href="#">Configuring a Custom Forwarding Class for Each Queue   281</a>
<a href="#">Applying Forwarding Classes to Interfaces   305</a>
<a href="#">Configuring Up to 16 Custom Forwarding Classes   283</a>
<a href="#">Controlling Network Access Using Traffic Policing Overview</a>

## 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 278](#).



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.
Queue 1	expedited-forwarding (ef)	The software delivers assured bandwidth, low loss, low delay, and low delay variation ( <i>jitter</i> ) 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.
Queue 2	assured-forwarding (af)	The software offers a high level of assurance that the packets are delivered as long as the packet flow from the customer stays within a certain service profile that you define.  The software accepts excess traffic, but applies a RED drop profile to determine if the excess packets are dropped and not forwarded.  Depending on router type, up to four drop probabilities (low, medium-low, medium-high, and high) are defined for this service class.



**Table 22: Default Forwarding Classes** *(Continued)*

Queue	Forwarding Class Name	Comments
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 283](#).



**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 | 275](#)

[Configuring a Custom Forwarding Class for Each Queue | 281](#)

[Change the Default Queuing and Marking of Host Outbound Traffic | 318](#)

[CoS Features and Limitations on PTX Series Routers | 785](#)

## 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 283), include the class statement at the [edit class-of-service forwarding-classes] hierarchy level.

You cannot commit a configuration that assigns the same forwarding class to two different queues.



**CAUTION:** We do not recommend classifying packets into a forwarding class that has no associated scheduler on the egress interface. Such a configuration can cause unnecessary packet drops because an unconfigured scheduling class might lack adequate buffer space. For example, if you configure a custom scheduler map that does not define queue 0, and the default classifier assigns incoming packets to the best-effort class



(queue 0), the unconfigured egress queue for the best-effort forwarding class might not have enough space to accommodate even short packet bursts.

A default congestion and transmission control mechanism is used when an output interface is not configured for a certain forwarding class, but receives packets destined for that unconfigured forwarding class. This default mechanism uses the delay buffer and weighted round robin (WRR) credit allocated to the designated forwarding class, with a default drop profile. Because the buffer and WRR credit allocation is minimal, packets might be lost if a larger number of packets are forwarded without configuring the forwarding class for the interface.



**CAUTION:** When you define a forwarding class for the same queue as one of the default forwarding classes, the default forwarding class is automatically removed. For example, if you define class `be` for queue 0, which is the queue for the default best-effort forwarding class, the best-effort class is removed.

If you define more than one forwarding class for a given queue number and use the name of a default forwarding class for one of the new classes, the new class with the default name is deleted.

To create custom forwarding class names and assign each forwarding class to any queue number:

1. Access the CoS forwarding class configuration hierarchy.

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

2. Specify the forwarding class name and queue number.

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

## RELATED DOCUMENTATION

[Understanding How Forwarding Classes Assign Classes to Output Queues | 275](#)

[Default Forwarding Classes | 277](#)

[Change the Default Queuing and Marking of Host Outbound Traffic | 318](#)



## Configuring Up to 16 Custom Forwarding Classes

### IN THIS SECTION

- [Enabling Eight Queues on Interfaces | 286](#)
- [Assigning Multiple Forwarding Classes and Default Forwarding Classes | 288](#)
- [Examples: Configuring Up to 16 Forwarding Classes | 289](#)

By default on all routers, four forwarding classes are mapped to four output queues, as shown in the topic ["Default Forwarding Classes" on page 277](#). 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*.

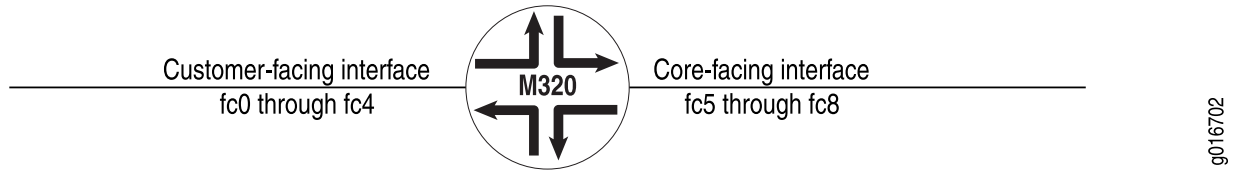


**NOTE:** You cannot use CoS-based forwarding features if you configure more than eight forwarding classes on the device.

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 30 on page 284](#).



Figure 30: Customer-Facing and Core-Facing Forwarding Classes





In this example, you need nine classifiers and, therefore, nine forwarding classes. The forwarding class-to-queue mapping is shown in [Table 23 on page 285](#).

**Table 23: Sample Forwarding Class-to-Queue Mapping**

Forwarding Class Names	Queue Number
fc0 fc5	0
fc1 fc6	1
fc2 fc7	2
fc3 fc8	3
fc4	4

To configure up to 16 forwarding classes, include the `class` and `queue-num` statements at the `[edit class-of-service forwarding-classes]` hierarchy level:

```
[edit class-of-service forwarding-classes]
class class-name queue-num queue-number;
```

You can configure up 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 305](#).

## 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 277](#).

## Examples: Configuring Up to 16 Forwarding Classes

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

Specify each forwarding class and queue you want mapped.

```
[edit]
user@host# edit class-of-service forwarding-classes
user@host# set class fc0 queue-num 0
user@host# set class fc1 queue-num 0
user@host# set class fc2 queue-num 1
user@host# set class fc3 queue-num 1
user@host# set class fc4 queue-num 2
user@host# set class fc5 queue-num 2
user@host# set class fc6 queue-num 3
user@host# set class fc7 queue-num 3
user@host# set class fc8 queue-num 4
user@host# set class fc9 queue-num 4
user@host# set class fc10 queue-num 5
user@host# set class fc11 queue-num 5
user@host# set class fc12 queue-num 6
user@host# set class fc13 queue-num 6
user@host# set class fc14 queue-num 7
user@host# set class fc15 queue-num 7
```

For PICs restricted to four queues, map four forwarding classes to each queue:



1. Specify each forwarding class and queue you want mapped.

```
[edit]
user@host# edit class-of-service restricted-queues
user@host# set forwarding-class fc0 queue 0
user@host# set forwarding-class fc1 queue 0
user@host# set forwarding-class fc2 queue 0
user@host# set forwarding-class fc3 queue 0
user@host# set forwarding-class fc4 queue 1
user@host# set forwarding-class fc5 queue 1
user@host# set forwarding-class fc6 queue 1
user@host# set forwarding-class fc7 queue 1
user@host# set forwarding-class fc8 queue 2
user@host# set forwarding-class fc9 queue 2
user@host# set forwarding-class fc10 queue 2
user@host# set forwarding-class fc11 queue 2
user@host# set forwarding-class fc12 queue 3
user@host# set forwarding-class fc13 queue 3
user@host# set forwarding-class fc14 queue 3
user@host# set forwarding-class fc15 queue 3
```

If you map multiple forwarding classes to a queue, the multiple forwarding classes must refer to the same scheduler. To configure a scheduler map applicable to an interface restricted to four queues:

1. Specify a scheduler map name and associate it with the scheduler configuration and forwarding class.

```
[edit]
user@host# edit class-of-service scheduler-maps interface-restricted
user@host# set forwarding-class be scheduler 00
user@host# set forwarding-class ef scheduler Q1
user@host# set forwarding-class ef1 scheduler Q1
user@host# set forwarding-class ef2 scheduler Q1
user@host# set forwarding-class af1 scheduler Q2
user@host# set forwarding-class af scheduler Q2
user@host# set forwarding-class nc scheduler Q3
user@host# set forwarding-class nc1 scheduler Q3
```



2. Map the forwarding classes to the restricted queues.

```
[edit]
user@host# edit class-of-service restricted-queues
user@host# set forwarding-class be queue 0
user@host# set forwarding-class ef queue 1
user@host# set forwarding-class ef1 queue 1
user@host# set forwarding-class ef2 queue 1
user@host# set forwarding-class af queue 2
user@host# set forwarding-class af1 queue 2
user@host# set forwarding-class nc queue 3
user@host# set forwarding-class nc1 queue 3
```

## RELATED DOCUMENTATION

[Understanding How Forwarding Classes Assign Classes to Output Queues | 275](#)

[Default Forwarding Classes | 277](#)

## 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 281](#).

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.

Change History Table

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

Release	Description
17.2R1	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 | 296
- Example: Configuring CoS-Based Forwarding | 299



## 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 User 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 299](#).

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.

**Change History Table**

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

Release	Description
17.2R1	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**

<a href="#">Load Balancing VPLS Non-Unicast Traffic Across Member Links of an Aggregate Interface</a>
<a href="#">Forwarding Policy Options Overview</a>   <a href="#">294</a>

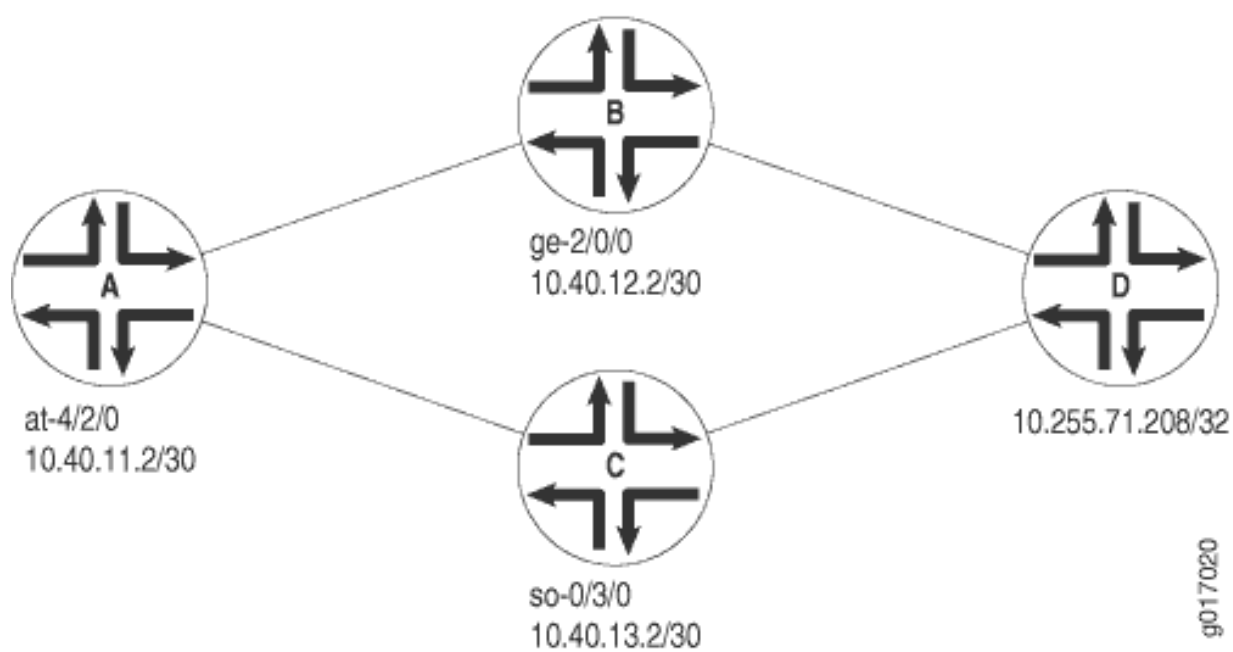
**Example: Configuring CoS-Based Forwarding**

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 31 on page 300](#).

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 31: 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](#) | 294

## 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

### IN THIS SECTION

- [Host Outbound Traffic Overview | 306](#)
- [Default Queuing and Marking of Host Outbound Traffic | 307](#)
- [Configured Queuing and Marking of Host Outbound Traffic | 307](#)
- [Configured Queuing and Marking of Outbound Routing Engine Traffic Only | 308](#)

This topic covers the following information:

### Host Outbound Traffic Overview

Host outbound traffic, also called locally generated traffic, consists of traffic generated by the Routing Engine and traffic generated by the distributed protocol handler.



## Routing Engine Sourced Traffic

Traffic sent from the Routing Engine includes control plane packets such as OSPF Hello packets, ICMP echo reply (ping) packets, and TCP-related packets such as BGP and LDP control packets.

## Distributed Protocol Handler Traffic

*Distributed protocol handler traffic* refers to traffic from the router's *periodic packet management* (PPM) process when it runs sessions distributed to the Packet Forwarding Engine (the default mode) in addition to the Routing Engine. The PPM process is responsible for periodic transmission of protocol Hello or other keepalive packets on behalf of its various client processes, such as Bidirectional Forwarding Detection (BFD) Protocol or Link Aggregation Control Protocol (LACP), and it also receives packets on behalf of client processes. In addition, PPM handles time-sensitive periodic processing and performs such tasks as sending process-specific packets and gathering statistics. By default, PPM sessions on the Routing Engine run distributed on the Packet Forwarding Engine, and this enables client processes to run on the Packet Forwarding Engine.



**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 309](#).

For most protocols, the router marks the type of service (ToS) field of Layer 3 packets in the host outbound traffic flow with DiffServ code point (DSCP) bits 000000 (which correlate with the best-effort forwarding class). For some protocols, such as BGP, the router marks the ToS field with 802.1p bits 110 (which correlate with the network-control forwarding class), while still assigning the packets to queue 0. The router does not remark Layer 2 traffic such as LACP control traffic on aggregated Ethernet. For more information, see ["Default DSCP and DSCP IPv6 Classifiers" on page 49](#).

## 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 | 309](#)

[Default DSCP and DSCP IPv6 Classifiers | 49](#)

[Understanding Queuing and Marking of Host Outbound Traffic | 306](#)

## Forwarding Classes and Fabric Priority Queues

### IN THIS SECTION

● [Default Fabric Priority Queuing | 309](#)

● [Overriding Default Fabric Priority Queuing | 309](#)

This topic covers the following information:



## 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](#) | 434

## Default Routing Engine Protocol Queue Assignments

[Table 24 on page 310](#) lists the default output queues to which Routing Engine sourced traffic is mapped by protocol type. In general, control protocol packets are sent over queue 3 and management traffic is sent over queue 0. The following caveats apply to these default queue assignments:

- For all packets sent to queue 3 over a VLAN-tagged interface, the software sets the 802.1p bit to 110, except for VRRP packets, in which case the software sets the 802.1p bit to 111.
- Outgoing BFD packets should be marked with VLAN-tagged 802.1p bit to 110; however, this is true only for RE based BFD. For inline BFD, it does not modify by default.
- Although BGP and LDP TCP traffic is sent to queue 0, it is marked with 802.1p bits 110 normally used for network control.
- For IPv4 and IPv6 packets, the software copies the IP type-of-service (ToS) value into the 802.1p field independently of which queue the packets are sent out.
- For MPLS packets, the software copies the EXP bits into the 802.1p field.



**Table 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
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



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

Routing Engine Protocol	Default Queue Assignment
Internet Control Message Protocol (ICMP)	Queue 0
Internet Group Management Protocol (IGMP) query	Queue 3
IGMP Report	Queue 0
Internet Key Exchange (IKE)	Queue 3
IP version 6 (IPv6) Neighbor Solicitation	Queue 3
IPv6 Neighbor Advertisement	Queue 3
IPv6 Router Advertisement	Queue 0
Label Distribution Protocol (LDP) User Datagram Protocol (UDP) hello	Queue 3
LDP keepalive and Session data	Queue 0
LDP TCP Retransmission	Queue 3
Link Aggregation Control Protocol (LACP)	Queue 3
Link Services (LS) PIC	If link fragmentation and interleaving (LFI) is enabled, all routing protocol packets larger than 128 bytes are transmitted from queue 0. This ensures that VoIP traffic is not affected. Fragmentation is supported on queue 0 only.
Multicast listener discovery (MLD)	Queue 0
Multicast Source Discovery Protocol (MSDP)	Queue 0



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

Routing Engine Protocol	Default Queue Assignment
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
Precision Time Protocol (PTP)	Queue 3
Real-time performance monitoring (RPM) probe packets	Queue 3
RSVP	Queue 3
Routing Information Protocol (RIP)	Queue 3
SNMP	Queue 0
SSH	Queue 0
sFlow monitoring technology	Queue 0
Telnet	Queue 0



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

Routing Engine Protocol	Default Queue Assignment
Two-Way Active Monitoring Protocol (TWAMP)	Queue 0
Virtual Router Redundancy Protocol (VRRP)	Queue 3
xnm-clear-text	Queue 0
xnm-ssl	Queue 0

## Assigning Forwarding Class and DSCP Value for Routing Engine-Generated Traffic

You can set the forwarding class and differentiated service code point (DSCP) value for traffic originating in the Routing Engine. To configure forwarding class and DSCP values that apply to Routing Engine-generated traffic only, apply an output filter to the loopback (lo.0) interface and set the appropriate forwarding class and DSCP bit configuration for various protocols. For example, you can set the DSCP value on OSPF packets that originate in the Routing Engine to 10 and assign them to the AF (assured forwarding) forwarding class while the DSCP value on ping packets are set to 0 and use forwarding class BE (best effort).

This particular classification ability applies to packets generated by the Routing Engine only.

The following example assigns Routing Engine sourced ping packets (using ICMP) a DSCP value of 38 and a forwarding class of af17, OSPF packets a DSCP value of 12 and a forwarding class of af11, and BGP packets (using TCP) a DSCP value of 10 and a forwarding class of af16.

```
[edit class-of-service]
forwarding-classes {
    class af11 queue-num 7;
    class af12 queue-num 1;
    class af13 queue-num 2;
    class af14 queue-num 4;
    class af15 queue-num 5;
    class af16 queue-num 4;
```



```

class af17 queue-num 6;
class af18 queue-num 7;
}

[edit firewall filter family inet]
filter loopback-filter {
    term t1 {
        from {
            protocol icmp; # For pings
        }
        then {
            forwarding-class af17;
            dscp 38;
        }
    }
    term t2 {
        from {
            protocol ospf; # For OSPF
        }
        then {
            forwarding-class af11;
            dscp 12;
        }
    }
    term t3 {
        from {
            protocol tcp; # For BGP
        }
        then {
            forwarding-class af16;
            dscp 10;
        }
    }
    term t4 {
        then accept; # Do not forget!
    }
}

[edit interfaces]
lo0 {
    unit 0 {
        family inet {
            filter {

```



```

        output loopback-filter;
    }
}
}
}

```



**NOTE:** This is not a complete router configuration. You still have to assign resources to the queues, configure the routing protocols, addresses, and so on.

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

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.



## Change the Default Queuing and Marking of Host Outbound Traffic

### IN THIS SECTION

- [Host Outbound Traffic Classification Overview | 318](#)
- [Platform-Specific Host Outbound Traffic Classification Behavior | 319](#)

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

### Host Outbound Traffic Classification Overview

TCP-related packets, such as BGP or LDP, use queue 3 (network control) for retransmitted traffic. Changing the defaults for Routing Engine sourced traffic does not affect transit or incoming traffic. The changes apply to all packets relating to Layer 3 and Layer 2 protocols, but not MPLS EXP bits or IEEE 802.1p bits. This feature applies to all application-level traffic such as FTP or ping operations as well.

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

The forwarding class selected must be properly configured on all interfaces.

To change the default forwarding class and DSCP bits for Routing Engine sourced traffic, include the `host-outbound-traffic` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
host-outbound-traffic {
    forwarding-class class-name;
    dscp-code-point value;
}
```



The following example places all Routing Engine sourced traffic into the network control forwarding class (assigned to queue 3 by default) with a DSCP value of 101010:

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

You can also modify the queue assignment of the forwarding class. For example:

```
[edit class-of-service]
forwarding-classes {
  class network-control queue-num 5;
}
```

Platform-Specific Host Outbound Traffic Classification Behavior

Use [Feature Explorer](#) to confirm platform and release support for [host outbound traffic classification](#).

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

Platform	Difference
ACX7000 Series Routers	ACX7000 Series Routers support only the forwarding-class <i>class-name</i> option.

RELATED DOCUMENTATION

<a href="#">Understanding How Forwarding Classes Assign Classes to Output Queues</a>	<a href="#">  275</a>
<a href="#">Default Routing Engine Protocol Queue Assignments</a>	<a href="#">  309</a>
<a href="#">Default DSCP and DSCP IPv6 Classifiers</a>	<a href="#">  49</a>



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

### IN THIS SECTION

- [Requirements | 320](#)
- [Overview | 320](#)
- [Configuration | 321](#)
- [Verification | 326](#)

This example shows how to configure a supported router in an IPv4 network so that traffic generated by the Routing Engine and traffic generated by the distributed protocol handler are assigned to different non-default queues and marked with different non-default DiffServ code point (DSCP) bits on all egress interfaces.

This configuration enables you to configure network-wide prioritization to control plane protocol hello packets and keep-alive packets generated by the router. This feature is supported for egress interfaces.

### Requirements

This example uses the following hardware and software components:

- Two 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 non-default queues and marked with different non-default DSCP bits.

- Distributed protocol handler sourced traffic is placed in queue 7 on all egress interfaces. Of those packets, Layer 3 packets are marked at egress with DSCP bits 001010.
- Routing Engine sourced traffic is placed in queue 6 on all egress interfaces. Of those packets, Layer 3 packets are marked at egress with DSCP bits 000011.



Because the 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

### IN THIS SECTION

- [CLI Quick Configuration | 321](#)
- [Configuring R1 Packet Counting | 322](#)
- [Configuring R2 Queuing and Re-Marking of Host Outbound Traffic | 323](#)
- [Configuring R2 Queuing and Re-Marking of Routing Engine Sourced Traffic | 323](#)

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

### CLI Quick Configuration

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

#### Router R1

```
set firewall family inet filter f_bfd_source term 1 from forwarding-class control-traffic then
count c_sent_bfd
set firewall family inet filter f_bfd_source term 1 then accept
set firewall family inet filter f_bfd_source term 2 from forwarding-class-except control-traffic
```



```

then count c_sent_other
set firewall family inet filter f_bfd_source term 2 then accept
set forwarding-options family inet filter output bfd_source

```

## Router R2

```

set class-of-service forwarding-classes queue-num 7 bfd_keepalive
set class-of-service host-outbound-traffic forwarding-class bfd_keepalive
set class-of-service host-outbound-traffic dscp-code-point 110000
set class-of-service forwarding-classes queue-num 6 re_control
set firewall family inet filter f_out_loopback term 1 then forwarding-class re_control
set firewall family inet filter f_out_loopback term 1 then dscp 001010
set firewall family inet filter f_out_loopback term 1 then accept
set interfaces lo0 unit 0 family inet filter output f_out_loopback

```

## Configuring R1 Packet Counting

### Step-by-Step Procedure

To configure Router R1 to count packets that arrive marked for the `network-control` forwarding class:

1. Configure the IPv4 firewall filter term that counts packets marked for the `network-control` forwarding class.

```

[edit]
user@R1# set firewall family inet filter f_bfd_source term 1 from forwarding-class control-
traffic then count c_sent_bfd
user@R1# set firewall family inet filter f_bfd_source term 1 then accept

```

2. Configure the IPv4 firewall filter term that counts all other packets.

```

[edit]
user@R1# set firewall family inet filter f_bfd_source term 2 from forwarding-class-except
control-traffic then count c_sent_other
user@R1# set firewall family inet filter f_bfd_source term 2 then accept

```



3. Apply the firewall filter to all egress packets.

```
[edit]
user@R1# set forwarding-options family inet filter output bfd_source
```

## Configuring R2 Queuing and Re-Marking of Host Outbound Traffic

### Step-by-Step Procedure

To configure Router R2 to place host outbound traffic in queue 7 and re-mark Layer 3 packets with DSCP bits 110000:

1. Define the `bfd_keepalive` forwarding class and map it to queue 7.

```
[edit]
user@R2# set class-of-service forwarding-classes queue-num 7 bfd_keepalive
```

2. Configure the router to place distributed protocol handler sourced traffic (and also Routing Engine sourced traffic) in queue 7 on all egress interfaces.

```
[edit]
user@R2# set class-of-service host-outbound-traffic forwarding-class bfd_keepalive
```

3. Configure the router to re-mark Layer 3 distributed protocol handler sourced traffic (and also Routing Engine sourced traffic) with DSCP bits 110000, which is compatible with ToS bits 1100 0000.

```
[edit]
user@R2# set class-of-service host-outbound-traffic dscp-code-point 110000
```

## Configuring R2 Queuing and Re-Marking of Routing Engine Sourced Traffic

### Step-by-Step Procedure

To configure Router R2 to place Routing Engine sourced traffic only in queue 6 and re-mark Layer 3 packets with DSCP bits 001010:



1. Define the `re_control` forwarding class and map it to queue 6.

```
[edit]
user@R2# set class-of-service forwarding-classes queue-num 6 re_control
```

2. Define the IPv4 firewall filter `f_out_loopback` that places matched packets in queue 6, re-marks matched Layer 3 packets with DSCP bits 001010, and accepts all matched packets.

```
[edit]
user@R2# set firewall family inet filter f_out_loopback term 1 then forwarding-class
re_control
user@R2# set firewall family inet filter f_out_loopback term 1 then dscp 001010
user@R2# set firewall family inet filter f_out_loopback term 1 then accept
```

3. Attach the filter to the output of the router's loopback address so that the filter actions apply to Routing Engine sourced traffic only.

```
[edit]
user@R2# set interfaces lo0 unit 0 family inet filter output f_out_loopback
```

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

```
[edit]
user@R2# commit
```

## Results

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

### Router R1

```
user@R1# show firewall
family inet {
  filter f_bfd_source {
    term 1 {
      from {
```



```

        forwarding-class control-traffic;
    }
    then {
        count c_sent_bfd;
        accept;
    }
}
term 2 {
    from {
        forwarding-class-except control-traffic;
    }
    then {
        count c_sent_other;
        accept;
    }
}
}
}

```

```

user@R1# show forwarding-options
family inet {
    filter {
        output bfd_source;
    }
}

```

## Router R2

```

user@R2# show class-of-service
forwarding-classes {
    queue-num 6 re_control;
    queue-num 7 bfd_keepalive;
}
host-outbound-traffic {
    forwarding-class bfd_keepalive;
}

```



```
dscp-code-point 110000;
}
```

```
user@R2# show firewall
family inet {
    filter f_out_loopback {
        term 1 {
            then {
                forwarding-class re_control;
                dscp 001010;
                accept;
            }
        }
    }
}
```

```
user@R2# show interfaces
lo0 {
    unit 0 {
        family inet {
            filter {
                output f_out_loopback;
            }
        }
    }
}
```

## Verification

### IN THIS SECTION

- [Verifying the Queue Assignment of the Traffic That R1 Is Sending in the BFD Session | 327](#)
- [Verifying That Router R1 Is Sending BFD Traffic | 328](#)
- [Verifying That Router R2 Is Receiving BFD Traffic | 328](#)

Before you begin verification, enable BFD sessions on both routers.



Confirm that the configuration is working properly.

## Verifying the Queue Assignment of the Traffic That R1 Is Sending in the BFD Session

### Purpose

Verify the class of service (CoS) forwarding class assignments and type of traffic sent from the BFD source endpoint on Router R1.

### Action

From operational mode on Router R1, check that BFD packets are sent out the session endpoint on Router R1. With no CoS configuration present, the command output displays statistics about queued and transmitted traffic for the four forwarding classes and four egress queues in use.

```
user@R1> show interfaces queue ge-1/0/19 egress
Physical interface: ge-1/0/19, Enabled, Physical link is Up
  Interface index: 175, SNMP ifIndex: 121
Forwarding classes: 8 supported, 4 in use
Egress queues: 4 supported, 4 in use
Queue: 0, Forwarding classes: best-effort
  Queued:
    ...
  Transmitted:
    ...
Queue: 1, Forwarding classes: expedited-forwarding
  Queued:
    ...
  Transmitted:
    ...
Queue: 2, Forwarding classes: assured-forwarding
  Queued:
    ...
  Transmitted:
    ...
Queue: 3, Forwarding classes: network-control
  Queued:
    ...
  Transmitted:
    ...
```



## Meaning

The statistics for egress queue 3 reflect BFD session traffic sent to Router R2.

## Verifying That Router R1 Is Sending BFD Traffic

### Purpose

Verify that Router R1 is sending BFD packets from its BFD session endpoint.

### Action

From operational mode on Router R1, check that the count of BFD packets that R1 sends out the BFD session endpoint continues to increment.

```
user@R1> clear firewall filter f_bfd_source
user@R1> show firewall filter f_bfd_source
Filter: bfd_source
Counters:
Name                Bytes      Packets
c_sent_bfd          2770       70
c_sent_other         0          0
```

```
user@R1> show firewall filter f_bfd_source
Filter: bfd_source
Counters:
Name                Bytes      Packets
c_sent_bfd          2182022    39482
c_sent_other         0          0
```

## Verifying That Router R2 Is Receiving BFD Traffic

### Purpose

Verify that Router R2 is receiving BFD packets at its BFD session endpoint.



## Action

From operational mode on router R2, check that the BFD session endpoint receives packets destined for the Routing Engine with DSCP bits set to 110000, the default DSCP CoS value for the network-control forwarding class. The DSCP bits 110000 map to ToS bits 1100 0000, or 0xC0.

```
user@R2> monitor traffic extensive ge-1/1/0 layer2-headers
Address resolution is ON. Use <no-resolve> to avoid any reverse lookup delay.
Address resolution timeout is 4s.
Listening on ge-1/1/0, capture size 1514 bytes

03:23:10.830472 bpf_flags 0x83, In
  Juniper PCAP Flags [Ext, no-L2, In], PCAP Extension(s) total length 16
    Device Media Type Extension TLV #3, length 1, value: Ethernet (1)
    Logical Interface Encapsulation Extension TLV #6, length 1, value: Ethernet (14)
    Device Interface Index Extension TLV #1, length 2, value: 132
    Logical Interface Index Extension TLV #4, length 4, value: 68
  -----original packet-----
  PFE proto 2 (ipv4): (tos 0xc0, ttl 255, id 1511, offset 0, flags [none], proto: UDP
(17), length: 52) 10.1.1.1.bfd-src > 10.1.1.2.bfd-ip: [udp sum ok]
  BFDv1, length: 24
  One-hop Control, State Up, Flags: [Control Plane Independent], Diagnostic: No Diagnostic
(0x00)
  Detection Timer Multiplier: 3 (30000 ms Detection time), BFD Length: 24
  My Discriminator: 0x00000002, Your Discriminator: 0x00000001
  Desired min Tx Interval:    10000 ms
  Required min Rx Interval:   10000 ms
  Required min Echo Interval:    0 ms
```

## Meaning

The example input packet entry confirms that the original packet was marked with tos 0xC0, which correlates to the default forwarding class network-control.

## RELATED DOCUMENTATION

[Understanding Queuing and Marking of Host Outbound Traffic | 306](#)

[Change the Default Queuing and Marking of Host Outbound Traffic | 318](#)

*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](#)



# Defining Output Queue Properties with Schedulers

## IN THIS CHAPTER

- [How Schedulers Define Output Queue Properties | 332](#)
- [Default Schedulers Overview | 336](#)
- [Configuring Schedulers | 338](#)
- [Configuring Scheduler Maps | 338](#)
- [Applying Scheduler Maps Overview | 339](#)
- [Applying Scheduler Maps to Physical Interfaces | 340](#)
- [Configuring Traffic Control Profiles for Shared Scheduling and Shaping | 340](#)
- [Configuring an Input Scheduler on an Interface | 343](#)
- [Understanding Interface Sets | 344](#)
- [Configuring Interface Sets | 345](#)
- [Interface Set Caveats | 348](#)
- [Configuring Internal Scheduler Nodes | 350](#)
- [Example: Configuring and Applying Scheduler Maps | 351](#)

## How Schedulers Define Output Queue Properties

### IN THIS SECTION

- [Queue Scheduling Components | 334](#)

You use *schedulers* to define the class-of-service (CoS) properties of output queues. You configure CoS properties in a scheduler, then map the scheduler to a forwarding class. Forwarding classes are in turn mapped to output queues. Classifiers map incoming traffic into forwarding classes based on CoS values



in well-known packet header fields (behavior aggregate classification) or on multiple packet header fields (multifield classification).

Output queue properties include the amount of interface bandwidth assigned to the queue, the size of the memory buffer allocated for storing packets, the scheduling priority of the queue, and the random early detection (RED) drop profiles associated with the queue to control packet drop during periods of congestion.

Scheduler maps map schedulers to forwarding classes. The output queue mapped to a forwarding class receives the port resources and properties defined in the scheduler mapped to that forwarding class. You apply a scheduler map to an interface to apply queue scheduling to a port. You can associate different scheduler maps with different interfaces to configure port-specific scheduling for forwarding classes (output queues).

To configure class-of-service (CoS) schedulers, include the following statements at the [edit class-of-service] hierarchy level:

```
[edit class-of-service]
interfaces {
  interface-name {
    scheduler-map map-name;
    scheduler-map-chassis map-name;
    shaping-rate rate;
    unit {
      output-traffic-control-profile profile-name;
      scheduler-map map-name;
      shaping-rate rate;
    }
  }
}
fabric {
  scheduler-map {
    priority (high | low) scheduler scheduler-name;
  }
}
scheduler-maps {
  map-name {
    forwarding-class class-name scheduler scheduler-name;
  }
}
schedulers {
  scheduler-name {
    buffer-size (percent percentage | remainder | temporal
```



```

microseconds
);
    drop-profile-map loss-priority (any | low | medium-low | medium-high | high) protocol (any |
non-tcp | tcp) drop-profile profile-name;
    excess-priority (low | high);
    excess-rate percent percentage;
    excess-rate (percent percentage | proportion value);
    priority priority-level;
    transmit-rate (rate | percent percentage remainder) <exact | rate-limit>;
}
}
traffic-control-profiles profile-name {
    delay-buffer-rate (percent percentage | rate);
    excess-rate percent percentage;
    guaranteed-rate (percent percentage | rate);
    scheduler-map map-name;
    shaping-rate (percent percentage | rate);
}

```

You cannot configure both the shaping-rate statement at the [edit class-of-service interfaces *interface-name*] hierarchy level and the transmit-rate rate-limit statement and option at the [edit class-of-service schedulers *scheduler-name*] hierarchy level. These statements are mutually exclusive. If you do configure both, you will not be able to commit the configuration:

```

[edit class-of-service]
'shaping-rate'
only one option (shaping-rate or transmit-rate rate-limit) can be configured at a time
error: commit failed (statements constraint check failed)

```



**NOTE:** For PTX Series Packet Transport Routers:

- The fabric and traffic-control-profiles statements at the [edit class-of-service] hierarchy level are not supported.

## Queue Scheduling Components

Table 25 on page 335 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.
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"> <li>• Drop profile—Sets the probability of dropping packets as the queue fills up.</li> <li>• Loss priority—Sets the traffic packet loss priority to which a drop profile applies.</li> </ul>
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	<p>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.</p>



**Table 25: Output Queue Scheduler Components** *(Continued)*

Output Queue Scheduler Component	Description
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> <p><b>NOTE:</b> On PTX10008 and PTX10016 routers, the default scheduler transmission rate is set to 25 percent for strict-high queues if you do not set the transmission rate for the schedulers for strict-high queues.</p>

## RELATED DOCUMENTATION

[Understanding How Forwarding Classes Assign Classes to Output Queues | 275](#)

[Default Schedulers Overview | 336](#)

[Configuring Schedulers | 338](#)

[Priority Scheduling Overview | 430](#)

## 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 371](#).

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 478](#)
- ["Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers" on page 471](#)
- ["Configuring Scheduler Transmission Rate" on page 369](#)
- ["Configuring Schedulers for Priority Scheduling" on page 433](#)

## 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

[Applying Scheduler Maps to Physical Interfaces | 340](#)

[Applying Scheduler Maps and Shaping Rate to Physical Interfaces on IQ PICs](#)

[Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs | 391](#)

[Oversubscribing Interface Bandwidth | 356](#)

[Providing a Guaranteed Minimum Rate | 372](#)

[Applying Scheduler Maps to Chassis-Level Queues | 991](#)



## Applying Scheduler Maps to Physical Interfaces

After you have defined a scheduler map, as described in ["Configuring Scheduler Maps" on page 338](#), 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 391](#).



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

When you apply a scheduler map to a physical interface, or when you modify the configuration of a scheduler map that is already applied to a physical interface, packets already in the output queues of the interface might get dropped. The amount of packet loss is not deterministic and depends on the offered traffic load at the time you apply or modify the scheduler map.

## Configuring Traffic Control Profiles for Shared Scheduling and Shaping

Shared scheduling and shaping allows you to allocate separate pools of shared resources to subsets of logical interfaces belonging to the same physical port. You configure shared scheduling and shaping by first creating a traffic-control profile, which specifies a shaping rate and references a scheduler map. You must then share this set of shaping and scheduling resources by applying an instance of the traffic-control profile to a subset of logical interfaces. You can apply a separate instance of the same (or a



different) traffic-control profile to another subset of logical interfaces, thereby allocating separate pools of shared resources.

Before you start this procedure:

- Make sure you define a scheduler map. For information about configuring schedulers and scheduler maps, see ["Configuring Schedulers" on page 338](#) and ["Configuring Scheduler Maps" on page 338](#). Gigabit Ethernet IQ2 interfaces support up to eight forwarding classes and queues.

To configure a traffic-control profile, perform the following steps:

1. Create the traffic control profile and configure a shaping rate for it.

```
[edit]
user@host# edit class-of-service traffic-control-profiles profile-name
user@host# set shaping-rate (percent percentage | rate)
```

You can configure the shaping rate as a percentage from 1 through 100 or as an absolute rate from 1000 through 6,400,000,000,000 bits per second (bps). The shaping rate corresponds to a peak information rate (PIR). For more information, see ["Oversubscribing Interface Bandwidth" on page 356](#).

2. Define an association between the traffic-control profile and a previously configured scheduler map by including the scheduler-map statement at the [edit class-of-service traffic-control-profiles *profile-name*] hierarchy level.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set scheduler-map map-name;
```

3. Configure the delay-buffer rate.

If you do not include this statement, the delay-buffer rate is based on the guaranteed rate if one is configured, or on the shaping rate if no guaranteed rate is configured.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set delay-buffer-rate (percent percentage | rate)
```

You can configure the delay-buffer rate as a percentage from 1 through 100 or as an absolute rate from 1000 through 6,400,000,000,000 bits per second. The delay-buffer rate controls latency. For more information, see ["Oversubscribing Interface Bandwidth" on page 356](#) and ["Providing a Guaranteed Minimum Rate" on page 372](#).



4. Configure a guaranteed minimum rate for the traffic-control profile.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set guaranteed-rate (percent percentage | rate)
```

You can configure the guaranteed rate as a percentage from 1 through 100 or as an absolute rate from 1000 through 6,400,000,000,000 bps. The guaranteed rate corresponds to a committed information rate (CIR). For more information, see ["Providing a Guaranteed Minimum Rate" on page 372](#).

You must now share an instance of the traffic-control profile.

5. Enable shared-scheduling on the interface.

```
[edit]
user@host# edit interfaces interface-name
user@host# set shared-scheduler
```

This statement enables logical interfaces belonging to the same physical port to share one set of shaping and scheduling resources.



**NOTE:** On each physical interface, the shared-scheduler and per-unit-scheduler statements are mutually exclusive. Even so, you can configure one logical interface for each shared instance. This effectively provides the functionality of per-unit scheduling.

6. (Optional) Apply the traffic-control profile to an input interface.

```
[edit]
user@host# edit class-of-service interfaces interface-name unit logical-unit-number
user@host# set input-traffic-control-profile profile-name shared-instance instance-name
```

These statements are explained in [Step 7](#).

7. (Optional) Apply the traffic-control profile to an output interface.

```
[edit]
user@host# edit class-of-service interfaces interface-name unit logical-unit-number
user@host# set output-traffic-control-profile profile-name shared-instance instance-name
```

The profile name references the traffic-control profile you configured in [Step 1](#) through [Step 4](#). The shared-instance name does not reference a configuration. It can be any text string you wish to apply to



multiple logical interfaces that you want to share the set of resources configured in the traffic-control profile. Each logical interface shares a set of scheduling and shaping resources with other logical interfaces that are on the same physical port and that have the same shared-instance name applied.

This concept is demonstrated in ["Example: Configuring Shared Resources on Ethernet IQ2 Interfaces" on page 1049](#).



**NOTE:** You cannot include the `output-traffic-control-profile` statement in the configuration if either the `scheduler-map` or `shaping-rate` statement is included in the logical interface configuration.

## RELATED DOCUMENTATION

[Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | 760](#)

[Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs | 391](#)

[Hierarchical Schedulers and Traffic Control Profiles | 449](#)

## Configuring an Input Scheduler on an Interface

As an alternative to shared input traffic-control profiles, you can configure each interface to use its own input scheduler. For each physical interface, you can apply an input scheduler map to the physical interface or its logical interfaces, but not both.

Before you start this procedure:

- Define a scheduler map at the `[edit class-of-service scheduler-maps]` hierarchy level.

To configure a separate input scheduler on the physical interface:

- Specify the name of the physical interface and the scheduler map.

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

To configure a separate input scheduler on a logical interface:



1. Specify the name of the physical and logical interface and scheduler map.

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

2. Enable the association of the scheduler map name and interface.

```
[edit]
user@host# edit interfaces interface-name
user@host# set per-unit-scheduler
```

The per-unit-scheduler statement enables one set of output queues for each logical interface configured under the physical interface.

On Gigabit Ethernet IQ2 PIC interfaces, configuration of the per-unit-scheduler statement requires that you configure VLAN tagging also. When you include the per-unit-scheduler 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.

## RELATED DOCUMENTATION

[Configuring Schedulers | 338](#)

[Configuring Scheduler Maps | 338](#)

[input-traffic-control-profile](#)

## Understanding Interface Sets

Although the interface set is applied at the [edit interfaces] hierarchy level, the CoS parameters for the interface set are defined at the [edit class-of-service interfaces] hierarchy level, usually with the output-traffic-control-profile *profile-name* statement.

This example applies a traffic control profile called tcp-set1 to an interface set called set-ge-0:

```
[edit class-of-service interfaces]
interface-set set-ge-0 {
```



```
output-traffic-control-profile tcp-set1;
}
```

## RELATED DOCUMENTATION

| [output-traffic-control-profile](#)

## Configuring Interface Sets

To configure an interface set, include the `interface-set` statement at the [edit class-of-service interfaces] hierarchy level:

```
[edit class-of-service interfaces]
interface-set interface-set-name {
    ...interface-cos-configuration-statements ...
}
```

To apply the interface set to interfaces, include the `interface-set` statement at the [edit interfaces] hierarchy level:

```
[edit interfaces]
interface-set interface-set-name {
    interface ethernet-interface-name {
        ... interface-cos-configuration-statements ...
    }
}
```

Interface sets can be defined in two major ways:

- As a list of logical interfaces or aggregated Ethernet interfaces (unit 100, unit 200, and so on)
- At the stacked VLAN level using a list of outer VLAN IDs (vlan-tags-outer 210, vlan-tags-outer 220, and so on).

The `svlan number` listing option with a single outer VLAN tag is a convenient way to specify a set of VLAN members having the same outer VLAN tags. Service providers can use these statements to group interfaces to apply scheduling parameters such as guaranteed rate and shaping rate to the traffic in the groups.



Whether using the logical interface listing option for a group of customer VLANs, aggregated Ethernet interfaces, or the S-VLAN set listing option for a group of VLAN outer tags, all traffic heading downstream must be gathered into an interface set with the `interface-set` statement at the `[edit class-of-service interfaces]` hierarchy level.

Regardless of listing convention, you can only use one of the types in an interface set. Examples of this limitation appear later in this section.



**NOTE:** Interface sets are currently used only by CoS, but they are applied at the `[edit interfaces]` hierarchy level to make them available to other services that might use them in future.

```
[edit interfaces]
interface-set interface-set-name {
  interface ethernet-interface-name {
    (unit logical-unit-number | vlan-tags-outer vlan-tag) {
      ...
    }
  }
}
```

The logical interface naming option lists Ethernet interfaces:

```
[edit interfaces]
interface-set unitl-set-ge-0 {
  interface ge-0/0/0 {
    unit 0;
    unit 1;
    ...
  }
}
```

The interface naming option lists aggregated Ethernet interfaces:

```
[edit interfaces]
interface-set demuxset1 {
  interface demux0 {
    unit 1;
    ..
  }
}
```



```

}
demux0 {
    unit 1 {
        demux-options {
            underlying-interface ae0.1;
        }
        family inet {
            demux-source {
                10.1.1.1/24;
            }
            address 10.1.1.1/24;
        }
    }
    ..
    ae0 {
        unit 1 {
        }
        ..
    }
}
class-of-service {
    interface-set demuxset1 {
        output-traffic-control-profile tcp2;
    }
}
}

```

The S-VLAN option lists only one S-VLAN (outer) tag value:

```

[edit interfaces]
interface-set svlan-set {
    interface ge-1/0/0 {
        vlan-tags-outer 2000;
    }
}

```

The S-VLAN naming option lists S-VLAN (outer) tag values:

```

[edit interfaces]
interface-set svlan-set-tags {
    interface ge-2/0/0 {

```



```

    vlan-tags-outer 2000;
    vlan-tags-outer 2001;
    vlan-tags-outer 2002;
    ...
}
}

```



**NOTE:** Ranges are not supported: you must list each VLAN or logical interface separately.

## RELATED DOCUMENTATION

[Interface Set Caveats](#) | 348

## Interface Set Caveats

When configuring interface sets, consider the following guidelines:

- Interface sets can be defined in two major ways: as a list of logical interfaces or groups of aggregated Ethernet logical interfaces (unit 100, unit 200, and so on), or at the stacked VLAN level using a list of outer VLAN IDs (vlan-tags-outer 210, vlan-tags-outer 220, and so on). You can configure sets of aggregated Ethernet interfaces on MIC or MPC interfaces only.
- You cannot specify an interface set mixing the *logical interface*, aggregated Ethernet, S-VLAN, or VLAN outer tag list forms of the interface-set statement.
- Keep the following guidelines in mind when configuring interface sets of logical interfaces over aggregated Ethernet:
  - Sets of aggregated Ethernet interfaces are supported on MIC and MPC interfaces only.
  - The supported interface stacks for aggregated Ethernet in an interface set include VLAN demux interfaces, IP demux interfaces, and PPPoE logical interfaces over VLAN demux interfaces.
  - The link membership list and scheduler mode of the interface set are inherited from the underlying aggregated Ethernet interface over which the interface set is configured.
  - When an aggregated Ethernet interface operates in link protection mode, or if the scheduler mode is configured to replicate member links, the scheduling parameters of the interface set are copied to each of the member links.



- If the scheduler mode of the aggregated Ethernet interface is set to scale member links, the scheduling parameters are scaled based on the number of active member links and applied to each of the aggregated interface member links.
- A logical interface can only belong to one interface set. If you try to add the same logical interface to different interface sets, the commit operation fails.

This example generates a commit error:

```
[edit interfaces]
interface-set set-one {
  interface ge-2/0/0 {
    unit 0;
    unit 2;
  }
}
interface-set set-two {
  interface ge-2/0/0 {
    unit 1;
    unit 3;
    unit 0; # COMMIT ERROR! Unit 0 already belongs to set-one.
  }
}
```

- When you delete any interface configuration that is part of an interface set, the commit operation fails with the following message: "Interface unit should have configuration defined for adding to interface-set."
- Members of an interface set cannot span multiple physical interfaces. Only one physical interface is allowed to appear in an interface set.

This configuration is not supported:

```
[edit interfaces]
interface-set set-group {
  interface ge-0/0/1 {
    unit 0;
    unit 1;
  }
  interface ge-0/0/2 { # This is NOT supported in the same interface set!
    unit 0;
    unit 1;
  }
}
```



```
    }
}
```

## RELATED DOCUMENTATION

[Configuring Interface Sets](#) | 345

## Configuring Internal Scheduler Nodes

A node in the hierarchy is considered internal if either of the following conditions apply:

- Any one of its children nodes has a traffic control profile configured and applied.
- You include the `internal-node` statement at the `[edit class-of-service interfaces interface-set set-name]` hierarchy level.

Why would it be important to make a certain node internal? Generally, there are more resources available at the logical interface (unit) level than at the interface set level. Also, it might be desirable to configure all resources at a single level, rather than spread over several levels. The `internal-node` statement provides this flexibility. This can be a helpful configuration device when interface-set queuing without logical interfaces is used exclusively on the interface.

The `internal-node` statement can be used to raise the interface set without children to the same level as the other configured interface sets with children, allowing them to compete for the same set of resources.

In summary, using the `internal-node` statement allows statements to all be scheduled at the same level with or without children.

The following example makes the interfaces sets `if-set-1` and `if-set-2` internal:

```
[edit class-of-service interfaces]
interface-set {
  if-set-1 {
    internal-node;
    output-traffic-control-profile tcp-200m-no-smap;
  }
  if-set-2 {
    internal-node;
    output-traffic-control-profile tcp-100m-no-smap;
```



```
}
}
```

If an interface set has logical interfaces configured with a traffic control profile, then the use of the `internal-node` statement has no effect.

Internal nodes can specify a `traffic-control-profile-remaining` statement.

## Example: Configuring and Applying Scheduler Maps

### IN THIS SECTION

- [Requirements | 351](#)
- [Overview | 351](#)
- [Configuration | 352](#)
- [Verification | 354](#)

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

### Requirements

Before you begin:

- Create and configure the forwarding classes. See ["Configuring a Custom Forwarding Class for Each Queue" on page 281](#).
- 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 352](#).

**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

IN THIS SECTION

[Procedure | 352](#)

Procedure

CLI Quick Configuration

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

```
set class-of-service scheduler-maps diffserv-cos-map forwarding-class be-class scheduler be-scheduler
set class-of-service scheduler-maps diffserv-cos-map forwarding-class ef-class scheduler ef-scheduler
set class-of-service scheduler-maps diffserv-cos-map forwarding-class af-class scheduler af-scheduler
```



```
set class-of-service scheduler-maps diffserv-cos-map forwarding-class nc-class scheduler nc-scheduler
set class-of-service interfaces ge-0/0/0 unit 0 scheduler-map diffserv-cos-map
```

## Step-by-Step Procedure

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

To configure and apply a scheduler map to a device's interface:

1. Configure a scheduler map for DiffServ CoS.

```
[edit class-of-service]
user@host# edit scheduler-maps diffserv-cos-map
```

2. Configure a best-effort forwarding class and scheduler.

```
[edit class-of-service scheduler-maps diffserv-cos-map]
user@host# set forwarding-class be-class scheduler be-scheduler
```

3. Configure an expedited forwarding class and scheduler.

```
[edit class-of-service scheduler-maps diffserv-cos-map]
user@host# set forwarding-class ef-class scheduler ef-scheduler
```

4. Configure an assured forwarding class and scheduler.

```
[edit class-of-service scheduler-maps diffserv-cos-map]
user@host# set forwarding-class af-class scheduler af-scheduler
```

5. Configure a network control class and scheduler.

```
[edit class-of-service scheduler-maps diffserv-cos-map]
user@host# set forwarding-class nc-class scheduler nc-scheduler
```



## 6. Apply the scheduler map to an interface.

```
[edit class-of-service]
user@host# set interfaces ge-0/0/0 unit 0 scheduler-map diffserv-cos-map
```

## Results

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

```
[edit]
user@host# show class-of-service
interfaces {
  ge-0/0/0 {
    unit 0 {
      scheduler-map diffserv-cos-map;
    }
  }
}
scheduler-maps {
  diffserv-cos-map {
    forwarding-class be-class scheduler be-scheduler;
    forwarding-class ef-class scheduler ef-scheduler;
    forwarding-class af-class scheduler af-scheduler;
    forwarding-class nc-class scheduler nc-scheduler;
  }
}
```

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

## Verification

### IN THIS SECTION

- [Verifying the Scheduler Map Configuration | 355](#)



# 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

<a href="#">Default Schedulers Overview</a>	<a href="#">  336</a>
<a href="#">Configuring Schedulers</a>	<a href="#">  338</a>
<a href="#">Configuring Scheduler Maps</a>	<a href="#">  338</a>



# Controlling Bandwidth with Scheduler Rates

## IN THIS CHAPTER

- [Oversubscribing Interface Bandwidth | 356](#)
- [Configuring Scheduler Transmission Rate | 369](#)
- [Providing a Guaranteed Minimum Rate | 372](#)
- [PIR-Only and CIR Mode | 377](#)
- [Excess Rate and Excess Priority Configuration Examples | 378](#)
- [Controlling Remaining Traffic | 385](#)
- [Bandwidth Sharing on Nonqueueing Packet Forwarding Engines Overview | 389](#)
- [Configuring Rate Limits on Nonqueueing Packet Forwarding Engines | 390](#)
- [Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs | 391](#)
- [Example: Applying Scheduler Maps and Shaping Rate to DLCIs | 401](#)
- [Example: Applying Scheduling and Shaping to VLANs | 407](#)
- [Applying a Shaping Rate to Physical Interfaces Overview | 417](#)
- [Configuring the Shaping Rate for Physical Interfaces | 417](#)
- [Example: Limiting Egress Traffic on an Interface Using Port Shaping for CoS | 419](#)
- [Configuring Input Shaping Rates for Both Physical and Logical Interfaces | 428](#)

## Oversubscribing Interface Bandwidth

### IN THIS SECTION

- [Verifying Configuration of Bandwidth Oversubscription | 364](#)
- [Examples: Oversubscribing Interface Bandwidth | 364](#)



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 391](#).

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 372](#).

For more information about Gigabit Ethernet IQ2 PICs, see ["CoS on Enhanced IQ2 PICs Overview" on page 1005](#).

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 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 478](#). For an example showing how the delay-buffer rates are applied, see ["Examples: Oversubscribing Interface Bandwidth" on page 364](#).

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 338](#) and ["Configuring Scheduler Maps" on page 338](#).

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 478](#).

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;
```



- 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 361 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.	<p>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 <a href="#">"Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size"</a> on page 478.</p> <p>Unshaped logical interfaces receive the remaining bandwidth and a delay buffer in proportion to the remaining bandwidth.</p>
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
<p>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>	<p>Logical interface receives a delay buffer based on the scaled shaping rate:</p> $\text{scaled shaping rate} = \frac{(\text{shaping-rate} * [\text{physical interface bandwidth}])}{\text{SUM}(\text{shaping-rates of all logical interfaces on the physical interface})}$ <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>



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

Configuration Scenario	Delay Buffer Allocation
<p>You oversubscribe the interface. You configure a shaping rate. You configure a delay-buffer rate.</p>	<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 &lt;= 10 Mbps: 400-millisecond (ms) delay buffer  delay-buffer-rate &lt;= 20 Mbps: 300-ms delay buffer  delay-buffer-rate &lt;= 30 Mbps: 200-ms delay buffer  delay-buffer-rate &lt;= 40 Mbps: 150-ms delay buffer  delay-buffer-rate &gt; 40 Mbps: 100-ms delay buffer</p> <p>On LSQ DLCIs, if total bundle bandwidth &lt; T1 bandwidth:</p> <p>delay-buffer-rate = 1 second</p> <p>On LSQ DLCIs, if total bundle bandwidth &gt;= 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 <a href="#">"Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 478</a>.</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>
<p>You oversubscribe the interface. You do not configure a shaping rate. You configure a guaranteed rate. You configure a delay-buffer rate.</p>	<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 478](#).

```
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

### IN THIS SECTION

- [Example: Configuring Scheduler Transmission Rate | 371](#)
- [Allocation of Leftover Bandwidth | 371](#)

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 338](#).

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



## 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 | 433](#)

[How Schedulers Define Output Queue Properties | 332](#)

[Configuring a Scheduler | 670](#)

*excess-rate*

*schedulers*

## Providing a Guaranteed Minimum Rate

### IN THIS SECTION

- [Verifying Configuration of Guaranteed Minimum Rate | 376](#)
- [Example: Providing a Guaranteed Minimum Rate | 376](#)

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-profiles *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" on page 1005](#).

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 478](#). For an example showing how the delay-buffer rates are applied, see ["Example: Providing a Guaranteed Minimum Rate" on page 376](#).



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 338](#) and ["Configuring Scheduler Maps" on page 338](#).



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 478](#).

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 375](#) 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.



**Table 28: Bandwidth and Delay Buffer Allocations by Configuration Scenario (Continued)**

Configuration Scenario	Delay Buffer Allocation
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 <code>q-pic-large-buffer</code> statement. For more information, see <a href="#">"Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 478</a> .
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 <code>q-pic-large-buffer</code> statement. For more information, see <a href="#">"Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 478</a> .

## Verifying Configuration of Guaranteed Minimum Rate

To verify your configuration, you can issue this following operational mode commands:

- `show class-of-service interfaces`
- `show class-of-service traffic-control-profile profile-name`

## Example: Providing a Guaranteed Minimum Rate

Two logical interface units, 0 and 1, are provisioned with a guaranteed minimum of 750 Kbps and 500 Kbps, respectively. For logical unit 1, the delay buffer is based on the guaranteed rate setting. For logical unit 0, a delay-buffer rate of 500 Kbps is specified. The actual delay buffers allocated to each logical interface are 2 seconds of 500 Kbps. The 2-second value is based on the following calculation:

```
delay-buffer-rate < [8 x 64 Kbps]): 2 seconds of delay-buffer-rate
```

For more information about this calculation, see ["Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 478](#).

```
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).



**NOTE:** For nodes that do not have a guaranteed rate configured, Juniper recommends that you configure a delay buffer rate with the same value as the shaping rate.

## Excess Rate and Excess Priority Configuration Examples

To configure the excess rate for nonqueuing Packet Forwarding Engines, include the *excess-rate* statement at the [edit class-of-service schedulers *scheduler-name*] hierarchy level.

To configure the excess priority for nonqueuing Packet Forwarding Engines, include the *excess-priority* statement at the [edit class-of-service schedulers *scheduler-name*] hierarchy level.

The relationship between the configured guaranteed rate, excess rate, guaranteed priority, excess priority, and offered load is not always obvious. The following tables show the expected throughput of a Gigabit Ethernet port with various bandwidth-sharing parameters configured on the queues.

The default behavior of a nonqueuing Gigabit Ethernet interface with multiple priority levels is shown in [Table 29 on page 378](#). 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



**Table 29: Current Behavior with Multiple Priority Levels (Continued)**

Queue	Guaranteed (Transmit) Rate	Guaranteed Priority	Excess Priority	Offered Load	Expected Throughput
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 379](#). 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 380](#). 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 380](#). 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 381](#). 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 381](#). 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 382](#). 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 382](#). 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 383](#). 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;
        }
    }
}
```



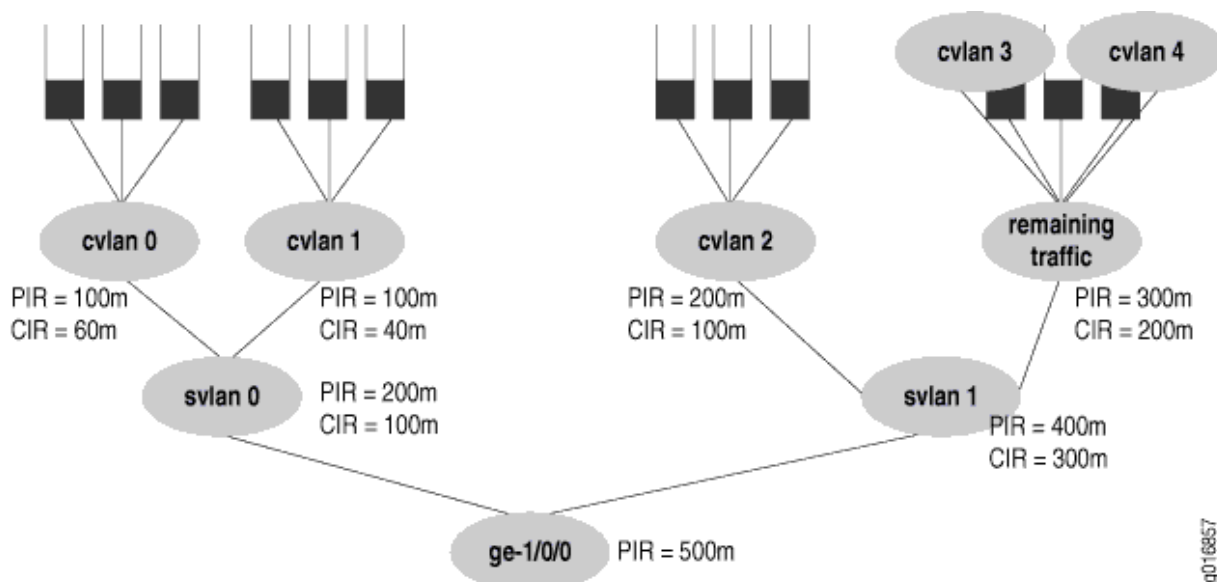
```
}
}
```

## 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 32 on page 385](#). 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 32: 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.

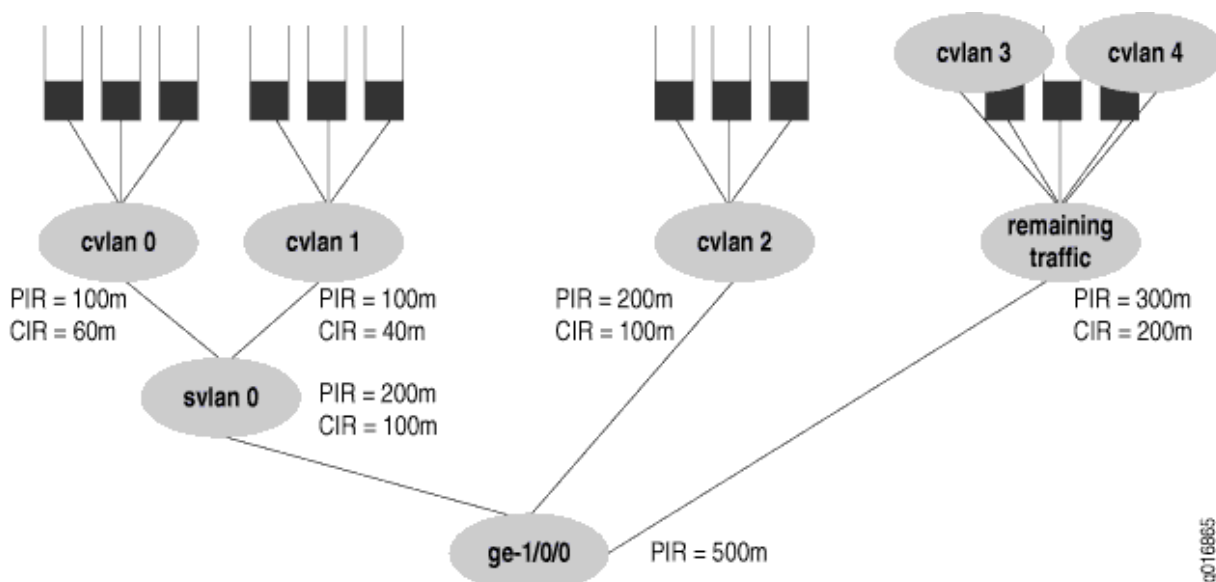
```
[edit class-of-service interfaces]
interface-set svlan0 {
    output-traffic-control-profile tcp-svlan0;
}
interface-set svlan1 {
    output-traffic-control-profile tcp-svlan1; # For explicitly shaped traffic.
    output-traffic-control-profile-remaining tcp-svlan1-remaining; # For all remaining 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 33 on page 387](#).



Figure 33: 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-if11 specifies scheduling and queuing for ge-1/0/0 unit 1.

This example does not include the [edit interfaces] configuration.

```
[edit class-of-service interfaces]
interface-set {
  svlan0 {
```



```

        output-traffic-control-profile tcp-svlan0; # Guarantee & shaper for svlan0.
    }
}
ge-1/0/0 {
    output-traffic-control-profile-remaining tcp-svlan0-rem;
    # Unit 3 and 4 are not explicitly configured, but captured by "remaining"
    unit 1 {
        output-traffic-control-profile tcp-ifl1; # Unit 1 be & ef queues.
    }
}

```

Here is how the traffic control profiles for this example are configured:

```

[edit class-of-service traffic-control-profiles]
tcp-svlan0 {
    shaping-rate 200m;
    guaranteed-rate 100m;
}
tcp-svlan0-rem {
    shaping-rate 300m;
    guaranteed-rate 200m;
    scheduler-map smap-svlan0-rem; # This specifies queues for remaining traffic
}
tcp-ifl1 {
    scheduler-map smap-ifl1;
}

```

Finally, here are the scheduler maps and queues for the example:

```

[edit class-of-service scheduler-maps]
smap-svlan0-rem {
    forwarding-class best-effort scheduler sched-foo;
}
smap-ifl1 {
    forwarding-class best-effort scheduler sched-bar;
    forwarding-class assured-forwarding scheduler sched-baz;
}

```

The configuration for the referenced schedulers are not given for this example.



## Bandwidth Sharing on Nonqueuing Packet Forwarding Engines Overview

You can configure bandwidth sharing rate limits, excess rate, and excess priority at the queue level on the following Juniper Networks routers and switches:

- EX Series switches
- 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 nonqueuing DPCs (rate limit, excess rate, and excess priority)

You configure rate limits when you have a concern that low-latency packets (such as high or strict-high priority packets for voice) might starve low-priority and medium-priority packets. In Junos OS, the low latency queue is implemented by rate-limiting packets to the transmit bandwidth. The rate-limiting is performed immediately before queuing the packet for transmission. All packets that exceed the rate limit are not queued, but dropped.

By default, if the excess priority is not configured for a queue, the excess priority will be the same as the normal queue priority. If none of the queues have an excess rate configured, then the excess rate will be the same as the transmit rate percentage. If at least one of the queues has an excess rate configured, then the excess rate for the queues that do not have an excess rate configured will be set to zero.

When the physical interface is on queuing hardware 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 209](#) 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 356](#).

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 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" on page 1005](#).

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 394 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
<b>IQ PICs</b>			
Physical interfaces	ATM2 IQ	Yes	<p>Example of supported configuration:</p> <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	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces t1-0/0/0:1] scheduler-map map-1;</pre>
Logical interfaces (DLCIs and VLANs only) configured on IQ PICs	Gigabit Ethernet IQ with VLAN tagging enabled	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces ge-0/0/0 unit 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
	E3 IQ with Frame Relay encapsulation	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces e3-0/0/0 unit 1] scheduler-map map-1;</pre>
	Channelized OC3 IQ with Frame Relay encapsulation	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces t1-1/0/0:1:1 unit 0] scheduler-map map-1;</pre>
	Channelized STM1 IQ with Frame Relay encapsulation	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces e1-0/0/0:1 unit 1] scheduler-map map-1;</pre>
	Channelized T3 IQ with Frame Relay encapsulation	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces t1-0/0/0 unit 1] scheduler-map map-1;</pre>
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**



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

Interface Type	PIC Type	Supported	Example Configuration
Physical interfaces	T3	Yes	Example of supported configuration:  [edit class-of-service interfaces t3-0/0/0] scheduler-map map-1;
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 396](#) 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
<b>Fixed Configuration MPCs</b>			



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

MPC	MIC	Supported	Example Configuration
16x10GE MPC	No	Yes	[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;
32x10GE MPC4E	No	Yes	[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;
2x100GE + 8x10GE MPC4E	No	Yes	[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;
6x40GE + 24x10GE MPC5E	No	No	No
6x40GE + 24x10GE MPC5EQ	No	Yes	[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;
2x100GE + 4x10GE MPC5E	No	No	No
2x100GE + 4x10GE MPC5EQ	No	Yes	[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;
<b>MPCs</b>			
MPC1	No	No	No
MPC1E	No	No	No



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

MPC	MIC	Supported	Example Configuration
MPC1 Q	Any supported MIC	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;</pre>
MPC1E Q	Any supported MIC	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;</pre>
MPC2	No	No	No
MPC2E	No	No	No
MPC2 Q	Any supported MIC	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;</pre>
MPC2E Q	Any supported MIC	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;</pre>
MPC2 EQ	Any supported MIC	Yes	<p>Example of supported configuration:</p> <pre>[edit class-of-service interfaces ge-0/0/0 unit 1] scheduler-map map-1;</pre>



**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" on page 1102](#) for scaling information on non-queuing MPCs.

2. Associate a scheduler with the interface by including the `scheduler-map` statement at the `[edit class-of-service interfaces interface-name unit logical-unit-number]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
scheduler-map map-name;
```

Alternatively, associate a scheduler with the interface by including the `scheduler-map` statement at the `[edit class-of-service traffic-control-profiles traffic control profile name]` hierarchy level and then include the `output-traffic-control-profile` statement at the `[edit class-of-service interfaces interface name unit logical unit number]` hierarchy level.

```
[edit class-of-service traffic-control-profiles traffic control profile name]
scheduler-map map-name;
```

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
output-traffic-control-profile traffic-control-profile-name;
```

3. Configure shaping on the interface by including the `shaping-rate` statement at the `[edit class-of-service interfaces interface-name unit logical-unit-number]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
shaping-rate rate;
```





**NOTE:** You can also apply the shaping rate to the traffic control profile.

By default, the logical interface bandwidth is the average of unused bandwidth for the number of logical interfaces that require default bandwidth treatment. You can specify a peak bandwidth rate in bps, either as a complete decimal number or as a decimal number followed by the abbreviation k (1000), m (1,000,000), or g (1,000,000,000). 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 | 407](#)

[Example: Applying Scheduler Maps and Shaping Rate to DLCIs | 401](#)

## Example: Applying Scheduler Maps and Shaping Rate to DLCIs

### IN THIS SECTION

- [Requirements | 401](#)
- [Overview | 402](#)
- [Configuration | 402](#)

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

### 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 407](#).

## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 403](#)
- [Procedure | 404](#)
- [Results | 405](#)



## 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

```



## Procedure

### Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see the [Junos OS CLI User Guide](#).

1. Configure the device interfaces.

```
[edit interfaces]
user@PE1# set 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 dlci 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 dlci 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 | 391](#)

[Example: Applying Scheduling and Shaping to VLANs | 407](#)

*per-unit-scheduler*

## Example: Applying Scheduling and Shaping to VLANs

### IN THIS SECTION

- [Requirements | 407](#)
- [Overview | 408](#)
- [Configuration | 408](#)

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

### 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

### IN THIS SECTION

- [CLI Quick Configuration | 408](#)
- [Procedure | 410](#)
- [Results | 413](#)

### CLI Quick Configuration

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

```
set interfaces xe-9/0/3 per-unit-scheduler
set interfaces xe-9/0/3 vlan-tagging
set interfaces xe-9/0/3 unit 1 vlan-id 101
set interfaces xe-9/0/3 unit 1 family inet address 10.1.1.1/24
set interfaces xe-9/0/3 unit 2 vlan-id 102
set interfaces xe-9/0/3 unit 2 family inet address 10.2.1.1/24
set class-of-service classifiers inet-precedence c8 forwarding-class be loss-priority low code-points 000
```



```

set class-of-service classifiers inet-precedence c8 forwarding-class ef loss-priority low code-
points 001
set class-of-service classifiers inet-precedence c8 forwarding-class af loss-priority low code-
points 010
set class-of-service classifiers inet-precedence c8 forwarding-class nc loss-priority low code-
points 011
set class-of-service classifiers inet-precedence c8 forwarding-class be1 loss-priority low code-
points 100
set class-of-service classifiers inet-precedence c8 forwarding-class ef1 loss-priority low code-
points 101
set class-of-service classifiers inet-precedence c8 forwarding-class af1 loss-priority low code-
points 110
set class-of-service classifiers inet-precedence c8 forwarding-class nc1 loss-priority low code-
points 111
set class-of-service forwarding-classes queue 0 be
set class-of-service forwarding-classes queue 1 ef
set class-of-service forwarding-classes queue 2 af
set class-of-service forwarding-classes queue 3 nc
set class-of-service forwarding-classes queue 4 be1
set class-of-service forwarding-classes queue 5 ef1
set class-of-service forwarding-classes queue 6 af1
set class-of-service forwarding-classes queue 7 nc1
set class-of-service traffic-control-profiles tcp_ifd shaping-rate 2500000000
set class-of-service traffic-control-profiles tcp_ifd overhead-accounting bytes -20
set class-of-service traffic-control-profiles tcp_gold scheduler-map gold
set class-of-service traffic-control-profiles tcp_gold shaping-rate 2500000000
set class-of-service traffic-control-profiles tcp_gold overhead-accounting bytes -20
set class-of-service traffic-control-profiles tcp_gold guaranteed-rate 1g
set class-of-service traffic-control-profiles tcp_silver scheduler-map silver
set class-of-service traffic-control-profiles tcp_silver shaping-rate 1g
set class-of-service traffic-control-profiles tcp_silver overhead-accounting bytes -20
set class-of-service traffic-control-profiles tcp_silver guaranteed-rate 500m
set class-of-service interfaces xe-9/0/3 output-traffic-control-profile tcp_ifd
set class-of-service interfaces xe-9/0/3 unit 1 output-traffic-control-profile tcp_gold
set class-of-service interfaces xe-9/0/3 unit 2 output-traffic-control-profile tcp_silver
set class-of-service scheduler-maps gold forwarding-class be1 scheduler gold_internet
set class-of-service scheduler-maps gold forwarding-class ef1 scheduler gold_video
set class-of-service scheduler-maps gold forwarding-class af1 scheduler gold_voice
set class-of-service scheduler-maps gold forwarding-class nc1 scheduler gold_reserved
set class-of-service scheduler-maps silver forwarding-class be scheduler silver_internet
set class-of-service scheduler-maps silver forwarding-class ef scheduler silver_video
set class-of-service scheduler-maps silver forwarding-class af scheduler silver_voice
set class-of-service scheduler-maps silver forwarding-class nc scheduler silver_reserved

```



```

set class-of-service schedulers gold_internet excess-rate percent 40
set class-of-service schedulers gold_internet buffer-size percent 20
set class-of-service schedulers gold_internet priority low
set class-of-service schedulers gold_video transmit-rate percent 50
set class-of-service schedulers gold_video buffer-size percent 50
set class-of-service schedulers gold_voice shaping-rate percent 10
set class-of-service schedulers gold_voice buffer-size percent 10
set class-of-service schedulers gold_voice priority strict-high
set class-of-service schedulers gold_reserved excess-rate percent 20
set class-of-service schedulers gold_reserved buffer-size percent 10
set class-of-service schedulers gold_reserved priority low
set class-of-service schedulers silver_internet excess-rate percent 40
set class-of-service schedulers silver_internet buffer-size percent 20
set class-of-service schedulers silver_internet priority low
set class-of-service schedulers silver_video transmit-rate percent 50
set class-of-service schedulers silver_video buffer-size percent 50
set class-of-service schedulers silver_voice shaping-rate percent 10
set class-of-service schedulers silver_voice buffer-size percent 10
set class-of-service schedulers silver_voice priority strict-high
set class-of-service schedulers silver_reserved excess-rate percent 20
set class-of-service schedulers silver_reserved buffer-size percent 10
set class-of-service schedulers silver_reserved priority low

```

## Procedure

### Step-by-Step Procedure

The following example requires you to navigate various levels in the configuration hierarchy. For information about navigating the CLI, see the [Junos OS CLI User Guide](#).

1. Configure the device interfaces.

```

[edit interfaces]
user@PE1# set xe-9/0/3 per-unit-scheduler
user@PE1# set xe-9/0/3 vlan-tagging
user@PE1# set xe-9/0/3 unit 1 vlan-id 101
user@PE1# set xe-9/0/3 unit 1 family inet address 10.1.1.1/24
user@PE1# set xe-9/0/3 unit 2 vlan-id 102
user@PE1# set xe-9/0/3 unit 2 family inet address 10.2.1.1/24

```



## 2. Configure the classifiers.

```
[edit class-of-service]
user@PE1# set classifiers inet-precedence c8 forwarding-class be loss-priority low code-
points 000
user@PE1# set classifiers inet-precedence c8 forwarding-class ef loss-priority low code-
points 001
user@PE1# set classifiers inet-precedence c8 forwarding-class af loss-priority low code-
points 010
user@PE1# set classifiers inet-precedence c8 forwarding-class nc loss-priority low code-
points 011
user@PE1# set classifiers inet-precedence c8 forwarding-class be1 loss-priority low code-
points 100
user@PE1# set classifiers inet-precedence c8 forwarding-class ef1 loss-priority low code-
points 101
user@PE1# set classifiers inet-precedence c8 forwarding-class af1 loss-priority low code-
points 110
user@PE1# set classifiers inet-precedence c8 forwarding-class nc1 loss-priority low code-
points 111
```

## 3. Configure the forwarding classes.

```
[edit class-of-service]
user@PE1# set forwarding-classes queue 0 be
user@PE1# set forwarding-classes queue 1 ef
user@PE1# set forwarding-classes queue 2 af
user@PE1# set forwarding-classes queue 3 nc
user@PE1# set forwarding-classes queue 4 be1
user@PE1# set forwarding-classes queue 5 ef1
user@PE1# set forwarding-classes queue 6 af1
user@PE1# set forwarding-classes queue 7 nc1
```

## 4. Configure the traffic control profiles.

```
[edit class-of-service]
user@PE1# set traffic-control-profiles tcp_ifd shaping-rate 2500000000
user@PE1# set traffic-control-profiles tcp_ifd overhead-accounting bytes -20
user@PE1# set traffic-control-profiles tcp_gold scheduler-map gold
user@PE1# set traffic-control-profiles tcp_gold shaping-rate 2500000000
user@PE1# set traffic-control-profiles tcp_gold overhead-accounting bytes -20
```



```

user@PE1# set traffic-control-profiles tcp_gold guaranteed-rate 1g
user@PE1# set traffic-control-profiles tcp_silver scheduler-map silver
user@PE1# set traffic-control-profiles tcp_silver shaping-rate 1g
user@PE1# set traffic-control-profiles tcp_silver overhead-accounting bytes -20
user@PE1# set traffic-control-profiles tcp_silver guaranteed-rate 500m

```

5. Map the traffic control profiles to their respective physical or logical interface.

```

[edit class-of-service]
user@PE1# set interfaces xe-9/0/3 output-traffic-control-profile tcp_ifd
user@PE1# set interfaces xe-9/0/3 unit 1 output-traffic-control-profile tcp_gold
user@PE1# set interfaces xe-9/0/3 unit 2 output-traffic-control-profile tcp_silver

```

6. Configure the scheduler maps.

```

[edit class-of-service]
user@PE1# set scheduler-maps gold forwarding-class be1 scheduler gold_internet
user@PE1# set scheduler-maps gold forwarding-class ef1 scheduler gold_video
user@PE1# set scheduler-maps gold forwarding-class af1 scheduler gold_voice
user@PE1# set scheduler-maps gold forwarding-class nc1 scheduler gold_reserved
user@PE1# set scheduler-maps silver forwarding-class be scheduler silver_internet
user@PE1# set scheduler-maps silver forwarding-class ef scheduler silver_video
user@PE1# set scheduler-maps silver forwarding-class af scheduler silver_voice
user@PE1# set scheduler-maps silver forwarding-class nc scheduler silver_reserved

```

7. Configure the schedulers.

```

[edit class-of-service]
user@PE1# set schedulers gold_internet excess-rate percent 40
user@PE1# set schedulers gold_internet buffer-size percent 20
user@PE1# set schedulers gold_internet priority low
user@PE1# set schedulers gold_video transmit-rate percent 50
user@PE1# set schedulers gold_video buffer-size percent 50
user@PE1# set schedulers gold_voice shaping-rate percent 10
user@PE1# set schedulers gold_voice buffer-size percent 10
user@PE1# set schedulers gold_voice priority strict-high
user@PE1# set schedulers gold_reserved excess-rate percent 20
user@PE1# set schedulers gold_reserved buffer-size percent 10
user@PE1# set schedulers gold_reserved priority low
user@PE1# set schedulers silver_internet excess-rate percent 40

```



```

user@PE1# set schedulers silver_internet buffer-size percent 20
user@PE1# set schedulers silver_internet priority low
user@PE1# set schedulers silver_video transmit-rate percent 50
user@PE1# set schedulers silver_video buffer-size percent 50
user@PE1# set schedulers silver_voice shaping-rate percent 10
user@PE1# set schedulers silver_voice buffer-size percent 10
user@PE1# set schedulers silver_voice priority strict-high
user@PE1# set schedulers silver_reserved excess-rate percent 20
user@PE1# set schedulers silver_reserved buffer-size percent 10
user@PE1# set schedulers silver_reserved priority low

```

## Results

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

```

user@PE1# show interfaces
interfaces {
  xe-9/0/3 {
    per-unit-scheduler;
    vlan-tagging;
    unit 1 {
      vlan-id 101;
      family inet {
        address 10.1.1.1/24;
      }
    }
    unit 2 {
      vlan-id 102;
      family inet {
        address 10.2.1.1/24;
      }
    }
  }
}

```

```

user@PE1# show class-of-service
class-of-service {
  classifiers {

```



```

inet-precedence c8 {
    forwarding-class be {
        loss-priority low code-points 000;
    }
    forwarding-class ef {
        loss-priority low code-points 001;
    }
    forwarding-class af {
        loss-priority low code-points 010;
    }
    forwarding-class nc {
        loss-priority low code-points 011;
    }
    forwarding-class be1 {
        loss-priority low code-points 100;
    }
    forwarding-class ef1 {
        loss-priority low code-points 101;
    }
    forwarding-class af1 {
        loss-priority low code-points 110;
    }
    forwarding-class nc1 {
        loss-priority low code-points 111;
    }
}

forwarding-classes {
    queue 0 be;
    queue 1 ef;
    queue 2 af;
    queue 3 nc;
    queue 4 be1;
    queue 5 ef1;
    queue 6 af1;
    queue 7 nc1;
}

traffic-control-profiles {
    tcp_ifd {
        shaping-rate 2500000000;
        overhead-accounting bytes -20;
    }
    tcp_gold {

```



```

        scheduler-map gold;
        shaping-rate 2500000000;
        overhead-accounting bytes -20;
        guaranteed-rate 1g;
    }
    tcp_silver {
        scheduler-map silver;
        shaping-rate 1g;
        overhead-accounting bytes -20;
        guaranteed-rate 500m;
    }
}
interfaces {
    xe-9/0/3 {
        output-traffic-control-profile tcp_ifd;
        unit 1 {
            output-traffic-control-profile tcp_gold;
        }
        unit 2 {
            output-traffic-control-profile tcp_silver;
        }
    }
}
scheduler-maps {
    gold {
        forwarding-class be1 scheduler gold_internet;
        forwarding-class ef1 scheduler gold_video;
        forwarding-class af1 scheduler gold_voice;
        forwarding-class nc1 scheduler gold_reserved;
    }
    silver {
        forwarding-class be scheduler silver_internet;
        forwarding-class ef scheduler silver_video;
        forwarding-class af scheduler silver_voice;
        forwarding-class nc scheduler silver_reserved;
    }
}
schedulers {
    gold_internet {
        excess-rate percent 40;
        buffer-size percent 20;
        priority low;
    }
}

```



```

gold_video {
    transmit-rate percent 50;
    buffer-size percent 50;
}
gold_voice {
    shaping-rate percent 10;
    buffer-size percent 10;
    priority strict-high;
}
gold_reserved {
    excess-rate percent 20;
    buffer-size percent 10;
    priority low;
}
silver_internet {
    excess-rate percent 40;
    buffer-size percent 20;
    priority low;
}
silver_video {
    transmit-rate percent 50;
    buffer-size percent 50;
}
silver_voice {
    shaping-rate percent 10;
    buffer-size percent 10;
    priority strict-high;
}
silver_reserved {
    excess-rate percent 20;
    buffer-size percent 10;
    priority low;
}
}
}

```

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

## RELATED DOCUMENTATION

[Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs](#) | 391



## Applying a Shaping Rate to Physical Interfaces Overview

On supported platforms, you can configure physical interfaces to shape traffic based on the rate-limited bandwidth of the total interface bandwidth. This allows you to shape the output of the physical interface, so that the interface transmits less traffic than it is physically capable of carrying.

If you do not configure a shaping rate on the physical interface, the default physical interface bandwidth is based on the channel bandwidth and the time slot allocation.

In general, the physical interface speed is the basis for calculating the various queue parameters for a physical interface such as delay buffer size, weighted round-robin (WRR) weight, drop profile, and so forth. However, when you apply a shaping rate by including the `shaping-rate` statement, the shaping rate on that physical interface becomes the basis for calculating all the queue parameters for that physical interface.

The minimum shaping rate value for the physical interface depends on the platform and line card. The maximum value of shaping rate is limited by the maximum transmission rate of the interface.

### RELATED DOCUMENTATION

[Configuring the Shaping Rate for Physical Interfaces | 417](#)

## Configuring the Shaping Rate for Physical Interfaces

To configure the shaping rate on the physical interface, either include the `shaping-rate` statement at the `[edit class-of-service interfaces interface-name]` hierarchy level or include the `output-traffic-control-profile` statement at the `[edit class-of-service interfaces interface-name]` hierarchy level.

You can specify a peak bandwidth rate in bps, either as a complete decimal number or as a decimal number followed by the abbreviation k (1000), m (1,000,000), or g (1,000,000,000). For physical interfaces, the range is from 1000 through 6,400,000,000,000 bps.

The maximum value of `shaping-rate` is limited by the maximum transmission rate of the interface.

The following are two sample configurations for applying a shaping rate of 5 Gbps on a physical interface (xe-4/0/0):



Apply a shaping rate at the [edit class-of-service interfaces *interface-name*] hierarchy:

```
[edit class-of-service]
interfaces {
  xe-4/0/0 {
    shaping-rate 5g;
  }
}
```

Apply a shaping rate using a traffic control profile:

```
[edit class-of-service]
traffic-control-profiles {
  shaping-output {
    shaping-rate 5g;
  }
}
interfaces {
  xe-4/0/0 {
    output-traffic-control-profile shaping-output;
  }
}
```

To view the results of your configuration, issue the following `show` commands:

- `show class-of-service interface interface-name`
- `show interfaces interface-name extensive`

## RELATED DOCUMENTATION

| [Applying a Shaping Rate to Physical Interfaces Overview](#) | 417



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

### IN THIS SECTION

- [Requirements | 419](#)
- [Overview | 419](#)
- [Configuration | 420](#)
- [Verification | 426](#)

This example shows how using port shaping as a form of class of service (CoS) enables you to limit traffic on an interface, so that you can control the amount of traffic passing through the interface.

### Requirements

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

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

### Overview

#### IN THIS SECTION

- [Topology | 420](#)

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

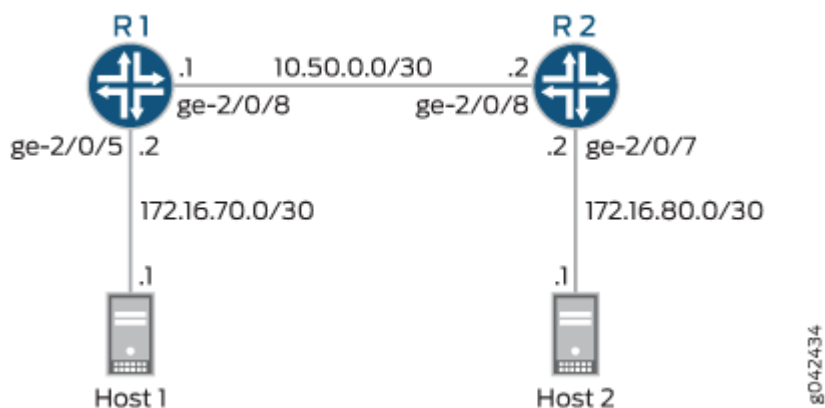


information in Device R1 (making changes for the interfaces used) and apply it to Device R2 to achieve port shaping on Device R2.

## Topology

This example uses the topology in [Figure 34 on page 420](#).

**Figure 34: Port Shaping Scenario**



## Configuration

### IN THIS SECTION

- [Procedure | 420](#)

## Procedure

### CLI Quick Configuration

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



### Device R1 Using Only Class of Service

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description 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 [Junos OS CLI User Guide](#).

You can configure port shaping on network interfaces, aggregated Ethernet interfaces (also known as link aggregation groups (LAGs)), and loopback interfaces.

To configure Device R1:

1. Configure the device interfaces.

```
[edit]
user@R1# set interfaces ge-2/0/5 description to-Host
user@R1# set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set interfaces ge-2/0/8 description to-R2
user@R1# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set interfaces lo0 unit 0 description 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

### IN THIS SECTION

- [Clearing the Counters | 426](#)
- [Sending TCP Traffic into the Network and Monitoring the Port Shaping | 427](#)

Confirm that the configuration is working properly.

### Clearing the Counters

#### Purpose

Confirm that the interface counters are cleared.



## Action

On Device R1, run the `clear interfaces statistics ge-2/0/8` command to reset the interface statistics to 0.

```
user@R1> clear interfaces statistics ge-2/0/8
```

## Sending TCP Traffic into the Network and Monitoring the Port Shaping

### Purpose

Make sure that the traffic is rate-limited on the output interface (ge-2/0/8) on Device R1 by sending traffic into the network using the host connected to Device R1.

### Action

1. Use a traffic generator to send several continuous streams of TCP packets with a source port of 80.

The `-s` flag sets the source port. The `-k` flag causes the source port to remain steady at 80 instead of incrementing. The `-d` flag sets the packet size. The `-c` flag sets the packet count to be sent.

The destination IP address of 172.16.80.1 represents a user that is downstream of Device R2. The user has requested a webpage from the host (the webserver emulated by the traffic generator), and the packets are sent in response to the request.



**NOTE:** Remember in this example the port shaping has been set to 160 Kbps.

```
[user@host]# hping 172.16.80.1 -s 80 -k -d 1500 -c 20 &
hping 172.16.80.1 -s 80 -k -d 1500 -c 20 &
.
.
.
```

2. On Device R1, check the interface counters by using the `show interfaces extensive ge-2/0/8` command.

```
user@R1> show interfaces extensive ge-2/0/8
```

Queue counters:	Queued packets	Transmitted packets	Dropped packets
0	17244	3741	<b>13470</b>
1	13	13	0
2	0	0	0



3	149363	149363	0
Queue number:	Mapped forwarding classes		
0	best-effort		
1	expedited-forwarding		
2	assured-forwarding		
3	network-control		

## Meaning

In the output you can see that 13470 packets have been dropped. These are the packets that exceeded the 160 Kbps shaping rate configured on ge-2/0/8.

## RELATED DOCUMENTATION

[Routing Policies, Firewall Filters, and Traffic Policers User Guide](#)

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

## Configuring Input Shaping Rates for Both Physical and Logical Interfaces

You can apply input shaping rates to both the physical interface and its logical interfaces. The rate specified at the physical level is distributed among the logical interfaces based on their input shaping-rate ratio.

To configure an input shaper on the physical interface:

- Specify the physical interface and associated shaping rate.

```
[edit]
user@host# edit class-of-service interfaces interface-name
user@host# set input-shaping-rate rate
```

To configure an input shaper on the logical interface:



- Specify the physical and logical interface names and associated shaping rate.

```
[edit]
user@host# edit class-of-service interfaces interface-name unit logical-unit-number
user@host# set input-shaping-rate (percent percentage | rate)
```

For each logical interface, you can specify a percentage of the physical rate or an actual rate. The Junos OS software converts actual rates into percentages of the physical rate.

## RELATED DOCUMENTATION

| [Configuring Shaping on 10-Gigabit Ethernet IQ2 PICs](#) | 1040



# Setting Transmission Order with Scheduler Priorities and Hierarchical Scheduling

## IN THIS CHAPTER

- [Priority Scheduling Overview | 430](#)
- [Configuring Schedulers for Priority Scheduling | 433](#)
- [Associating Schedulers with Fabric Priorities | 434](#)
- [Hierarchical Class of Service Overview | 436](#)
- [Hierarchical Class of Service Network Scenarios | 439](#)
- [Understanding Hierarchical Scheduling | 441](#)
- [Priority Propagation in Hierarchical Scheduling | 444](#)
- [Hierarchical CoS for Metro Ethernet Environments | 448](#)
- [Hierarchical Schedulers and Traffic Control Profiles | 449](#)
- [Example: Building a Four-Level Hierarchy of Schedulers | 451](#)
- [Hierarchical Class of Service for Network Slicing | 457](#)
- [Configuring Ingress Hierarchical CoS | 461](#)

## Priority Scheduling Overview

### IN THIS SECTION

- [Strict-High Priority Configuration Overview | 432](#)

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



Priority scheduling determines the order in which an output interface transmits traffic from its queues, thus ensuring that queues containing important traffic have better access to the outgoing interface. Junos accomplishes priority scheduling by examining the assigned priority of each individual queue and whether each individual queue is within its defined bandwidth profile. Junos determines whether an individual queue is within its bandwidth profile by comparing, at regular intervals, the amount of data transmitted by the queue against the amount of bandwidth allocated to it by the configured scheduler transmission rate (`transmit-rate`) defined at the `[edit class-of-service schedulers scheduler-name]` hierarchy level. When the transmitted amount is less than the allocated amount, the queue is considered to be *in profile*. A queue is *out of profile* when its transmitted amount is larger than its allocated amount.

The queues for a given output physical or logical interface are divided into sets based on their priority. Any such set contains queues of the same priority.

Junos traverses the sets in descending order of priority. If at least one of the queues in the set has a packet to transmit, the software selects that set. A queue from the set is selected based on the weighted round robin (WRR) algorithm, which operates within the set.

Junos performs priority queuing using the following steps:

1. The software locates all high-priority queues that are currently in profile. These queues are serviced first in a weighted round-robin fashion.
2. The software locates all medium-high priority queues that are currently in profile. These queues are serviced second in a weighted round-robin fashion.
3. The software locates all medium-low priority queues that are currently in profile. These queues are serviced third in a weighted round-robin fashion.
4. The software locates all low-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](#) | 332



## 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.

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 | 430](#)

[How Schedulers Define Output Queue Properties | 332](#)

## Associating Schedulers with Fabric Priorities

### IN THIS SECTION

- [Example: Associating a Scheduler with a Fabric Priority | 435](#)



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](#) | 308

## 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, slice, and physical port.

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

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

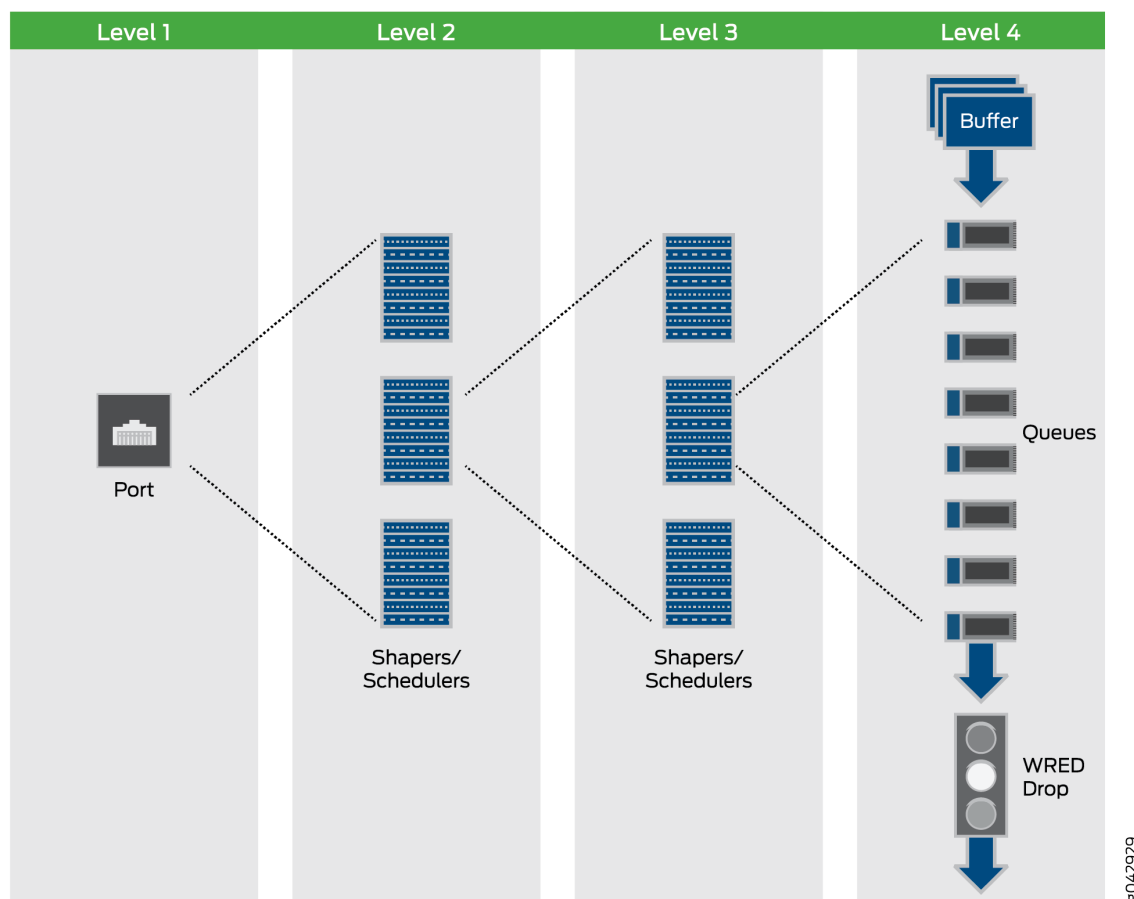
A classifier is a filter that labels traffic at the device ingress based on configurable parameters such as application or destination. Traffic is classified into what is called a forwarding equivalence class (FEC). The FEC defines a class of traffic that receives common treatment.

Schedulers, and their associated shapers, are the functions that control the traffic bandwidth, *jitter* (delay variation), and packet loss priority at the egress of the device.

Hierarchical schedulers are used to apply multiple levels of scheduling and shaping with each level applied to different classifications such as forwarding equivalence class, VLAN, and physical interface (port) as shown in [Figure 35 on page 437](#).



Figure 35: Hierarchical Scheduling Architecture



**NOTE:** Hierarchical class of service is also referred to as Hierarchical Quality of Service (HQoS) in other vendor's documentation.

A typical application of HCoS is to configure multiple levels of egress schedulers and shapers, at the subscriber edge, using dynamic profiles to provide traffic shaping and prioritization at the subscriber VLAN level and for multiple classes of traffic.

Dynamic profiles are a mechanism that allows you to dynamically apply schedulers and shapers to individual subscribers or groups of subscribers.

To learn more about HCoS, the following topics are very helpful:

- ["Junos CoS on MX Series 5G Universal Routing Platforms Overview" on page 753](#)
- ["CoS Features and Limitations on MX Series Routers" on page 754](#)
- ["CoS Features of the Router Hardware, PIC, MIC, and MPC Interface Families" on page 871](#)



- ["How Schedulers Define Output Queue Properties" on page 332](#)

- *Subscriber Access Network Overview*
- *CoS for Subscriber Access Overview*
- *Hierarchical Class of Service for Subscriber Management Overview*

The Junos OS hierarchical schedulers support up to five levels of scheduler hierarchies on MX Series devices when using enhanced queuing Dense Port Concentrators (DPCs) or fine-grained queuing Modular Port Concentrators (MPCs), and Modular Interface Cards (MICs). It is important to know the capabilities of your hardware with respect to HCoS. The following are a few tips to help you:

- Only certain hardware supports the five-level scheduler hierarchy of HCoS.
- The number of queues and logical interfaces supported is dependent upon exactly what hardware you are using.
- The MX Series Packet Forwarding Engine handles guaranteed bandwidth and scheduler node weight differently than other Packet Forwarding Engines.
- The fine-grained queuing MPCs and MICs have a certain granularity with respect to the shaping and delay buffer values. The values used are not necessarily exactly the values configured.

To learn more about platform support for HCoS, use the Juniper Networks Feature Explorer (<https://pathfinder.juniper.net/feature-explorer/>). In the Feature Explorer, search on *hierarchical schedulers*.

In addition, it is important to note the following:

- HCoS is most frequently used to enforce service level agreements at the subscriber edged using dynamic traffic control profiles.
- Hierarchical schedulers can also be applied to Ethernet pseudowire interfaces, aggregated Ethernet interfaces, Layer 2 Tunnel Protocol (L2TP) network server (LNS) inline services, and GRE tunnels.
- Hierarchical ingress policing is a feature that is complimentary to and often used in conjunction with HCoS.
- There are other features in Junos OS that have similar sounding names.



**NOTE:** The *hierarchical scheduler and shaper* feature supported on the SRX Series Firewalls is not the HCoS feature described here.

Before planning HCoS for you network, you should learn about HCoS, define you needs, plan how you want to implement HCoS, and test the operation in a simulated environment.



**Table 40: Resources for Learning More About HCoS**

Document	Description
<a href="#">Day One: Deploying Basic QoS</a> 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.
<a href="#">Day One: Dynamic Subscriber Management</a> Juniper Networks Books	Learn how to use HCoS in conjunction with dynamic traffic control profiles for subscriber management. This book also includes troubleshooting.
QoS Enabled Networks John Wiley & Sons	This book is an additional source for studying QoS.

Documentation related to HCoS is consolidated in the [Hierarchical Class of Service User Guide](#).

## RELATED DOCUMENTATION

*Hierarchical Class of Service for Subscriber Management Overview*

[Hierarchical Class of Service Network Scenarios](#) | 439

[Understanding Hierarchical Scheduling](#) | 441

## Hierarchical Class of Service Network Scenarios

### IN THIS SECTION

- [Services to Subscribers](#) | 440
- [Services to Businesses](#) | 440
- [Wireless Backhaul](#) | 441



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 | 436](#)

*Hierarchical Class of Service for Subscriber Management Overview*

## Understanding Hierarchical Scheduling

### IN THIS SECTION

- [Hierarchical Scheduling Terminology | 442](#)
- [Scheduler Node-Level Designations in Hierarchical Scheduling | 442](#)
- [Hierarchical Scheduling at Non-Leaf Nodes | 444](#)

Hierarchical class of service (HCoS) is a set of capabilities that enable you to apply unique CoS treatment for network traffic based on criteria such as user, application, VLAN, and physical port.

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

This topic covers the following information:



## Hierarchical Scheduling Terminology

Hierarchical scheduling introduces some new CoS terms and also uses some familiar terms in different contexts:

- **Customer VLAN (C-VLAN)**—A C-VLAN, defined by IEEE 802.1ad. A stacked VLAN contains an outer tag corresponding to the S-VLAN, and an inner tag corresponding to the C-VLAN. A C-VLAN often corresponds to CPE. Scheduling and shaping is often used on a C-VLAN to establish minimum and maximum bandwidth limits for a customer. See also *S-VLAN*.
- **Interface set**—A logical group of interfaces that describe the characteristics of set of service VLANs, logical interfaces, customer VLANs, or aggregated Ethernet interfaces. Interface sets establish the set and name the traffic control profiles. See also *Service VLAN*.
- **Scheduler**— A scheduler defines the scheduling and queuing characteristics of a queue. Transmit rate, scheduler priority, and buffer size can be specified. In addition, a drop profile may be referenced to describe WRED congestion control aspects of the queue. See also *Scheduler map*.
- **Scheduler map**—A scheduler map is referenced by traffic control profiles to define queues. The scheduler map establishes the queues that comprise a scheduler node and associates a forwarding class with a scheduler. See also *Scheduler*.
- **Stacked VLAN**—An encapsulation on an S-VLAN with an outer tag corresponding to the S-VLAN, and an inner tag corresponding to the C-VLAN. See also *Service VLAN* and *Customer VLAN*.
- **Service VLAN (S-VLAN)**—An S-VLAN, defined by IEEE 802.1ad, often corresponds to a network aggregation device such as a DSLAM. Scheduling and shaping is often established for an S-VLAN to provide CoS for downstream devices with little buffering and simple schedulers. See also *Customer VLAN*.
- **Traffic control profile**—Defines the characteristics of a scheduler node. Traffic control profiles are used at several levels of the CLI, including the physical interface, interface set, and *logical interface* levels. Scheduling and queuing characteristics can be defined for the scheduler node using the *shaping-rate*, *guaranteed-rate*, and *delay-buffer-rate* statements. Queues over these scheduler nodes are defined by referencing a scheduler map. See also *Scheduler* and *Scheduler map*.
- **VLAN**—Virtual LAN, defined on an Ethernet logical interface.

## Scheduler Node-Level Designations in Hierarchical Scheduling

Scheduler hierarchies are composed of nodes and queues. Queues terminate the hierarchy. Nodes can be either root nodes, leaf nodes, or internal (non-leaf) nodes. Internal nodes are nodes that have other nodes as “children” in the hierarchy.



Scheduler hierarchies consist of levels, starting with Level 1 at the physical port. This chapter establishes a four-level scheduler hierarchy which, when fully configured, consists of the physical interface (Level 1), the interface set (Level 2), one or more logical interfaces (Level 3), and one or more queues (Level 4).



**NOTE:** 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 41 on page 443 describes the possible combinations of scheduler nodes and their corresponding node level designations for a hierarchical queuing MIC or MPC.

**Table 41: 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.

### Change History Table

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

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

## Priority Propagation in Hierarchical Scheduling

Priority propagation is useful for mixed traffic environments when, for example, you want to make sure that the voice traffic of one customer does not suffer due to the data traffic of another customer. Nodes and queues are serviced in the order of their priority. The default priority of a queue is low, and you can explicitly configure a queue priority by including the `priority` statement at the `[edit class-of-service schedulers scheduler-name]` hierarchy level.

You cannot directly configure the priorities of all hierarchical scheduling elements. The priorities of internal nodes, for example, are determined as follows:

- The highest priority of an active child, that is, a child currently containing traffic. (Interface sets only take the highest priority of their active children.)
- Whether the node is above its configured guaranteed rate (CIR) or not (this is only relevant if the physical interface is in CIR mode).



Each queue has a configured priority and a hardware priority. The usual mapping between the configured priority and the hardware priority is shown in [Table 42 on page 445](#).

**Table 42: 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 43 on page 445](#).

**Table 43: 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



**Table 43: Internal Node Queue Priority for CIR Mode (Continued)**

Configured Priority of Highest Active Child Node	Hardware Priority Below Guaranteed Rate	Hardware Priority Above Guaranteed Rate
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 44 on page 446](#).

**Table 44: Internal Node Queue Priority for PIR-Only Mode**

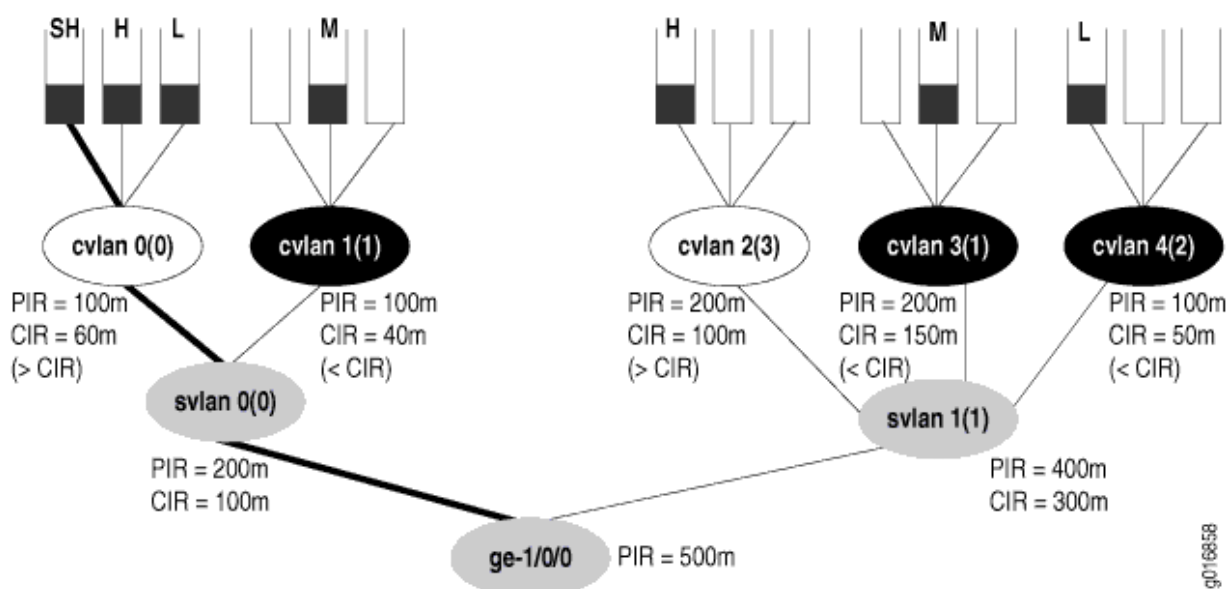
Configured Priority	Hardware Priority
Strict-high	0
High	0
Medium-high	1
Medium-low	1
Low	2

A physical interface with hierarchical schedulers configured is shown in [Figure 36 on page 447](#). 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 36: 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 43 on page 445](#) 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 | 1005](#)

[Enhanced Queuing DPC CoS Properties | 1071](#)

[CoS Features and Limitations on MIC and MPC Interfaces | 1095](#)

[Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 1150](#)



## Hierarchical CoS for Metro Ethernet Environments

In metro Ethernet environments, a virtual LAN (VLAN) typically corresponds to a customer premises equipment (CPE) device and the VLANs are identified by an inner VLAN tag on Ethernet frames (called the customer VLAN, or C-VLAN, tag). A set of VLANs can be grouped at the DSL access multiplexer (DSLAM) and identified by using the same outer VLAN tag (called the service VLAN, or S-VLAN, tag). The service VLANs are typically gathered at the Broadband Remote Access Server (B-RAS) level. Hierarchical schedulers let you provide shaping and scheduling at the service VLAN level as well as other levels, such as the physical interface. In other words, you can group a set of logical interfaces and then apply scheduling and shaping parameters to the logical interface set as well as to other levels.

You can apply CoS shaping and scheduling at one of four different levels, including the VLAN set level. (Some devices support up to five levels of scheduler hierarchies.)

The supported scheduler hierarchy is as follows:

- The physical interface (level 1)
- The service VLAN (level 2)
- The logical interface or customer VLAN (level 3)
- The queue (level 4)

Users can specify a traffic control profile (`output-traffic-control-profile`) that can specify a shaping rate, a guaranteed rate, and a scheduler map with transmit rate and buffer delay. The scheduler map contains the mapping of queues (forwarding classes) to their respective schedulers (schedulers define the properties for the queue). Queue properties can specify a transmit rate and buffer management parameters such as buffer size and drop profile.

To configure CoS hierarchical scheduling, you must enable hierarchical scheduling by including the `hierarchical-scheduler` statement at the physical interface.

### RELATED DOCUMENTATION

---

[Understanding Hierarchical Scheduling | 441](#)

---

[Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 1150](#)

---

[CoS on Enhanced IQ2 PICs Overview | 1005](#)

---

*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.

Change History Table

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

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

RELATED DOCUMENTATION



---

[Providing a Guaranteed Minimum Rate | 372](#)

---

[Configuring Scheduler Maps | 338](#)

---

[Configuring Traffic Control Profiles for Shared Scheduling and Shaping | 340](#)

## Example: Building a Four-Level Hierarchy of Schedulers

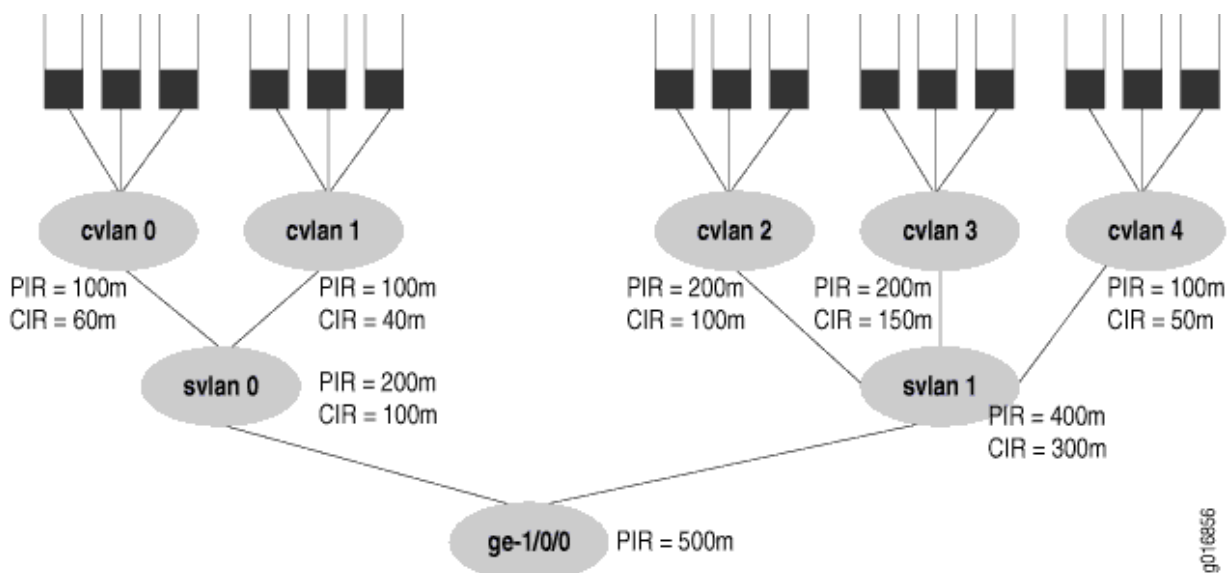
### IN THIS SECTION

- [Configuring the Interface Sets | 452](#)
- [Configuring the Interfaces | 453](#)
- [Configuring the Traffic Control Profiles | 453](#)
- [Configuring the Schedulers | 454](#)
- [Configuring the Drop Profiles | 455](#)
- [Configuring the Scheduler Maps | 456](#)
- [Applying the Traffic Control Profiles | 456](#)

This section provides a more complete example of building a 4-level hierarchy of schedulers. The configuration parameters are shown in [Figure 37 on page 452](#). The queues are shown at the top of the figure with the other three levels of the hierarchy below.



Figure 37: Building a Scheduler Hierarchy



The figure's PIR values are configured as the shaping rates and the CIRs are configured as the guaranteed rate on the Ethernet interface `ge-1/0/0`. The PIR can be oversubscribed (that is, the sum of the children PIRs can exceed the parent's, as in `svlan 1`, where  $200 + 200 + 100$  exceeds the parent rate of 400). However, the sum of the children node level's CIRs must never exceed the parent node's CIR, as shown in all the service VLANs (otherwise, the guaranteed rate could never be provided in all cases).

This configuration example presents all details of the CoS configuration for the interface in the figure (`ge-1/0/0`), including:

### Configuring the Interface Sets

```
[edit interfaces]
interface-set svlan-0 {
  interface ge-1/0/0 {
    unit 0;
    unit 1;
  }
}
interface-set svlan-1 {
  interface ge-1/0/0 {
    unit 2;
    unit 3;
    unit 4;
```



```

    }
}

```

## Configuring the Interfaces

The keyword to configure hierarchical schedulers is at the physical interface level, as is VLAN tagging and the VLAN IDs. In this example, the interface sets are defined by logical interfaces (units) and not outer VLAN tags. All VLAN tags in this example are customer VLAN tags.

```

[edit interface ge-1/0/0]
hierarchical-scheduler;
vlan-tagging;
unit 0 {
    vlan-id 100;
}
unit 1 {
    vlan-id 101;
}
unit 2 {
    vlan-id 102;
}
unit 3 {
    vlan-id 103;
}
unit 4 {
    vlan-id 104;
}

```

## Configuring the Traffic Control Profiles

The traffic control profiles hold parameters for levels above the queue level of the scheduler hierarchy. This section defines traffic control profiles for both the service VLAN level (logical interfaces) and the customer VLAN (VLAN tag) level.

```

[edit class-of-service traffic-control-profiles]
tcp-500m-shaping-rate {
    shaping-rate 500m;
}
tcp-svlan0 {
    shaping-rate 200m;
}

```



```

    guaranteed-rate 100m;
    delay-buffer-rate 300m; # This parameter is not shown in the figure.
}
tcp-svlan1 {
    shaping-rate 400m;
    guaranteed-rate 300m;
    delay-buffer-rate 100m; # This parameter is not shown in the figure.
}
tcp-cvlan0 {
    shaping-rate 100m;
    guaranteed-rate 60m;
    scheduler-map tcp-map-cvlan0; # Applies scheduler maps to customer VLANs.
}
tcp-cvlan1 {
    shaping-rate 100m;
    guaranteed-rate 40m;
    scheduler-map tcp-map-cvlan1; # Applies scheduler maps to customer VLANs.
}
tcp-cvlan2 {
    shaping-rate 200m;
    guaranteed-rate 100m;
    scheduler-map tcp-map-cvlanx; # Applies scheduler maps to customer VLANs.
}
tcp-cvlan3 {
    shaping-rate 200m;
    guaranteed-rate 150m;
    scheduler-map tcp-map-cvlanx; # Applies scheduler maps to customer VLANs
}
tcp-cvlan4 {
    shaping-rate 100m;
    guaranteed-rate 50m;
    scheduler-map tcp-map-cvlanx; # Applies scheduler maps to customer VLANs
}

```

## Configuring the Schedulers

The schedulers hold the information about the queues, the last level of the hierarchy. Note the consistent naming schemes applied to repetitive elements in all parts of this example.

```

[edit class-of-service schedulers]
sched-cvlan0-qx {

```



```

    priority low;
    transmit-rate 20m;
    buffer-size temporal 100ms;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}
sched-cvlan1-q0 {
    priority high;
    transmit-rate 20m;
    buffer-size percent 40;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}
sched-cvlanx-qx {
    transmit-rate percent 30;
    buffer-size percent 30;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}
sched-cvlan1-qx {
    transmit-rate 10m;
    buffer-size temporal 100ms;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}

```

## Configuring the Drop Profiles

This section configures the drop profiles for the example. For more information about interpolated drop profiles, see ["RED Drop Profiles for Congestion Management" on page 464](#).

```

[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 454](#).

```
[edit class-of-service scheduler-maps]
tcp-map-cvlan0 {
    forwarding-class voice scheduler sched-cvlan0-qx;
    forwarding-class video scheduler sched-cvlan0-qx;
    forwarding-class data scheduler sched-cvlan0-qx;
}
tcp-map-cvlan1 {
    forwarding-class voice scheduler sched-cvlan1-q0;
    forwarding-class video scheduler sched-cvlan1-qx;
    forwarding-class data scheduler sched-cvlan1-qx;
}
tcp-map-cvlanx {
    forwarding-class voice scheduler sched-cvlanx-qx;
    forwarding-class video scheduler sched-cvlanx-qx;
    forwarding-class data scheduler sched-cvlanx-qx;
}
```

## Applying the Traffic Control Profiles

This section applies the traffic control profiles to the proper levels of the hierarchy.



**NOTE:** Although a shaping rate can be applied directly to the physical interface, hierarchical schedulers must use a traffic control profile to hold this parameter.

```
[edit class-of-service interfaces]
ge-1/0/0 {
    output-traffic-control-profile tcp-500m-shaping-rate;
    unit 0 {
        output-traffic-control-profile tcp-cvlan0;
    }
    unit 1 {
        output-traffic-control-profile tcp-cvlan1;
    }
    unit 2 {
```



```

        output-traffic-control-profile tcp-cvlan2;
    }
    unit 3 {
        output-traffic-control-profile tcp-cvlan3;
    }
    unit 4 {
        output-traffic-control-profile tcp-cvlan4;
    }
}
interface-set svlan0 {
    output-traffic-control-profile tcp-svlan0;
}
interface-set svlan1 {
    output-traffic-control-profile tcp-svlan1;
}

```



**NOTE:** You should be careful when using a `show interfaces queue` command that references nonexistent class-of-service logical interfaces. When multiple logical interfaces (units) are not configured under the same interface set or physical interface, but are referenced by a command such as `show interfaces queue ge-10/0/1.12 forwarding-class be` or `show interfaces queue ge-10/0/1.13 forwarding-class be` (where logical units 12 and 13 are not configured as a class-of-service interfaces), these interfaces display the same traffic statistics for each logical interface. In other words, even if there is no traffic passing through a particular unconfigured logical interface, as long as one or more of the other unconfigured logical interfaces under the same interface set or physical interface is passing traffic, this particular logical interface displays statistics counters showing the total amount of traffic passed through all other unconfigured logical interfaces together.

## Hierarchical Class of Service for Network Slicing

### IN THIS SECTION

- [Understanding Network Slicing | 458](#)
- [Workflow for Creating Slices | 459](#)
- [Hierarchical Class of Service \(CoS\) Queuing for Slice Per-hop-behavior | 461](#)



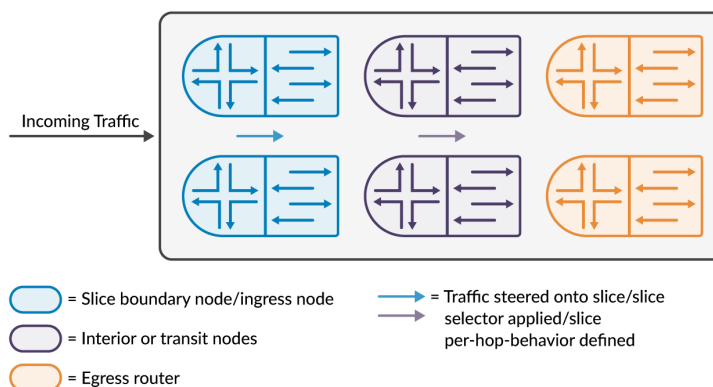
## Understanding Network Slicing

Network slicing is the partitioning of a physical network into multiple logical networks. Each logical network is called a slice. On virtue of being a logical network, a slice is a designated set of network resources, such as interfaces, firewall filters, policers, virtual output queues, schedulers, shapers, traffic control profiles etc. to carry traffic.

### Slice Domain

A set of connected physical nodes such as routers and switches (along with their links) that participate in network slicing is called a slice domain. The slice domain has ingress nodes, transit nodes, and egress nodes. Ingress and egress nodes are located at the borders of the slice domain. The ingress nodes receive traffic into the domain and may classify them before forwarding them to the transit nodes. The egress nodes forward traffic out of the slice domain, and before doing so, may classify the packets.

**Figure 38: Slice Domain**

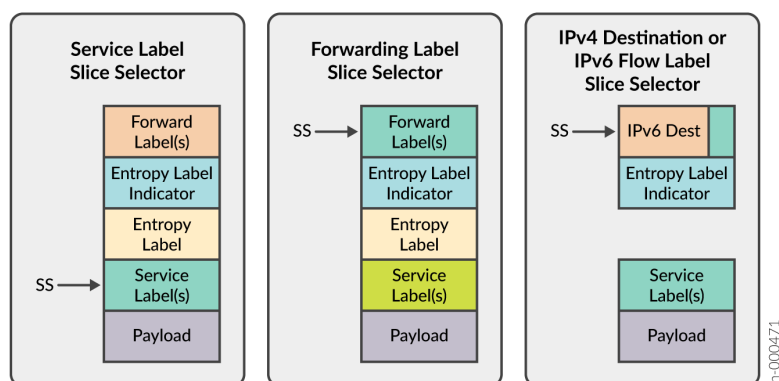


### Slice Selector

By definition, a slice selector is information in the packet header. The information is used by the boundary nodes and/or transit nodes of a slice domain to classify and/or process packets. There are various options to encapsulate/identify/designate a slice selector in the packet. As an example, a Service Label in the packet header can be used as a slice selector. If defined, this label is allotted a position in the packet header and is checked at this position by firewall filters to determine/designate slices. Similarly, there are several other options as depicted in the following figure.



Figure 39: Slice Selector Types



## Workflow for Creating Slices

Slices as entities are created by specifying them under network slicing hierarchy under services. Then these slices are used to steer packets and to manage traffic destined to slices.

### Hierarchical class of service for slices

You define a traffic control profile for a slice under a physical or aggregated Ethernet interface. Note that you can define traffic control profiles for multiple slices under a physical or aggregated Ethernet interface. See *slice (CoS Interfaces)*.

See "[Hierarchical Class of Service \(CoS\) Queuing for Slice Per-hop-behavior](#)" on page 461 to understand how slices (as part of a hierarchy) are used to control traffic.

### Packet steering

Packet steering is the process of marking/matching packets to/from slices. Packets can be steered using firewall filters (firewall steering) and/or routing policy (route steering).

### Firewall steering

- A firewall filter can be used at the ingress node to mark matched packets as belonging to slices using the "slice" action. See *slice (firewall filter action)*.
- A firewall filter can also be used at the transit node to match slice packets using the "slice" match condition. See *slice (firewall filter match condition)*. The packet can then be marked to another slice if required by the firewall filter or a policer applied to this packet etc.
- Packets that are not marked/matched to to/from slices are processed as non-slice traffic.

### Route steering



- An export policy can be used at the ingress and/or transit node to mark matched routes as belonging to slices. See *slice (export routing policy action)*.
- The export policy can also attach a firewall filter to the route. The firewall filter is used to typically apply a policer to the packets matching the route on the ingress side. This firewall filter is not attached to any interface. Rather, it is part of the forwarding information of the route. See *filter (export routing policy action)*.
- The slice and/or firewall filter will be part of the route's next-hop forwarding information. See *show route extensive expanded-nh* to view slices and/or firewall filters attached to routes.
- Packets that do not match routes with slice information, are classified as non-slice traffic. Packets that match routes with no slice information, are also classified as non-slice traffic.

To summarize, the following are the configurations that are to be enabled before creating slices, can be used to create slices, or manage packets belonging to slices.

- Specify the slices under network slicing hierarchy under services- – Refer to *network-slicing*.
- Enable enhanced-ip mode – Refer *network-services*.
- Perform class of service configuration to enable a slice under an interface and also apply an output traffic control profile for the slice – See *slice (CoS Interfaces)*. See "[Hierarchical Class of Service \(CoS\) Queuing for Slice Per-hop-behavior](#)" on page 461 to understand how slices (as part of a hierarchy) are used to control traffic.
- Configure firewall filters to steer routes to slices and/or match routes from slices – See *slice (firewall filter match condition)* and *slice (firewall filter action)*.
- Use routing policy to steer routes to slices and/or attach firewall – See *slice (export routing policy action)* and *filter (export routing policy action)*.
- View slices and/or firewall filters attached to routes. See *show route extensive <route> expanded-nh*.
- View slices attached to the forwarding table. See *show class-of-service forwarding-table slice*.
- Show mapping of traffic control profiles to slices. See *show class-of-service forwarding-table slice mapping*.
- View traffic control profile(s) attached to a slice under an interface. See *show class-of-service slice <slice\_name> interface <interface\_name>*.
- View statistics for a slice. See *show cos help flow queue-stats*.



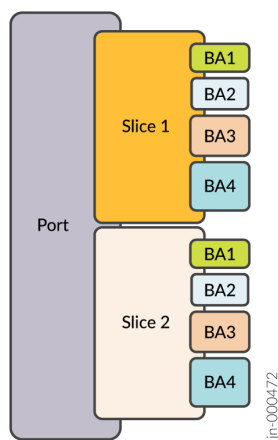
## Hierarchical Class of Service (CoS) Queuing for Slice Per-hop-behavior

In hierarchical CoS, packets are classified at various levels. It could be at the port level, followed by the logical unit level, and then at the queue level. This means that packets are passing through a hierarchy. At every stage of the hierarchy, packets are being classified, shaped, scheduled etc.

In the context of network slicing, a slice also becomes part of the hierarchy. Shapers, schedulers, and traffic control profiles can be applied to the slice. Just as queues are made available to logical units, queues are made available to slices as well.

As the following figure shows, the queues (labeled BA1, BA2, BA3, BA4) are made available to the slice. The queues map to forwarding classes (FCs). Based on behavioral aggregate classifiers, packets are classified into FCs, and subsequently into a corresponding queue (BA1 or BA2 or BA3 etc.).

**Figure 40: Hierarchical Queuing**



As ["Workflow for Creating Slices" on page 459](#) describes, you configure network slicing using a combination of firewall filters and Class of Service (CoS) configuration. See *slice (CoS Interfaces)* to read on how to enable slice(s) under any interface using CoS configuration.

## Configuring Ingress Hierarchical CoS

You can configure ingress CoS parameters, including hierarchical schedulers, on devices with Enhanced Queuing DPCs (that is, line cards that have a queuing chip). In general, the supported configuration statements apply to per-unit schedulers or to hierarchical schedulers.





**NOTE:** Ingress CoS is not supported on line cards that do not contain a queuing chip.

To configure ingress CoS for per-unit schedulers, include the following statements at the [edit class-of-service interfaces *interface-name*] hierarchy level:



**NOTE:** The input-scheduler-map and input-traffic-control-profile statements are mutually exclusive at the same hierarchy level.

```
[edit class-of-service interfaces interface-name]
input-excess-bandwidth-share (proportional value | equal);
input-scheduler-map map-name;
input-shaping-rate rate;
input-traffic-control-profile profile-name shared-instance instance-name;
unit logical-unit-number;
    input-scheduler-map map-name;
    input-shaping-rate (percent percentage | rate);
    input-traffic-control-profile profile-name shared-instance instance-name;
}
```

To configure ingress CoS for hierarchical schedulers, include the interface-set *interface-set-name* statement at the [edit class-of-service interfaces] hierarchy level:

```
[edit class-of-service interfaces]
interface-set interface-set-name {
    input-excess-bandwidth-share (proportional value | equal);
    input-traffic-control-profile profile-name shared-instance instance-name;
    input-traffic-control-profile-remaining profile-name;
    interface interface-name {
        input-excess-bandwidth-share (proportional value | equal);
        input-traffic-control-profile profile-name shared-instance instance-name;
        input-traffic-control-profile-remaining profile-name;
        unit logical-unit-number;
            input-traffic-control-profile profile-name shared-instance instance-name;
        }
    }
}
```



For many platforms, CoS queuing and scheduling are enabled on the egress side but disabled on the ingress side by default. To enable ingress CoS, configure the traffic-manager statement with ingress-and-egress mode:

```
[edit chassis fpc slot-number pic pic-number]
traffic-manager mode ingress-and-egress;
```



**NOTE:** If you enable ingress CoS settings and inline services on the same FPC, the FPC moves to the offline state. This behavior is expected because null route detection is triggered that causes the FPC to move to the offline state.

Configured CoS features on the ingress are independent of CoS features on the egress, with the following exceptions:

- If you configure a per-unit or hierarchical scheduler at the [edit class-of-service interfaces] hierarchy level, the schedulers apply in both the ingress and egress directions.
- You cannot configure the same logical interface on an ingress and an egress interface set. A logical interface can only belong to one interface set.
- The DPC's frame buffer is shared between ingress and egress configurations.

The following behavior aggregate (BA) classification tables are supported on the ingress side:

- inet-precedence
- DSCP
- exp (MPLS)
- DSCP for IPv6
- IEEE 802.1p

## RELATED DOCUMENTATION

[Configuring Traffic Control Profiles for Shared Scheduling and Shaping | 340](#)

[Enhanced Queuing DPC CoS Properties | 1071](#)



# Controlling Congestion with Scheduler RED Drop Profiles, Buffers, PFC, and ECN

## IN THIS CHAPTER

- [RED Drop Profiles for Congestion Management | 464](#)
- [Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers | 471](#)
- [Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows | 473](#)
- [Mapping PLP to RED Drop Profiles | 475](#)
- [Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size | 478](#)
- [Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy | 496](#)
- [Example: Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy | 498](#)
- [Understanding PFC Using DSCP at Layer 3 for Untagged Traffic | 502](#)
- [Configuring DSCP-based PFC for Layer 3 Untagged Traffic | 504](#)
- [PFC Watchdog | 506](#)
- [CoS Explicit Congestion Notification | 511](#)
- [Example: Configuring Static and Dynamic ECN | 522](#)

## RED Drop Profiles for Congestion Management

### SUMMARY

This topic describes the use and configuration of *random early detection (RED)* drop profiles for congestion management. A *drop profile* is a mechanism of RED that defines parameters that allow packets to be dropped from a queue based on how full the queue is. Drop profiles define the meanings of the packet loss priorities.

### IN THIS SECTION

- [Manage Congestion with RED Drop Profiles and Packet Loss Priorities | 465](#)
- [Configure RED Drop Profiles to Define Packet Drop or ECN Behaviors | 467](#)



## Manage Congestion with RED Drop Profiles and Packet Loss Priorities

You can configure two parameters to control congestion in each output queue. One parameter, *delay-buffer bandwidth*, enables queue growth to absorb burst traffic up to the specified product of delay-buffer time and output rate. Once the specified delay buffer becomes full, packets with 100 percent drop probability are dropped from the tail of the queue. For more information, see ["Managing Congestion on the Egress Interface by Configuring the Scheduler Buffer Size" on page 478](#).

The other parameter, which this topic covers, defines the *drop probabilities* across the range of delay-buffer occupancy, supporting the *RED process*. When the number of packets queued is greater than the ability of the router or switch to empty a queue, the queue requires a method for determining which packets to drop from the network. To address this, you can enable RED on individual queues.

Depending on the drop probabilities, RED might drop many packets long before the buffer becomes full, or it might drop only a few packets even if the buffer is almost full.

A *drop profile* is a mechanism of RED that defines parameters that allow packets to be dropped from the network. Drop profiles define the meanings of the packet loss priorities.

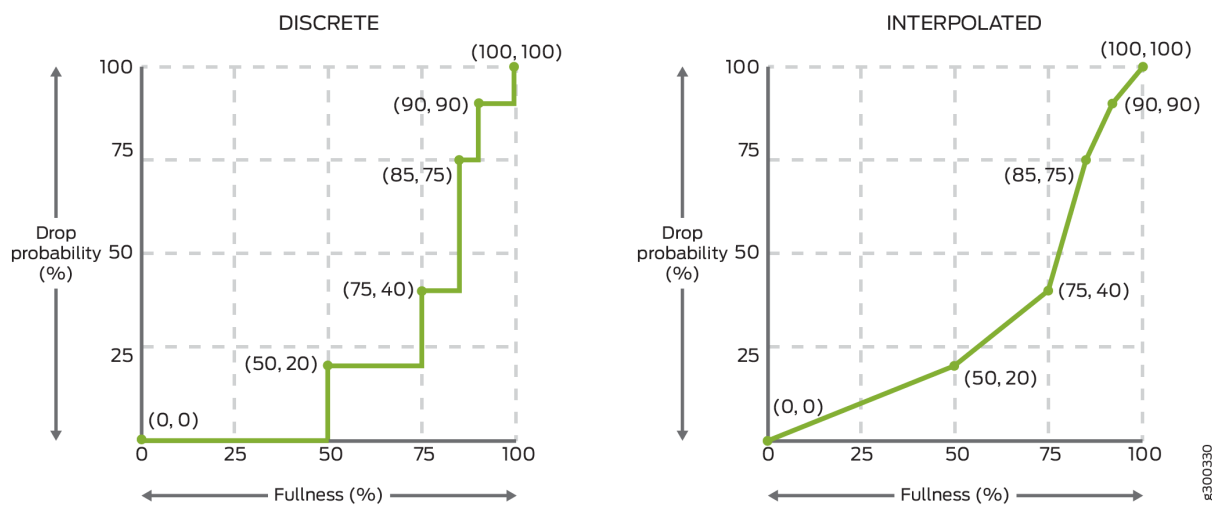
When you configure drop profiles, there are two important values:

- *queue fullness* represents a percentage of the memory used to store packets in relation to the total amount that has been allocated for a specific queue.
- *drop probability* is a percentage value that correlates to the likelihood that an individual packet is dropped from the network.

How these two variables function is illustrated in graph format. [Figure 41 on page 466](#) shows both a discrete and an interpolated graph. Although the formation of these graph lines is different, the application of the profile is the same. When a packet joins the tail of the queue, a random number from 0 to 100 is calculated by the router or switch. This random number is plotted against the drop profile using the current queue fullness of that particular queue. When the random number falls above the graph line, the packet is transmitted onto the physical media. When the number falls below the graph line, the packet is dropped from the network.



Figure 41: Discrete and Interpolated Drop Profiles



You create drop profiles by defining multiple fill levels and drop probabilities.

To create the discrete profile graph as shown in [Figure 41 on page 466](#) on the left, the software begins at the bottom-left corner, representing a 0-percent fill level and a 0-percent drop probability. This configuration creates a line horizontally to the right on the fullness level (l) x-axis until it reaches the first defined fill level, 50-percent for this configuration, which is designated to have a drop probability (p) of 20-percent. The software then continues the line horizontally along the fill level until the next drop probability is reached at the designated data point of 75-percent fill level, which has a designated drop-probability of 40-percent. The line is then continued horizontally to the next fill level of 85-percent and the designated drop probability of 75-percent. The line continues horizontally to the next designated fill level of 90-percent, which has a designated drop probability of 90-percent, and a line is created to data point 90-percent (l), 90-percent (p) (l90 p90). From the l90 p90 point, the line continues horizontally to the 100-percent fill level, which has a drop probability of 100 percent, at which the line rises to the end-point of 100-100, which is 100 percent fill level with a 100 percent drop probability.

If you specify an interpolated drop profile, in the first quadrant the initial line segment spans from the origin (0,0) to the next defined point. From that defined fill-level/drop-probability point, a second line runs to the next point, and so forth, until a final line segment connects (100, 100). The software automatically constructs a drop profile containing 64 fill levels at drop probabilities that approximate the calculated line segments.



**NOTE:** For consistent behavior across router families, include the pair (100, 100) in the drop profile configuration.

You can create a smoother graph line by configuring the profile with the `interpolate` statement. This enables the software to automatically generate 64 data points on the graph beginning at (0, 0) and ending at (100, 100). Along the way, the graph line intersects specific data points that you have defined.





**NOTE:** If you configure the `interpolate` statement, you can specify more than 64 pairs, but the system generates only 64 discrete entries.

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

You specify drop probabilities in the drop profile section of the class-of-service (CoS) configuration hierarchy and map them to corresponding loss priorities in each scheduler configuration. For each scheduler, you can configure multiple separate drop profiles, one for each combination of loss priority.

You can configure a maximum of 32 different drop profiles.

Use [Feature Explorer](#) to confirm platform and release support for specific features.

Review the ["Platform-Specific RED Drop Profile Behavior" on page 469](#) section for notes related to your platform.

## Configure RED Drop Profiles to Define Packet Drop or ECN Behaviors

You enable RED by applying a drop profile to a scheduler. When RED is operational on an interface, the queue no longer drops all excess packets at the tail of the queue. Rather, a controlled fraction of packets are dropped, or marked with ECN (if enabled). Some output-buffered routers perform RED drops of oldest packets at the head of the queue. Other routers perform RED as packets enter a queue. When a queue becomes full, tail-drops (100%) supersede random dropping.

To configure RED drop profiles, include the following statements at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
drop-profiles {
  profile-name {
    fill-level percentage drop-probability percentage;
    interpolate {
      drop-probability [ values ];
      fill-level [ values ];
    }
  }
}
```



To configure a drop profile, include either the `interpolate` statement and its options, or the `fill-level` and `drop-probability` *percentage* values.

For example, the following shows a discrete configuration and an interpolated configuration that correspond to the graphs in [Figure 41 on page 466](#). The values defined in the configurations are matched to represent the data points in the graph lines.

### Create a Discrete Configuration

```
class-of-service {
  drop-profiles {
    discrete-style-profile {
      fill-level 0 drop-probability 0;
      fill-level 50 drop-probability 20;
      fill-level 75 drop-probability 40;
      fill-level 85 drop-probability 75;
      fill-level 90 drop-probability 90;
      fill-level 100 drop-probability 100;
    }
  }
}
```

### Create an Interpolated Configuration

```
class-of-service {
  drop-profiles {
    interpolated-style-profile {
      interpolate {
        fill-level [ 0 50 75 85 90 100 ];
        drop-probability [ 0 20 40 75 90 100 ];
      }
    }
  }
}
```

To configure a drop profile:

1. Create the drop profile by specifying a name for it.

```
[edit]
user@host# edit class-of-service drop-profiles profile-name
```



2. (Optional) Specify the fill-level and drop-probability values for the drop profile.

```
[edit class-of-service drop-profiles profile-name]
user@host# set fill-level percentage drop-probability percentage
```

Repeat this step for each fill-level and drop-probability.

3. (Optional) Specify values for interpolating the relationship between queue fill level and drop probability.

```
[edit class-of-service drop-profiles profile-name]
user@host# set interpolate drop-probability percentage drop-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
```

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 471.

## Platform-Specific RED Drop Profile Behavior

Use [Feature Explorer](#) to confirm platform and release support for RED drop profiles.

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



Platform	Difference
ACX5448	<ul style="list-style-type: none"> <li>• The ACX5448 router does not support interpolate drop-probability and supports only the discrete method.</li> <li>• The ACX5448 router supports configuring drop profiles for loss-priority <i>low</i>, <i>medium-high</i> and <i>high</i>.</li> <li>• You can specify two fill levels in each drop profile map. The drop probability you associate with the lowest fill level must be 0.</li> </ul>
ACX7000 Series	<ul style="list-style-type: none"> <li>• ACX7000 Series routers do not support interpolate drop-probability and support only the discrete method.</li> <li>• ACX7000 Series routers support configuring drop profiles for loss-priority <i>low</i>, <i>medium-high</i> and <i>high</i>.</li> <li>• You can specify two fill levels in each drop profile map. The drop probability you associate with the lowest fill level must be 0.</li> </ul>
MX Series	<ul style="list-style-type: none"> <li>• 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.</li> <li>• 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 <a href="#">"Configuring WRED on Enhanced Queuing DPCs" on page 1076</a>.</li> <li>• 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.</li> </ul>



*(Continued)*

Platform	Difference
PTX Series	<ul style="list-style-type: none"> <li>PTX Series routers running Junos OS and PTX10003 routers running Junos OS Evolved support up to 64 (fill level, drop probability) pairs per discrete or interpolated drop profile.</li> <li>Other PTX Series routers running Junos OS Evolved support up to two (fill level, drop probability) pairs per interpolated drop profile. A single-pair drop profile is discrete. Two pairs are interpolated.</li> </ul>

## RELATED DOCUMENTATION

*drop-probability (Interpolated Value)*

*drop-probability (Percentage)*

## Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers

RED drop profiles take action on outgoing packets. When tricolor marking is enabled, 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 ["RED Drop Profiles for Congestion Management" on page 464](#).

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 10](#).



**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

[RED Drop Profiles for Congestion Management](#) | 464

## 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 44](#) and ["Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields" on page 132](#).

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 63](#) and ["Using Multifield Classifiers to Set Packet Loss Priority" on page 137](#).

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 expeditd-forwarding;
    }
}
}

```

##### 5. Save your configuration.

```

[edit firewall]
user@host# commit

```

## RELATED DOCUMENTATION

[Mapping PLP to RED Drop Profiles | 475](#)

[RED Drop Profiles for Congestion Management | 464](#)

[Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers | 471](#)

[Configuring Schedulers | 338](#)

## Mapping PLP to RED Drop Profiles

Loss priority settings help determine which packets are dropped from the network during periods of congestion. The software supports multiple packet loss priority (PLP) designations: *low* and *high*. (In addition, *medium-low* and *medium-high* PLPs are supported when you configure tricolor marking.) You can set PLP by configuring a behavior aggregate or multifield classifier.

A drop-profile map examines the loss priority setting of an outgoing packet: *high*, *medium-high*, *medium-low*, *low*, or *any*.

Obviously, *low*, *medium-low*, *medium-high*, and *high* are relative terms, which by themselves have no meaning. Drop profiles define the meanings of the loss priorities. In the following example, the *low-drop* drop profile defines the meaning of *low* PLP as a 10 percent drop probability when the fill level is 75 percent and a 40 percent drop probability when the fill level is 95 percent. The *high-drop* drop profile defines the meaning of *high* PLP as a 50 percent drop probability when the fill level is 25 percent and a 90 percent drop probability when the fill level is 50 percent.



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

1. Create the low drop profile.

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

2. Specify values for interpolating the relationship between the queue fill level and drop probability for the low drop profile.

```
[edit class-of-service drop-profiles low-drop]
user@host# edit interpolate
user@host# set drop-probability [10 40]
user@host# set fill-level [75 95]
```

3. Create the high drop profile.

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

4. Specify values for interpolating the relationship between the queue fill level and drop probability for the high drop profile.

```
[edit class-of-service drop-profiles high-drop]
user@host# edit interpolate
user@host# set drop-probability [50 90]
user@host# set fill-level [25 50]
```

5. Specify the scheduler name.

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



6. Define the loss-priority for each low drop profile.

```
[edit class-of-service schedulers best-effort]
user@host# set drop-profile-map loss-priority low protocol any drop-profile low-drop
```

7. Define the loss-priority for each high drop profile.

```
[edit class-of-service schedulers best-effort]
user@host# set drop-profile-map loss-priority high protocol any drop-profile high-drop
```

8. Verify your configuration.

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

```
drop-profiles {
  low-drop {
    interpolate {
      fill-level [ 75 95 ];
      drop-probability [ 10 40 ];
    }
  }
  high-drop {
    interpolate {
      fill-level [ 25 50 ];
      drop-probability [ 50 90 ];
    }
  }
}

schedulers {
  best-effort {
    drop-profile-map loss-priority low protocol any drop-profile low-drop;
    drop-profile-map loss-priority high protocol any drop-profile high-drop;
  }
}
```



## 9. Save your configuration.

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

## RELATED DOCUMENTATION

[Managing Congestion by Setting Packet Loss Priority for Different Traffic Flows | 473](#)

[RED Drop Profiles for Congestion Management | 464](#)

[Determining Packet Drop Behavior by Configuring Drop Profile Maps for Schedulers | 471](#)

[Configuring Schedulers | 338](#)

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

## IN THIS SECTION

- [Configuring Large Delay Buffers for Slower Interfaces | 481](#)
- [Configuring the Maximum Delay Buffer for NxDS0 Interfaces | 486](#)
- [Example: Configuring Large Delay Buffers for Slower Interfaces | 488](#)
- [Example: Configuring the Delay Buffer Value for a Scheduler | 489](#)
- [Example: Configuring the Physical Interface Shaping Rate | 492](#)
- [Complete Configuration | 493](#)
- [Enabling and Disabling the Memory Allocation Dynamic per Queue | 494](#)

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 45 on page 479](#).
- 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 45 on page 479](#). 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 45: 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



**Table 45: Buffer Size Temporal Value Ranges by Routing Device Type (Continued)**

Routing Devices	Temporal Value Ranges
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
<b>With Large Buffer Sizes Enabled</b>	
IQ PICs on all routers	1 through 500,000 microseconds
<b>Gigabit Ethernet IQ VLANs</b>	
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



**Table 45: Buffer Size Temporal Value Ranges by Routing Device Type (Continued)**

Routing Devices	Temporal Value Ranges
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

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 46 on page 481](#) shows some recommended buffer sizes needed to absorb typical burst sizes from various upstream interface types.

**Table 46: 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:

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 47 on page 482](#).

**Table 47: Maximum Delay Buffer with q-pic-large-buffer Enabled by Interface**

Platform, PIC, or Interface Type	Maximum Buffer Size
<b>With Large Buffer Sizes Not Enabled</b>	
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
<b>With Large Buffer Sizes Enabled</b>	
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



**Table 47: Maximum Delay Buffer with q-pic-large-buffer Enabled by Interface (Continued)**

Platform, PIC, or Interface Type	Maximum Buffer Size
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



**Table 47: Maximum Delay Buffer with q-pic-large-buffer Enabled by Interface (Continued)**

Platform, PIC, or Interface Type	Maximum Buffer Size
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 391](#).

For  $\Lambda$ xDS0 interfaces, the larger buffer sizes can be up to 4,000,000 microseconds, depending on the number of DS0 channels in the  $\Lambda$ xDS0 interface. For slower  $\Lambda$ xDS0 interfaces with fewer channels, the delay buffer can be relatively larger than for faster  $\Lambda$ xDS0 interfaces with more channels. This is shown in [Table 49 on page 486](#).

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 48 on page 484](#).

**Table 48: 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> <math>1.5 \text{ Mbps} * 0.3 * 500,000 \text{ microseconds} =</math>  <math>225,000 \text{ bits} = 28,125 \text{ bytes}</math> </p>



Table 48: Delay-Buffer Calculations (Continued)

Delay Buffer Configuration	Formula	Example
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> <math>1.5 \text{ Mbps} * 0.2 * 500,000 \text{ microseconds} = 150,000 \text{ bits} = 18,750 \text{ bytes}</math> </p>
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 &lt; T1 bandwidth, the delay-buffer rate is 1 second.</p> <p>For total bundle bandwidth &gt;= T1 bandwidth, the delay-buffer rate is 200 milliseconds (ms).</p>	



## Configuring the Maximum Delay Buffer for NxDS0 Interfaces

Because *NxDS0* interfaces carry less bandwidth than a T1 or E1 interface, the buffer size on an *NxDS0* interface can be relatively larger, depending on the number of DS0 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 1xDS0 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 49 on page 486](#) shows the delay-buffer calculations for 1xDS0 through 32xDS0 interfaces.

**Table 49: NxDS0 Transmission Rates and Delay Buffers**

Interface Speed	Delay Buffer Size
<b>1xDS0 Through 4xDS0: Maximum Delay Buffer Time Is 4,000,000 Microseconds</b>	
1xDS0: 64 Kbps	32 KB
2xDS0: 128 Kbps	64 KB
3xDS0: 192 Kbps	96 KB
<b>4xDS0 Through 7xDS0: Maximum Delay Buffer Time Is 2,000,000 Microseconds</b>	
4xDS0: 256 Kbps	64 KB
5xDS0: 320 Kbps	80 KB
6xDS0: 384 Kbps	96 KB
7xDS0: 448 Kbps	112 KB



**Table 49: NxDS0 Transmission Rates and Delay Buffers (Continued)**

Interface Speed	Delay Buffer Size
<b>8xDS0 Through 15xDS0: Maximum Delay Buffer Time Is 1,000,000 Microseconds</b>	
8xDS0: 512 Kbps	64 KB
9xDS0: 576 Kbps	72 KB
10xDS0: 640 Kbps	80 KB
11xDS0: 704 Kbps	88 KB
12xDS0: 768 Kbps	96 KB
13xDS0: 832 Kbps	104 KB
14xDS0: 896 Kbps	112 KB
15xDS0: 960 Kbps	120 KB
<b>16xDS0 Through 32xDS0: Maximum Delay Buffer Time Is 500,000 Microseconds</b>	
16xDS0: 1024 Kbps	64 KB
17xDS0: 1088 Kbps	68 KB
18xDS0: 1152 Kbps	72 KB
19xDS0: 1216 Kbps	76 KB
20xDS0: 1280 Kbps	80 KB
21xDS0: 1344 Kbps	84 KB



**Table 49: NxDS0 Transmission Rates and Delay Buffers (Continued)**

Interface Speed	Delay Buffer Size
22xDS0: 1408 Kbps	88 KB
23xDS0: 1472 Kbps	92 KB
24xDS0: 1536 Kbps	96 KB
25xDS0: 1600 Kbps	100 KB
26xDS0: 1664 Kbps	104 KB
27xDS0: 1728 Kbps	108 KB
28xDS0: 1792 Kbps	112 KB
29xDS0: 1856 Kbps	116 KB
30xDS0: 1920 Kbps	120 KB
31xDS0: 1984 Kbps	124 KB
32xDS0: 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 48 on page 484](#).

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:

```
available interface bandwidth * configured percentage buffer-size * maximum buffer = queue buffer

1.5 Mbps * 0.2 * 500,000 microseconds = 150,000 bits = 18,750 bytes
```

Assuming that the sched-exped scheduler is assigned to 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:

```
available interface bandwidth * configured percentage transmit-rate
* configured temporal buffer-size = queue buffer

1.5 Mbps * 0.2 * 300,000 microseconds = 90,000 bits = 11,250 bytes
```

To configure this example:

1. Configure the sched-best scheduler.

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

2. Specify the transmit-rate of 10 percent.

```
[edit class-of-service schedulers sched-best]
user@host# set transmit-rate percent 10
```

3. Specify the buffer size as 20 percent.

```
[edit class-of-service schedulers sched-best]
user@host# set buffer-size percent 20
```



4. Configure the sched-exped scheduler.

```
[edit]
user@host# up
[edit class-of-service schedulers]
user@host# edit sched-exped
```

5. Specify the transmit-rate of 20 percent.

```
[edit class-of-service schedulers sched-exped]
user@host# set transmit-rate percent 20
```

6. Specify the buffer size temporal value (300,000 microseconds).

```
[edit class-of-service schedulers sched-exped]
user@host# set buffer-size temporal 300000
```

7. Verify the configuration.

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

```
schedulers {
  sched-best {
    transmit-rate percent 10;
    buffer-size percent 20;
  }
  sched-exped {
    transmit-rate percent 20;
    buffer-size temporal 300k;
  }
}
```

8. Save the configuration.

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



## Example: 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 49 on page 486](#).

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 4xDSO, 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:



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 | 498](#)

*red-buffer-occupancy*

## Example: Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy

### IN THIS SECTION

- [Requirements | 499](#)
- [Overview | 499](#)
- [Configuration | 499](#)



This topic provides two examples for configuring the weighted RED buffer occupancy feature to manage traffic bursts.

## Requirements

Weighted RED buffer occupancy is configured on a per-PIC basis 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

### IN THIS SECTION

- [CLI Quick Configuration | 500](#)
- [Example: Configuring a Q-PIC to Use a Weight Value of 1/2 in Average Buffer Occupancy Calculations | 500](#)
- [Example: Configuring a Q-PIC to Use a Weight Value of 1/4 in Average Buffer Occupancy Calculations | 500](#)
- [Results | 501](#)



## 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 | 496](#)

*red-buffer-occupancy*



## Understanding PFC Using DSCP at Layer 3 for Untagged Traffic

### IN THIS SECTION

- [Overview of DSCP-based PFC | 502](#)
- [Limitations of DSCP-based PFC | 503](#)

Protocols such as Remote Direct Memory Access (RDMA) over converged Ethernet version 2 (RoCEv2) require lossless behavior for traffic across Layer 3 connections to Layer 2 Ethernet subnetworks. Traditionally, priority-based flow control (PFC) can be used to prevent traffic loss when congestion occurs on Layer 2 or Layer 3 interfaces for VLAN-tagged traffic by selectively pausing traffic on any of eight priorities corresponding to IEEE 802.1p code points in the VLAN headers of incoming traffic on an interface. However, *untagged* traffic—traffic without VLAN tagging—cannot be examined for IEEE 802.1p code points on which to pause traffic.

Starting in Junos OS Release 17.4R1, to support lossless traffic flow at Layer 3 for untagged traffic, we support enabling PFC for Layer 3 interfaces and Layer 2 access interfaces using Distributed Services code point (DSCP) values in the Layer 3 IP header of incoming traffic, rather than IEEE 802.1p code point values in a Layer 2 VLAN header.

### Overview of DSCP-based PFC

PFC is a data center bridging technology operating at Layer 2, and DSCP information is exchanged in IP headers at Layer 3. However, you can configure DSCP-based PFC, which preserves lossless behavior across Layer 3 network connections for untagged traffic.

PFC operates by generating pause frames for traffic identified on configured code points in incoming traffic to notify the peer to pause transmission when the link is congested. With DSCP-based PFC enabled, pause frames are triggered based on a configured 6-bit DSCP value (corresponding to decimal values 0-63) in the Layer 3 IP header of incoming traffic.

However, PFC can only send pause frames with a 3-bit PFC priority—one of 8 code points corresponding to decimal values 0-7—which, for VLAN-tagged traffic, usually corresponds to the IEEE 802.1p code points in the incoming traffic VLAN headers. Untagged traffic provides no reference for IEEE 802.1p code point values, so to trigger PFC on a DSCP value, the DSCP value must be mapped explicitly in the configuration to a PFC priority to use in the PFC pause frames sent to the peer when congestion occurs for that code point. You can map traffic on a DSCP value to a PFC priority when you define the no-loss forwarding class with which you want to classify DSCP-based PFC traffic. The forwarding class must also be mapped to an output queue with no-loss behavior.





**NOTE:** You cannot assign the same PFC priority to more than one forwarding class because the mapped PFC priority value is used as the forwarding class ID when DSCP-based PFC is configured.

A DSCP classifier (instead of an IEEE 802.1p classifier) is also required to specify that incoming traffic with the above-configured DSCP value belongs to the no-loss forwarding class. Any DSCP values for which DSCP-based PFC is enabled on a interface must be specified in either the default DSCP classifier or in a user-defined DSCP classifier associated with the interface.

To enable DSCP-based PFC on an interface, define an input congestion notification profile with the same DSCP value (and desired buffering parameters), and associate it with the interface.

The peer device should have a matching PFC configuration for the mapped PFC priority code points.

**Limitations of DSCP-based PFC**

The following are limitations of DSCP-based PFC:

- You cannot configure both DSCP-based PFC and IEEE 802.1p PFC under the same congestion notification profile, or associate both a DSCP-based congestion notification profile and an IEEE 802.1p congestion notification profile with the same interface.
- DSCP-based PFC is supported on Layer 3 interfaces and Layer 2 access interfaces for untagged traffic only. PFC behavior is unpredictable if VLAN-tagged packets are received on an interface with DSCP-based PFC enabled.
- Each no-loss forwarding class can only be associated with a unique 3-bit PFC priority value from 0 through 7.

**Change History Table**

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

Release	Description
17.4R1	Starting in Junos OS Release 17.4R1, to support lossless traffic flow at Layer 3 for untagged traffic, we support enabling PFC for Layer 3 interfaces and Layer 2 access interfaces using Distributed Services code point (DSCP) values in the Layer 3 IP header of incoming traffic, rather than IEEE 802.1p code point values in a Layer 2 VLAN header.



## RELATED DOCUMENTATION

[Configuring DSCP-based PFC for Layer 3 Untagged Traffic | 504](#)

*Understanding CoS Classifiers*

*Understanding CoS Flow Control (Ethernet PAUSE and PFC)*

*Understanding CoS Forwarding Classes*

*Understanding CoS IEEE 802.1p Priorities for Lossless Traffic Flows*

## Configuring DSCP-based PFC for Layer 3 Untagged Traffic

You can configure DSCP-based PFC to support lossless behavior for untagged traffic across Layer 3 connections to Layer 2 subnetworks for protocols such as Remote Direct Memory Access (RDMA) over converged Ethernet version 2 (RoCEv2).

With DSCP-based PFC, pause frames are generated to notify the peer that the link is congested based on a configured 6-bit Distributed Services code point (DSCP) value in the Layer 3 IP header of incoming traffic, rather than a 3-bit IEEE 802.1p code point in the Layer 2 VLAN header.

Because PFC can only send pause frames corresponding to PFC priority code points, the 6-bit configured DSCP value must be mapped to a 3-bit PFC priority to use in pause frames when DSCP-based PFC is triggered. Configuring the mapping involves mapping the PFC priority value to a no-loss forwarding class when you map the forwarding class to a queue, defining a congestion notification profile to enable PFC on traffic with the desired DSCP value, and configuring a DSCP classifier to associate the PFC priority-mapped forwarding class (along with the loss priority) with the configured DSCP value on which to trigger PFC pause frames.

The peer device should have output PFC and a corresponding flow control queue configured to match the PFC priority configuration on the device.

To configure DSCP-based PFC:

1. Map a lossless forwarding class to a PFC priority—a 3-bit value represented in decimal form (0-7)—to use in the PFC pause frames.

You must also assign an output queue to the forwarding class with the `queue-num` option. The `no-loss` option is required in this case to support lossless behavior for DSCP-based PFC, and the `pfc-priority` statement specifies the priority value mapping, as follows:

```
[edit class-of-service]
user@switch# set forwarding-classes class class-name queue-num queue-number no-loss
user@switch# set forwarding-classes class class-name pfc-priority pfc-priority
```



2. Define an input congestion notification profile to enable PFC on traffic specified by the desired 6-bit DSCP value, and optionally configure the maximum receive unit (MRU) at this time (used to determine PFC buffer headroom space reserved for the link):

```
[edit class-of-service]
user@switch# set congestion-notification-profile name input dscp code-point code-point-bits
pfc mru mru-value
```



**NOTE:** You cannot configure both DSCP-based PFC and IEEE 802.1p PFC under the same congestion notification profile.

3. Set up a DSCP classifier for the configured DSCP value and no-loss forwarding class mapped in the previous steps:

```
[edit class-of-service]
user@switch# set classifiers dscp classifier-name forwarding-class class-name loss-priority
level code-points code-point-bits
```

4. Assign the classifier and congestion notification profile set up in the previous steps to an interface on which you are enabling DSCP-based PFC:

```
[edit class-of-service]
user@switch# set interfaces interface-name classifiers dscp classifier-name
user @switch# set interfaces interface-name congestion-notification-profile profile-name
```

For example, with the following sample commands configuring DSCP-based PFC for interface xe-0/0/1, PFC pause frames will be generated with PFC priority 3 when incoming traffic with DSCP value 110000 becomes congested:

```
set interfaces xe-0/0/1 unit 0 family inet address 10.1.1.2/24
set class-of-service forwarding-classes class fc1 queue-num 1 no-loss
set class-of-service forwarding-classes class fc1 pfc-priority 3
set class-of-service congestion-notification-profile dpfc-cnp input dscp code-point 110000 pfc
set class-of-service classifiers dscp dpfc forwarding-class fc1 loss-priority low code-points 110000
set class-of-service interfaces xe-0/0/1 congestion-notification-profile dpfc-cnp
set class-of-service interfaces xe-0/0/1 classifiers dscp dpfc
```



## RELATED DOCUMENTATION

[Understanding PFC Using DSCP at Layer 3 for Untagged Traffic | 502](#)

[Configuring CoS PFC \(Congestion Notification Profiles\)](#)

[Defining CoS BA Classifiers \(DSCP, DSCP IPv6, IEEE 802.1p\)](#)

[Defining CoS Forwarding Classes](#)

## PFC Watchdog

### IN THIS SECTION

- [PFC Watchdog Overview | 506](#)
- [How to Configure PFC Watchdog | 507](#)

## PFC Watchdog Overview

### IN THIS SECTION

- [Understanding PFC Watchdog | 507](#)

Priority-based flow control (PFC) pause frames are used in lossless Ethernet to pause the link partner from sending packets. These PFC pause frames can propagate through the whole network and can cause the traffic on the PFC streams to halt. Use the PFC watchdog to detect and resolve PFC pause storms.

The PFC watchdog monitors PFC-enabled ports for PFC pause storms. When a PFC-enabled port receives PFC pause frames for an extended period of time and PFC watchdog does not detect flow control frames on that port, PFC watchdog mitigates the situation. It does this by disabling the queue where the PFC pause storm was detected for a configurable length of time called the recovery time. After the recovery time passes, PFC watchdog re-enables the affected queue.

Use [Feature Explorer](#) to confirm platform and release support for specific features.



## Understanding PFC Watchdog

The PFC watchdog has three functions: detection, mitigation, and restoration.

The PFC watchdog checks the status of PFC queues at regular intervals called polling intervals. If the PFC watchdog finds a PFC queue with a non-zero pause timer, it compares the queue's current transmit counter register to the last recorded value. If the PFC queue has not transmitted any packets since the last polling interval, the PFC watchdog checks if there are any packets in the queue. If there are packets on the queue that are not being transmitted and there are no flow control frames on that port, the PFC watchdog detects a stall condition.

After the PFC watchdog detects a stall condition, it disables the queue where it detected the PFC pause storm for a period of time called the recovery time. During that time, it flushes all packets in the queue and prevents new packets from being added to the queue. The system monitors all packet drops on the PFC queue during the recovery time.

When the recovery time ends, the PFC watchdog collects the ingress drop counters and any other drop counters associated with disabling the PFC queue. The PFC watchdog maintains a count of the packets lost during the last recovery and the total number of lost packets due to PFC mitigation since the device was started. The PFC watchdog then restores the queue and re-enables PFC.

## How to Configure PFC Watchdog

### IN THIS SECTION

- [Enable PFC Watchdog | 507](#)
- [Detection | 508](#)
- [Mitigation | 509](#)
- [Restoration | 509](#)
- [Verification | 509](#)
- [Monitoring PFC Watchdog | 510](#)

### Enable PFC Watchdog

PFC watchdog only works for PFC queues. To designate a queue as a PFC queue, use the `flow-control-queue` statement with the queue number:

```
set class-of-service congestion-notification-profile cnp output ieee-802.1 code-point 011 flow-
```



```
control-queue 3
set class-of-service congestion-notification-profile cnp output ieee-802.1 code-point 100 flow-
control-queue 4
```

Enable PFC watchdog using the `pfc-watchdog` statement at the `[edit class-of-service congestion-notification-profile profile-name]` hierarchy level:

```
set class-of-service congestion-notification-profile profile-name pfc-watchdog
```

Enabling PFC watchdog on the congestion notification profile without configuring other options enables the PFC watchdog with the default values. By default, the polling interval is 100 ms, the detection period is set to 2 (that is, two polling intervals, or 200 ms), and the recovery time is 200 ms. To learn how to configure non-default values, read the following sections.

### Detection

The PFC watchdog monitors the PFC-enabled queues periodically for continuous PFC pause assertion by the downstream device when the queue is empty. If this occurs, PFC watchdog detects a stall condition. The system must detect this stall condition within a specified amount of time. This length of time is determined by how you configure two statements: `poll-interval` and `detection`.

The PFC watchdog checks the status of PFC queues at regular intervals. Configure this interval in milliseconds using the `poll-interval` statement. The PFC watchdog checks the status of the queues once per polling interval. The default interval is 100 ms. The minimum interval is 100 ms and the maximum is 1000 ms.

```
set class-of-service congestion-notification-profile profile-name pfc-watchdog poll-interval time
```

The PFC watchdog must detect stall conditions for at least two consecutive polling intervals before it determines that a PFC queue has stalled. Configure the `detection` statement to control how many polling intervals the PFC watchdog waits before it mitigates the stalled traffic. The default is two polling intervals. The maximum number is 10 polling intervals.

```
set class-of-service congestion-notification-profile profile-name pfc-watchdog detection number
of polling intervals
```

The total detection time is the length of the polling interval multiplied by the number of polling intervals.



## Mitigation

When the PFC watchdog detects that a PFC queue has stalled, it moves the queue to the mitigation state. Configure the `pfc-watchdog-action` statement to specify the action that the PFC watchdog takes to mitigate the traffic congestion. The only option is the drop action. When the PFC watchdog detects that a PFC queue has stalled, it drops all queued packets and all newly arriving packets for the stalled PFC queue.

```
set class-of-service congestion-notification-profile profile-name pfc-watchdog watchdog-action
drop
```

## Restoration

Use the `recovery` statement to configure how long the PFC watchdog disables the affected queue for before it restores PFC. The minimum recovery period is 200 ms and the maximum is 10,000 ms.

```
set class-of-service congestion-notification-profile profile-name pfc-watchdog recovery time
```

After the recovery time passes, the PFC watchdog re-enables PFC on the affected queues.

## Verification

Use the following command to verify you have configured the PFC watchdog correctly:

```
show class-of-service congestion-notification-profile
```

Name: cnp, Index: 0

Type: Input

Cable Length: 100

Type: Output

Priority	Flow-Control-Queues
011	

3

Priority	Flow-Control-Queues
100	

4

PFC Watchdog : enabled

PFC-action : drop

Polling Interval : 100 ms



```
Detection Time : 200 ms
Recovery Time : 200 ms
```

The detection time shown is the polling interval multiplied by the detection period. In this case, the polling interval is 100 ms, so the configured detection time was two.

### Monitoring PFC Watchdog

You can view the number of PFC pause storms that have been detected and recovered, as well as the number of packets that have been dropped, on the PFC queues on an interface. Use the following command to view the PFC watchdog statistics on a particular interface.

```
show interfaces interface extensive
...

Priority Flow Control Watchdog Statistics:
      Detected   Recovered   LastPacketDropCount   TotalPacketDropCount
Queue : 0        0           0           0           0
Queue : 1        0           0           0           0
Queue : 2        0           0           0           0
Queue : 3        0           0           0           0
Queue : 4        0           0           0           0
Queue : 5        0           0           0           0
Queue : 6        0           0           0           0
Queue : 7        0           0           0           0
...

```

You can view the actions that PFC watchdog takes in the system log.

- When the PFC watchdog is enabled on a new port, the system log displays this message: CDA PfcWd: PFC Watchdog detection enabled on ifd: et-0/0/16 Poll Interval:100ms Detection Period:200ms Recovery Interval:200ms
- When the PFC watchdog detects a stall condition, the system log displays this message: CDA PfcWd: PFC Storm Detected! on ifd:et-0/0/16 Queue: 3 Priority: 3 BLOCKED for AutoRecovery Recovery Time: 200ms
- When the queue recovers from the PFC pause storm, the system log displays this message: CDA PfcWd: PFC Storm Recovered on Port ifd:et-0/0/16 Queue: 3 Priority: 3 UNBLOCKED after AutoRecovery Recovery Time: 200ms



## RELATED DOCUMENTATION

*Configuring CoS PFC (Congestion Notification Profiles)*

*congestion-notification-profile*

## CoS Explicit Congestion Notification

### IN THIS SECTION

- [How ECN Works | 512](#)
- [WRED Drop Profile Control of ECN Thresholds | 516](#)
- [Dynamic ECN | 519](#)
- [Support, Limitations, and Notes | 519](#)
- [Platform-Specific Behavior | 520](#)

Explicit congestion notification (ECN) enables end-to-end congestion notification between two endpoints on TCP/IP based networks. The two endpoints are an ECN-enabled sender and an ECN-enabled receiver. ECN must be enabled on both endpoints and on all of the intermediate devices between the endpoints for ECN to work properly. Any device in the transmission path that does not support ECN breaks the end-to-end ECN functionality.

ECN notifies networks about congestion with the goal of reducing packet loss and delay by making the sending device decrease the transmission rate until the congestion clears, without dropping packets. RFC 3168, *The Addition of Explicit Congestion Notification (ECN) to IP*, defines ECN.

ECN is disabled by default. Normally, you enable ECN only on queues that handle best-effort traffic because other traffic types use different methods of congestion notification—lossless traffic uses priority-based flow control (PFC) and strict-high priority traffic receives all of the port bandwidth it requires up to the point of a configured maximum rate.

You enable ECN on individual output queues (as represented by forwarding classes) by enabling ECN in the queue scheduler configuration, mapping the scheduler to forwarding classes (queues), and then applying the scheduler to interfaces.

There are two types of ECN that you can use: Static ECN and Dynamic ECN (D-ECN). Static ECN requires you to manually configure the thresholds that trigger congestion notifications, and the thresholds stay the same until you change the setting. Dynamic ECN adjusts the thresholds automatically based on real-time conditions such as queue length and traffic patterns.



Both ECN versions can be enabled on your device at the same time, but only one version can be assigned to a particular queue at a time.



**NOTE:** For ECN to work on a queue, you must also apply a weighted random early detection (WRED) packet drop profile to the queue.

## How ECN Works

Without ECN, devices respond to network congestion by dropping TCP/IP packets. Dropped packets signal the network that congestion is occurring. Devices on the IP network respond to TCP packet drops by reducing the packet transmission rate to allow the congestion to clear. However, the packet drop method of congestion notification and management has some disadvantages. For example, packets are dropped and must be retransmitted. Also, bursty traffic can cause the network to reduce the transmission rate too much, resulting in inefficient bandwidth utilization.

Instead of dropping packets to signal network congestion, ECN marks packets to signal network congestion, without dropping the packets. For ECN to work, all of the devices in the path between two ECN-enabled endpoints must have ECN enabled. ECN is negotiated during the establishment of the TCP connection between the endpoints.

ECN-enabled devices determine the queue congestion state based on the WRED packet drop profile configuration applied to the queue, so each ECN-enabled queue must also have a WRED drop profile. If a queue fills to the level at which the WRED drop profile has a packet drop probability greater than zero (0), the device might mark a packet as experiencing congestion. Whether or not a device marks a packet as experiencing congestion is the same probability as the drop probability of the queue at that fill level.

ECN communicates whether or not congestion is experienced by marking the two least-significant bits in the differentiated services (DiffServ) field in the IP header. The most significant six bits in the DiffServ field contain the Differentiated Services Code Point (DSCP) bits. The state of the two ECN bits signals whether or not the packet is an ECN-capable packet and whether or not congestion has been experienced.

ECN-capable senders mark packets as ECN-capable. If a sender is not ECN-capable, it marks packets as not ECN-capable. If an ECN-capable packet experiences congestion at the egress queue of a device, the device marks the packet as experiencing congestion. When the packet reaches the ECN-capable receiver (destination endpoint), the receiver echoes the congestion indicator to the sender (source endpoint) by sending a packet marked to indicate congestion.

After receiving the congestion indicator from the receiver, the source endpoint reduces the transmission rate to relieve the congestion. This is similar to the result of TCP congestion notification and management, but instead of dropping the packet to signal network congestion, ECN marks the packet and the receiver echoes the congestion notification to the sender. Because the packet is not dropped, the packet does not need to be retransmitted.



### ECN Bits in the DiffServ Field

The two ECN bits in the DiffServ field provide four codes that determine if a packet is marked as an ECN-capable transport (ECT) packet, meaning that both endpoints of the transport protocol are ECN-capable, and if there is congestion experienced (CE), as shown in [Table 50 on page 513](#):

**Table 50: ECN Bit Codes**

ECN Bits (Code)	Meaning
00	Non-ECT—Packet is marked as not ECN-capable
01	ECT(1)—Endpoints of the transport protocol are ECN-capable
10	ECT(0)—Endpoints of the transport protocol are ECN-capable
11	CE—Congestion experienced

Codes 01 and 10 have the same meaning: the sending and receiving endpoints of the transport protocol are ECN-capable. There is no difference between these codes.

### End-to-End ECN Behavior

After the sending and receiving endpoints negotiate ECN, the sending endpoint marks packets as ECN-capable by setting the DiffServ ECN field to ECT(1) (01) or ECT(0) (10). Every intermediate device between the endpoints must have ECN enabled or it does not work.

When a packet traverses a device and experiences congestion at an output queue that uses the WRED packet drop mechanism, the device marks the packet as experiencing congestion by setting the DiffServ ECN field to CE (11). Instead of dropping the packet (as with TCP congestion notification), the device forwards the packet.

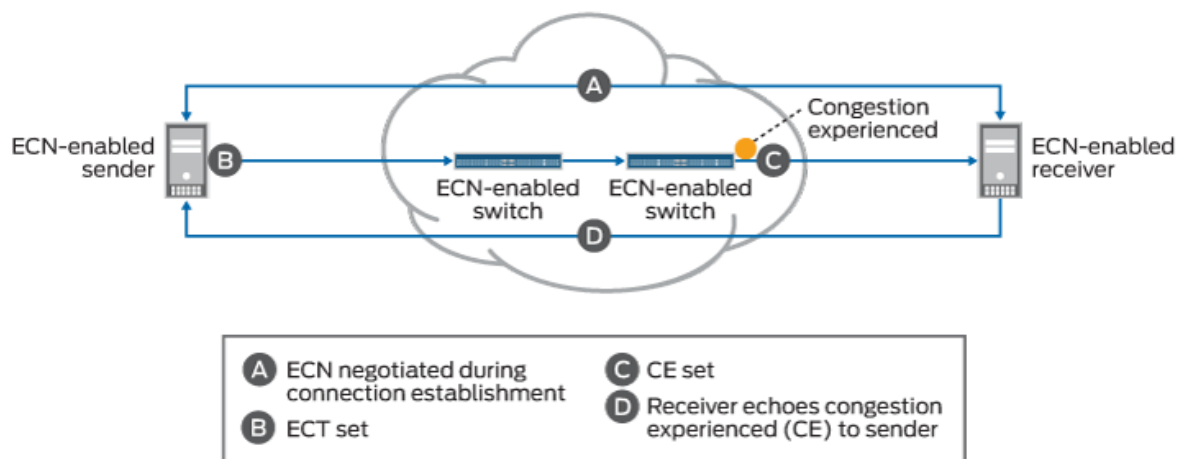


**NOTE:** At the egress queue, the WRED algorithm determines whether or not a packet is drop eligible based on the queue fill level (how full the queue is). If a packet is drop eligible and marked as ECN-capable, the packet can be marked CE and forwarded. If a packet is drop eligible and is not marked as ECN-capable, it might be dropped. See ["WRED Drop Profile Control of ECN Thresholds" on page 516](#) for more information about the WRED algorithm.



When the packet reaches the receiver endpoint, the CE mark tells the receiver that there is network congestion. The receiver then sends (echoes) a message to the sender that indicates there is congestion on the network. The sender acknowledges the congestion notification message and reduces its transmission rate. [Figure 42 on page 514](#) summarizes how ECN works to mitigate network congestion:

**Figure 42: Explicit Congestion Notification**



8042495

End-to-end ECN behavior includes:

1. The ECN-capable sender and receiver negotiate ECN capability during the establishment of their connection.
2. After successful negotiation of ECN capability, the ECN-capable sender sends IP packets with the ECT field set to the receiver.



**NOTE:** You must enable ECN on *all* of the intermediate devices in the path between the sender and the receiver.

3. If the WRED algorithm on a device egress queue determines that the queue is experiencing congestion and the packet is drop eligible, the device can mark the packet as “congestion experienced” (CE) to indicate to the receiver that there is congestion on the network. If the packet has already been marked CE (congestion has already been experienced at the egress of another device), the device forwards the packet with CE marked.

If there is no congestion at the egress queue, the device forwards the packet and does not change the ECT-enabled marking of the ECN bits, so the packet is still marked as ECN-capable but not as experiencing congestion.



4. The receiver receives a packet marked CE to indicate that congestion was experienced along the congestion path.
5. The receiver echoes (sends) a packet back to the sender with the ECE bit (bit 9) marked in the flag field of the TCP header. The ECE bit is the ECN echo flag bit, which notifies the sender that there is congestion on the network.
6. The sender reduces the data transmission rate and sends a packet to the receiver with the CWR bit (bit 8) marked in the flag field of the TCP header. The CWR bit is the congestion window reduced flag bit, which acknowledges to the receiver that the congestion experienced notification was received.
7. When the receiver receives the CWR flag, the receiver stops setting the ECE bit in replies to the sender.

Table 51 on page 515 summarizes the behavior of traffic on ECN-enabled queues.

**Table 51: Traffic Behavior on ECN-Enabled Queues**

Incoming IP Packet Marking of ECN Bits	ECN Configuration on the Output Queue	Action if WRED Algorithm Determines Packet is Drop Eligible	Outgoing Packet Marking of ECN Bits
Non-ECT (00)	Does not matter	Drop.	No ECN bits marked
ECT (10 or 01)	ECN disabled	Drop	Packet dropped—no ECN bits marked
ECT (10 or 01)	ECN enabled	Do not drop. Mark packet as experiencing congestion (CE, bits 11).	Packet marked ECT (11) to indicate congestion
CE (11)	ECN disabled	Drop	Packet dropped—no ECN bits marked
CE (11)	ECN enabled	Do not drop. Packet is already marked as experiencing congestion, forward packet without changing the ECN marking.	Packet marked ECT (11) to indicate congestion

When an output queue is not experiencing congestion as defined by the WRED drop profile mapped to the queue, all packets are forwarded, and no packets are dropped.



## ECN Compared to PFC and Ethernet PAUSE

ECN is an end-to-end network congestion notification mechanism for IP traffic. Priority-based flow control (PFC) (IEEE 802.1Qbb) and Ethernet PAUSE (IEEE 802.3X) are different types of congestion management mechanisms.

ECN requires that an output queue must also have an associated WRED packet drop profile. Output queues used for traffic on which PFC is enabled should not have an associated WRED drop profile. Interfaces on which Ethernet PAUSE is enabled should not have an associated WRED drop profile.

PFC is a peer-to-peer flow control mechanism to support lossless traffic. PFC enables connected peer devices to pause flow transmission during periods of congestion. PFC enables you to pause traffic on a specified type of flow on a link instead of on all traffic on a link. For example, you can (and should) enable PFC on lossless traffic classes such as the `fcoe` forwarding class. Ethernet PAUSE is also a peer-to-peer flow control mechanism, but instead of pausing only specified traffic flows, Ethernet PAUSE pauses all traffic on a physical link.

With PFC and Ethernet PAUSE, the sending and receiving endpoints of a flow do not communicate congestion information to each other across the intermediate devices. Instead, PFC controls flows between two PFC-enabled peer devices (for example, devices) that support data center bridging (DCB) standards. PFC works by sending a pause message to the connected peer when the flow output queue becomes congested. Ethernet PAUSE simply pauses all traffic on a link during periods of congestion and does not require DCB.

PFC works this way: if a device output queue fills to a certain threshold, the device sends a PFC pause message to the connected peer device that is transmitting data. The pause message tells the transmitting device to pause transmission of the flow. When the congestion clears, the device sends another PFC message to tell the connected peer to resume transmission. (If the output queue of the transmitting device also reaches a certain threshold, that device can in turn send a PFC pause message to the connected peer that is transmitting to it. In this way, PFC can propagate a transmission pause back through the network.)

See *Understanding CoS Flow Control (Ethernet PAUSE and PFC)* for more information. You can also refer to *Understanding PFC Functionality Across Layer 3 Interfaces*.

## WRED Drop Profile Control of ECN Thresholds

You apply WRED drop profiles to forwarding classes (which are mapped to output queues) to control how the device marks ECN-capable packets. A scheduler map associates a drop profile with a scheduler and a forwarding class, and then you apply the scheduler map to interfaces to implement the scheduling properties for the forwarding class on those interfaces.

Drop profiles define queue fill level (the percentage of queue fullness) and drop probability (the percentage probability that a packet is dropped) pairs. When a queue fills to a specified level, traffic that matches the drop profile has the drop probability paired with that fill level. When you configure a drop



profile, you configure pairs of fill levels and drop probabilities to control how packets drop at different levels of queue fullness.

The first fill level and drop probability pair is the drop start point. Until the queue reaches the first fill level, packets are not dropped. When the queue reaches the first fill level, packets that exceed the fill level have a probability of being dropped that equals the drop probability paired with the fill level.

The last fill level and drop probability pair is the drop end point. When the queue reaches the last fill level, all packets are dropped unless they are configured for ECN.



**NOTE:** Lossless queues (forwarding class configured with the `no-loss` packet drop attribute) and strict-high priority queues do not use drop profiles. Lossless queues use PFC to control the flow of traffic. Strict-high priority queues receive all of the port bandwidth they require up to the configured maximum bandwidth limit.

Different devices support different amounts of fill level/drop probability pairs in drop profiles.



**NOTE:** Do not configure the last fill level as 100 percent.

The drop profile configuration affects ECN packets as follows:

- Drop start point—ECN-capable packets might be marked as congestion experienced (CE).
- Drop end point—ECN-capable packets are always marked CE.

As a queue fills from the drop start point to the drop end point, the probability that an ECN packet is marked CE is the same as the probability that a non-ECN packet is dropped if you apply the drop profile to best-effort traffic. As the queue fills, the probability of an ECN packet being marked CE increases, just as the probability of a non-ECN packet being dropped increases when you apply the drop profile to best-effort traffic.

At the drop end point, all ECN packets are marked CE, but the ECN packets are not dropped. When the queue fill level exceeds the drop end point, all ECN packets are marked CE. At this point, all non-ECN packets are dropped. ECN packets (and all other packets) are tail-dropped if the queue fills completely.

To configure a WRED packet drop profile and apply it to an output queue (using hierarchical scheduling on devices that support ETS):

1. Configure a drop profile using the statement `set class-of-service drop-profiles profile-name interpolate fill-level drop-start-point fill-level drop-end-point drop-probability 0 drop-probability percentage`.
2. Map the drop profile to a queue scheduler using the statement `set class-of-service schedulers scheduler-name drop-profile-map loss-priority (low | medium-high | high) protocol any drop-profile profile-name`. The name of the drop-profile is the name of the WRED profile configured in Step 1.



3. Map the scheduler, which Step 2 associates with the drop profile, to the output queue using the statement `set class-of-service scheduler-maps map-name forwarding-class forwarding-class-name scheduler scheduler-name`. The forwarding class identifies the output queue. Forwarding classes are mapped to output queues by default, and can be remapped to different queues by explicit user configuration. The scheduler name is the scheduler configured in Step 2.
4. Associate the scheduler map with a traffic control profile using the statement `set class-of-service traffic-control-profiles tcp-name scheduler-map map-name`. The scheduler map name is the name configured in Step 3.
5. Associate the traffic control profile with an interface using the statement `set class-of-service interface interface-name forwarding-class-set forwarding-class-set-name output-traffic-control-profile tcp-name`. The output traffic control profile name is the name of the traffic control profile configured in Step 4.

The interface uses the scheduler map in the traffic control profile to apply the drop profile (and other attributes, including the enable ECN attribute) to the output queue (forwarding class) on that interface. Because you can use different traffic control profiles to map different schedulers to different interfaces, the same queue number on different interfaces can handle traffic in different ways.

You can configure a WRED packet drop profile and apply it to an output queue on devices that support port scheduling (ETS hierarchical scheduling is either not supported or not used). To configure a WRED packet drop profile and apply it to an output queue on devices that support port scheduling (ETS hierarchical scheduling is either not supported or not used):

1. Configure a drop profile using the statement `set class-of-service drop-profiles profile-name interpolate fill-level level1 level2 ... level32 drop-probability probability1 probability2 ... probability32`. You can specify as few as two fill level/drop probability pairs or as many as 32 pairs.
2. Map the drop profile to a queue scheduler using the statement `set class-of-service schedulers scheduler-name drop-profile-map loss-priority (low | medium-high | high) drop-profile profile-name`. The name of the drop-profile is the name of the WRED profile configured in Step 1.
3. Map the scheduler, which Step 2 associates with the drop profile, to the output queue using the statement `set class-of-service scheduler-maps map-name forwarding-class forwarding-class-name scheduler scheduler-name`. The forwarding class identifies the output queue. Forwarding classes are mapped to output queues by default, and can be remapped to different queues by explicit user configuration. The scheduler name is the scheduler configured in Step 2.
4. Associate the scheduler map with an interface using the statement `set class-of-service interfaces interface-name scheduler-map scheduler-map-name`.

The interface uses the scheduler map to apply the drop profile (and other attributes) to the output queue mapped to the forwarding class on that interface. Because you can use different scheduler maps on different interfaces, the same queue number on different interfaces can handle traffic in different ways.



## Dynamic ECN

Dynamic ECN enhances the ECN feature set by providing a way to automate the thresholds triggering a congestion notification event. Junos OS Evolved monitors real-time conditions like queue length and traffic patterns to evaluate whether or not a threshold should be adjusted. This results in a faster response to congestion events than static ECN, and improves congestion control efficiency.

D-ECN is more difficult to implement than static ECN, and requires active monitoring. You should evaluate your network conditions and configuration to decide if D-ECN is the best fit for your network.

You can check [Feature Explorer](#) to see if your device supports D-ECN.

## Support, Limitations, and Notes

If the WRED algorithm that is mapped to a queue does not find a packet drop eligible, then the ECN configuration and ECN bits marking does not matter. The packet transport behavior is the same as when ECN is not enabled.

ECN is disabled by default. Normally, you enable ECN only on queues that handle best-effort traffic, and you do not enable ECN on queues that handle lossless traffic or strict-high priority traffic.

ECN supports the following:

- IPv4 and IPv6 packets
- Untagged, single-tagged, and double-tagged packets
- The outer IP header of IP tunneled packets (but not the inner IP header)

ECN does not support the following:

- IP packets with MPLS encapsulation
- The inner IP header of IP tunneled packets (however, ECN works on the outer IP header)
- Multicast, broadcast, and destination lookup fail (DLF) traffic
- Non-IP traffic



**NOTE:** To apply a WRED drop profile to non-ECT traffic, configure a multifield (MF) classifier to assign non-ECT traffic to a different output queue that is not ECN-enabled, and then apply the WRED drop profile to that queue.



## Platform-Specific Behavior

Use [Feature Explorer](#) to confirm platform and release support for ECN.

Use the following table to review platform-specific behaviors for this feature.

Platform	Difference
PTX10001-36MR, PTX10004, PTX10008, PTX10016	<p>You can implement low-threshold ECN (begin ECN marking as soon as the buffer starts filling up) by defining a buffer rate for a scheduler and a low percentage drop profile.</p> <p>The buffer rate acts as the base rate for buffer size calculation. The buffer rate is the target rate of a VOQ, which is the intended egress queue rate during typical congestion. Set the buffer-rate at the <code>[edit class-of-service schedulers <i>scheduler-name</i>]</code> hierarchy level.</p> <p>You can also define more granular (to the tenths of a percent) and smaller fill-level percentages for drop profiles. That is, you can set a fill-level as low as 0.7 percent.</p>



*(Continued)*

Platform	Difference
QFX5000 Series	<p>On QFX5K platforms, ECN functionality is tightly integrated with WRED thresholds. WRED thresholds are static, so ECN also works based on static calculations of buffer thresholds. However, actual shared buffer usage of queues is dynamic. Following are the formulas used for ECN marking threshold calculations at the time of ECN configuration.</p> <ul style="list-style-type: none"> <li>• max buffer access eligibility for ECN enabled queue = <math>((\text{shared pool size} * \text{hardware\_alpha}) / (1 + \text{hardware\_alpha})) + \text{egress queue dedicated buffer}</math></li> <li>• ECN marking start threshold = WRED start fill level percent * Max buffer access eligibility for ECN enabled queue</li> <li>• ECN 100% marking threshold = WRED end fill level percent * Max buffer access eligibility for ECN enabled queue</li> </ul> <p>During congestion for ECN capable packets, ECN CE marking starts after ECN marking start threshold is reached. ECN capable packets are probabilistically ECN CE marked until ECN 100% marking threshold is reached. After this threshold, all ECN capable packets are ECN CE marked until the queue reaches max buffer access eligibility for ECN enabled queue. After this threshold, tail drops occur.</p> <p>In the above calculation of max buffer access eligibility for ECN enabled queue, the best case scenario of a single queue alone competing for shared buffer space is assumed. However, the actual shared buffer use for a congested queue can decrease dynamically based on the number of competing queues for the shared buffer at any point of time. Following is the formula to calculate the actual dynamic max buffer usage per queue.</p> <ul style="list-style-type: none"> <li>• actual max buffer usage for ECN enabled queue = <math>(\text{shared pool size} * \text{hw\_alpha}) / (1 + (\text{hw\_alpha} * \text{number of competing queues})) + \text{egress queue dedicated buffer} + \text{ingress Pg dedicated buffer by the traffic flow}</math></li> </ul> <p>Two parameters, number of competing queues and ingress Pg dedicated buffer by the traffic flow, used in actual dynamic max buffer usage per queue calculation cannot be considered while</p>



*(Continued)*

Platform	Difference
	<p>calculating the ECN threshold as both these parameters are dynamic in nature. This creates a possibility that actual max buffer usage for an ECN enabled queue can be below the calculated static ECN marking thresholds.</p> <p>Therefore, with certain shared buffer and WRED fill level configurations, there is a possibility of packet tail drops due to shared buffer exhaustion occurring even before ECN marking on ECN enabled lossy queues. For lossless queues, due to the above limitation, PFC can start from an ingress port before ECN marking, as the PFC XOFF threshold is dynamic, unlike the static ECN threshold. You can determine proper ECN marking thresholds by monitoring the peak buffer usage of congested queues and fine tuning the ECN/WRED thresholds accordingly.</p>
QFX10000 Series	<ul style="list-style-type: none"> <li>On QFX10000 switches, when you enable a queue for ECN and apply a WRED drop profile to the queue, the WRED drop profile only sets the thresholds for marking ECN traffic as experiencing congestion (CE, 11). On ECN-enabled queues, the WRED drop profile does not set drop thresholds for non-ECT (00) traffic (traffic that is not ECN-capable). Instead, the switch uses the tail-drop algorithm on traffic that is marked non-ECT on ECN-enabled queues during periods of congestion.</li> </ul>

## RELATED DOCUMENTATION

[Example: Configuring Static and Dynamic ECN](#) | 522

## Example: Configuring Static and Dynamic ECN

### IN THIS SECTION

● [Requirements](#) | 523



- Overview | 523
- Configuration | 525
- Verification | 529

This example shows how to enable explicit congestion notification (ECN) on an output queue.

## Requirements

This example uses the following hardware and software components:

- One device that supports ECN.



**NOTE:** Use [Feature Explorer](#) to confirm platform and release support for static and dynamic ECN.

## Overview

ECN enables end-to-end congestion notification between two endpoints on TCP/IP based networks. The two endpoints are an ECN-enabled sender and an ECN-enabled receiver. ECN must be enabled on both endpoints and on all of the intermediate devices between the endpoints for ECN to work properly. Any device in the transmission path that does not support ECN breaks the end-to-end ECN functionality.

A weighted random early detection (WRED) packet drop profile must be applied to the output queues on which ECN is enabled. ECN uses the WRED drop profile thresholds to mark packets when the output queue experiences congestion.

ECN reduces packet loss by forwarding ECN-capable packets during periods of network congestion instead of dropping those packets. (TCP notifies the network about congestion by dropping packets.) During periods of congestion, ECN marks ECN-capable packets that egress from congested queues. When the receiver receives an ECN packet that is marked as experiencing congestion, the receiver echoes the congestion state back to the sender. The sender then reduces its transmission rate to clear the congestion.

ECN is disabled by default. You can enable ECN on best-effort traffic. ECN should not be enabled on lossless traffic queues, which uses priority-based flow control (PFC) for congestion notification, and ECN should not be enabled on strict-high priority traffic queues.

There are two types of ECN available: Static ECN and Dynamic ECN. Static ECN requires you to manually set the threshold levels that trigger a notification event. Dynamic ECN automatically adjusts the thresholds based on real-time conditions like queue length and traffic patterns.



To enable ECN on an output queue, you not only need to enable ECN in the queue scheduler, you also need to:

- Configure a WRED packet drop profile.
- Configure a queue scheduler that includes the WRED drop profile and enables ECN. (This example shows only ECN and drop profile configuration; you can also configure bandwidth, priority, and buffer settings in a scheduler.)
- Map the queue scheduler to a forwarding class (output queue) in a scheduler map.
- If you are using enhanced transmission selection (ETS) hierarchical scheduling, add the forwarding class to a forwarding class set (priority group).
- If you are using ETS, associate the queue scheduler map with a traffic control profile (priority group scheduler for hierarchical scheduling).
- If you are using ETS, apply the traffic control profile and the forwarding class set to an interface. On that interface, the output queue uses the scheduler mapped to the forwarding class, as specified by the scheduler map attached to the traffic control profile. This enables ECN on the queue and applies the WRED drop profile to the queue.

If you are using port scheduling, apply the scheduler map to an interface. On that interface, the output queue uses the scheduler mapped to the forwarding class in the scheduler map, which enables ECN on the queue and applies the WRED drop profile to the queue.

[Table 52 on page 524](#) shows the configuration components for this example.

**Table 52: Components of the ECN Configuration Example**

Component	Settings
Drop profile (with two fill level/ drop probability pairs)	Name: be-dp Drop start fill level: 30 percent Drop end fill level: 75 percent Drop probability at drop start (minimum drop rate): 0 percent Drop probability at drop end (maximum drop rate): 80 percent
Scheduler	Name: be-sched ECN: enabled Drop profile: be-dp Transmit rate: 25% Buffer size: 25% Priority: low



**Table 52: Components of the ECN Configuration Example *(Continued)***

Component	Settings
Scheduler map	Name: be-map Forwarding class: best-effort Scheduler: be-sched  <b>NOTE:</b> By default, the best-effort forwarding class is mapped to output queue 0.
Forwarding class set (ETS only)	Name: be-pg Forwarding class: best-effort (queue 0)
Traffic control profile (ETS only)	Name: be-tcp Scheduler map: be-map
Interface (ETS only)	Name: xe-0/0/20 Forwarding class set: be-pg (Output) traffic control profile: be-tcp
Interface (port scheduling only)	Name: xe-0/0/20



**NOTE:** Only devices that support ETS hierarchical scheduling support forwarding class set and traffic control profile configuration. Direct port scheduling does not use the hierarchical scheduling structure.

## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 526](#)
- [Configuring Static ECN | 526](#)
- [Configuring Dynamic ECN | 528](#)



## CLI Quick Configuration

To quickly configure the drop profile, scheduler with ECN enabled, and to map the scheduler to an output queue on an interface, copy the following commands, paste them in a text file, remove line breaks, change variables and details to match your network configuration, and then copy and paste the commands into the CLI at the [edit] hierarchy level.

## ETS Quick Configuration

```
[edit class-of-service]
set drop-profile be-dp interpolate fill-level 30 fill-level 75 drop-probability 0 drop-
probability 80
set schedulers be-sched explicit-congestion-notification
set schedulers be-sched drop-profile-map loss-priority low protocol any drop-profile be-dp
set schedulers be-sched transmit-rate percent 25
set schedulers be-sched buffer-size percent 25
set schedulers be-sched priority low
set scheduler-maps be-map forwarding-class best-effort scheduler be-sched
set forwarding-class-sets be-pg class best-effort
set traffic-control-profiles be-tcp scheduler-map be-map
set interfaces xe-0/0/20 forwarding-class-set be-pg output-traffic-control-profile be-tcp
```

## Port Scheduling Quick Configuration

```
[edit class-of-service]
set drop-profile be-dp interpolate fill-level 30 fill-level 75 drop-probability 0 drop-
probability 80
set schedulers be-sched explicit-congestion-notification
set schedulers be-sched drop-profile-map loss-priority low protocol any drop-profile be-dp
set schedulers be-sched transmit-rate percent 25
set schedulers be-sched buffer-size percent 25
set schedulers be-sched priority low
set scheduler-maps be-map forwarding-class best-effort scheduler be-sched
set interfaces xe-0/0/20 scheduler-map be-map
```

## Configuring Static ECN

### Step-by-Step Procedure

To configure Static ECN:



1. Configure the WRED packet drop profile `be-dp`. This example uses a drop start point of 30 percent, a drop end point of 75 percent, a minimum drop rate of 0 percent, and a maximum drop rate of 80 percent:

```
[edit class-of-service]
user@switch# set drop-profile be-dp interpolate fill-level 30 fill-level 75 drop-probability
0 drop-probability 80
```

2. Create the scheduler `be-sched` with ECN enabled and associate the drop profile `be-dp` with the scheduler:

```
[edit class-of-service]
user@switch# set schedulers be-sched explicit-congestion-notification
user@switch# set schedulers be-sched drop-profile-map loss-priority low protocol any drop-
profile be-dp
user@switch# set be-sched transmit-rate percent 25
user be-sched transmit-rate percent 25
user@switch# set be-sched buffer-size percent 25
user@switch# set be-sched buffer-size percent 25
user@switch# set be-sched priority low
```

3. Map the scheduler `be-sched` to the best-effort forwarding class (output queue 0) using scheduler map `be-map`:

```
[edit class-of-service]
user@switch# set scheduler-maps be-map forwarding-class best-effort scheduler be-sched
```

4. If you are using ETS, add the forwarding class `best-effort` to the forwarding class set `be-pg`; if you are using direct port scheduling, skip this step:

```
[edit class-of-service]
user@switch# set forwarding-class-sets be-pg class best-effort
```



5. If you are using ETS, associate the scheduler map `be-map` with the traffic control profile `be-tcp`; if you are using direct port scheduling, skip this step:

```
[edit class-of-service]
user@switch# set traffic-control-profiles be-tcp scheduler-map be-map
```

6. If you are using ETS, associate the traffic control profile `be-tcp` and the forwarding class set `be-pg` with the interface on which you want to enable ECN on the best-effort queue:

```
[edit class-of-service]
user@switch# set interfaces xe-0/0/20 forwarding-class-set be-pg output-traffic-control-profile be-tcp
```

If you are using direct port scheduling, associate the scheduler map `be-map` with the interface on which you want to enable ECN on the best-effort queue:

```
[edit class-of-service]
user@switch# set interfaces xe-0/0/20 scheduler-map be-map
```

## Configuring Dynamic ECN

### Step-by-Step Procedure

To configure Dynamic ECN:

1. Follow the steps to configure Static ECN.
2. Enable the Dynamic ECN profile:

```
[edit class-of-service]
user@switch# set decn-profile profile-name queue queue number floor floor value offset offset value
```

Floor and offset values range from 0 to 524287.

Floor and offset values do not need to be configured. If no value is selected, the maximum value of 524287 will be selected by default.



3. Assign the Dynamic ECN profile to an interface:

```
[edit class-of-service]
user@switch# set interfaces interface decn-profile profile-name
```

### Verification

IN THIS SECTION

- [Verifying That ECN Is Enabled | 529](#)

### Verifying That ECN Is Enabled

#### Purpose

Verify that ECN is enabled in the scheduler be-sched by showing the configuration for the scheduler map be-map.

#### Action

Display the scheduler map configuration using the operational mode command `show class-of-service scheduler-map be-map`:

```
user@switch> show class-of-service scheduler-map be-map
Scheduler map: be-map, Index: 12240

Scheduler:be-sched, Forwarding class: best-effort, Index: 115
  Transmit rate: 25 percent, Rate Limit: none, Buffer size: 25 percent,
  Buffer Limit: none, Priority: low
  Excess Priority: unspecified, Explicit Congestion Notification: enable
  ECN Type: static/dynamic Offset offset-value Floor floor-value
  Drop profiles:
    Loss priority  Protocol  Index  Name
    Low           any       3312   be-dp
    Medium-high   any       1      <default-drop-profile>
    High          any       1      <default-drop-profile>
```



## Meaning

The `show class-of-service scheduler-map` operational command shows the configuration of the scheduler associated with the scheduler map and the forwarding class mapped to that scheduler. The output shows that:

- The scheduler associated with the scheduler map is `be-sched`.
- The scheduler map applies to the forwarding class `best-effort` (output queue 0).
- The scheduler `be-sched` has a transmit rate of 25 percent, a queue buffer size of 25 percent, and a drop priority of `low`.
- Explicit congestion notification state is `enable`.
- The WRED drop profile used for low drop priority traffic is `be-dp`.

## RELATED DOCUMENTATION

| [CoS Explicit Congestion Notification](#) | 511



# Altering Outgoing Packet Headers Using Rewrite Rules

## IN THIS CHAPTER

- [Rewriting Packet Headers to Ensure Forwarding Behavior | 532](#)
- [Applying Default Rewrite Rules | 535](#)
- [Configuring Rewrite Rules | 538](#)
- [Configuring Rewrite Rules Based on PLP | 540](#)
- [Applying IEEE 802.1p Rewrite Rules to Dual VLAN Tags | 541](#)
- [Applying IEEE 802.1ad Rewrite Rules to Dual VLAN Tags | 543](#)
- [Rewriting IEEE 802.1p Packet Headers with an MPLS EXP Value | 544](#)
- [Setting IPv6 DSCP and MPLS EXP Values Independently | 546](#)
- [Configuring DSCP Values for IPv6 Packets Entering the MPLS Tunnel | 547](#)
- [Setting Ingress DSCP Bits for Multicast Traffic over Layer 3 VPNs | 549](#)
- [Applying Rewrite Rules to Output Logical Interfaces | 550](#)
- [Rewriting MPLS and IPv4 Packet Headers | 553](#)
- [Rewriting the EXP Bits of All Three Labels of an Outgoing Packet | 558](#)
- [Defining a Custom Frame Relay Loss Priority Map | 560](#)
- [Example: Per-Node Rewriting of EXP Bits | 562](#)
- [Example: Rewriting CoS Information at the Network Border to Enforce CoS Strategies | 563](#)
- [Example: Remarking Diffserv Code Points to MPLS EXPs to Carry CoS Profiles Across a Service Provider's L3VPN MPLS Network | 577](#)
- [Example: Remarking Diffserv Code Points to 802.1P PCPs to Carry CoS Profiles Across a Service Provider's VPLS Network | 609](#)
- [Assigning Rewrite Rules on a Per-Customer Basis Using Policy Maps | 640](#)
- [Host Outbound Traffic IEEE802.1p Rewrite | 646](#)



## 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.

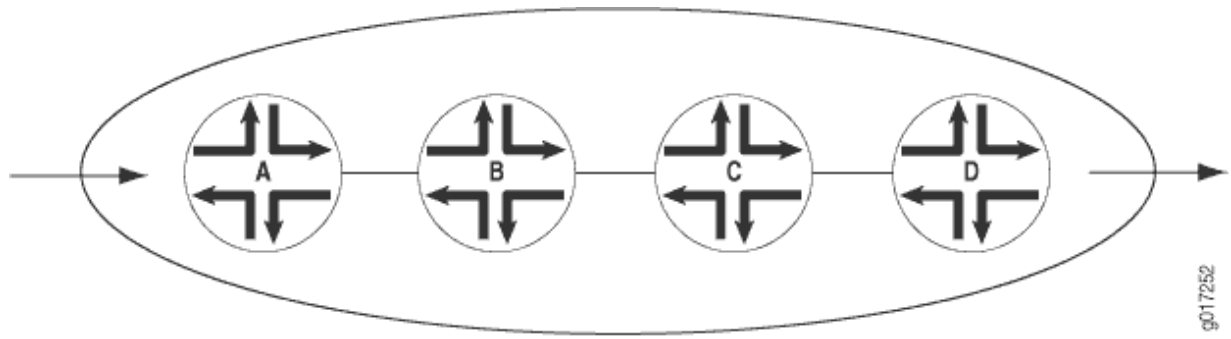
You configure rewrite rules to alter CoS values in outgoing packets on the outbound interfaces of an edge router to meet the policies of a targeted peer. This allows the downstream routing device in a neighboring network to classify each packet into the appropriate service group.

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 43 on page 533](#) 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 43: 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 53 on page 535 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 56, and the default forwarding classes shown in "Default Forwarding Classes" on page 277.

When the software detects packets whose CoS values match the forwarding class and PLP values in the first two columns in Table 53 on page 535, the software maps the header bits of those packets to the code-point aliases in the last column in Table 53 on page 535. 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 56.

**Table 53: Default Packet Header Rewrite Mappings**

Map from Forwarding Class	Loss Priority	Map to DSCP/DSCP IPv6/ EXP/IEEE/IP
expedited-forwarding	low	ef
expedited-forwarding	high	ef
assured-forwarding	low	af11



**Table 53: Default Packet Header Rewrite Mappings (Continued)**

Map from Forwarding Class	Loss Priority	Map to DSCP/DSCP IPv6/ EXP/IEEE/IP
assured-forwarding	high	af12 (DSCP/DSCP IPv6/EXP)
best-effort	low	be
best-effort	high	be
network-control	low	nc1/cs6
network-control	high	nc2/cs7

Starting with Junos OS Release 21.1, PTX routers support two forwarding classes and four loss priorities for MPLS EXP default rewrite rules, as [Table 54 on page 536](#) shows.

**Table 54: Default MPLS EXP Rewrite Mapping for PTX Routers**

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



**Table 54: Default MPLS EXP Rewrite Mapping for PTX Routers** *(Continued)*

Map from Forwarding Class	Loss Priority	Map to MPLS EXP Bits
network-control	high	111

In the following example, the `so-1/2/3.0` interface is assigned the default DSCP rewrite rule. One result of this configuration is that each packet exiting the interface with the `expedited-forwarding` forwarding class and the `high` or `low` loss priority has its DSCP bits rewritten to the DSCP `ef` code-point alias. ["Default Aliases for CoS Value Bit Patterns Overview" on page 56](#) 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 56](#) 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 56](#) and [Table 53 on page 535](#).

```
class-of-service {
  interfaces {
    so-1/2/3 {
      unit 0 {
        rewrite-rules {
          dscp default;
        }
      }
    }
  }
}
```

## RELATED DOCUMENTATION

[Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic](#) | 44



## 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 10.

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.

Change History Table

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

Release	Description
18.1	Starting with Junos OS Release 18.1R1, MX Series routers with MPCs support rewrite rules that rewrite the first three bits of the IPv6 DSCP value through the <code>inet6-precedence</code> statement at the <code>[edit class-of-service rewrite-rules]</code> hierarchy level.

RELATED DOCUMENTATION

[Applying Rewrite Rules to Output Logical Interfaces](#) | 550

*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 275](#).



## Applying IEEE 802.1p Rewrite Rules to Dual VLAN Tags

### IN THIS SECTION

- [Example: Applying an IEEE 802.1p Rewrite Rule to Dual VLAN Tags | 542](#)

By default, when you apply an IEEE 802.1p rewrite rule to an output logical interface, the software rewrites the IEEE bits in the outer VLAN tag only.

For Gigabit Ethernet IQ2 PICs, 10-port 10-Gigabit OSE PICs, and 10-Gigabit Ethernet IQ2 PICs only, you can rewrite the IEEE bits in both the outer and inner VLAN tags of the tagged Ethernet frames. When you enable class of service (CoS) rewrite for both tags, the same IEEE 802.1p rewrite table is used for the inner and outer VLAN tag.

For IQ PICs, you can only configure one IEEE 802.1 rewrite rule on a physical port. All logical ports (units) on that physical port should apply the same IEEE 802.1 rewrite rule.

To rewrite both the outer and inner VLAN tags, include the `vlan-tag outer-and-inner` statement at the [edit class-of-service interfaces *interface-name* unit *logical-unit-number* rewrite-rules ieee-802.1 (*rewrite-name* | default)] hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules
  ieee-802.1 (rewrite-name | default)]
  vlan-tag outer-and-inner;
```

To explicitly specify the default behavior, include the `vlan-tag outer` statement at the [edit class-of-service interfaces *interface-name* unit *logical-unit-number* rewrite-rules ieee-802.1 (*rewrite-name* | default)] hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules
  ieee-802.1 (rewrite-name | default)]
  vlan-tag outer;
```

For more information about VLAN tags, see the [Junos OS Network Interfaces Library for Routing Devices](#).

On MX routers and EX Series switches, you can perform IEEE 802.1p and DEI rewriting based on forwarding class and PLP at the VPLS ingress PE. You rewrite (mark) the IEEE 802.1p or DEI bits on frames at the VPLS ingress PE based on the value of the forwarding class and PLP established for the



traffic. You can rewrite either the outer tag only or the outer and inner tag. When both tags are rewritten, both get the same value. To configure these rewrite rules, include the `ieee-802.1` statement at the `[edit class-of-services routing-instance routing-instance-name rewrite-rules]` hierarchy level.



**NOTE:** For MX80, MX240, MX480, and MX960 routers with MPC/MICs, rewrite on LSI interfaces is not supported (the routers, with DPC, do support rewrite on LSI interfaces).

On routing devices with IQ2 or IQ2-E PICs, you can perform IEEE 802.1p and DEI rewriting based on forwarding-class and packet loss priority (PLP) at the VPLS ingress provider edge (PE) router. You rewrite (mark) the IEEE 802.1p or DEI bits on frames at the VPLS ingress PE based on the value of the forwarding-class and PLP established for the traffic. You can rewrite either the outer tag only or both the outer and inner tags. When both tags are rewritten, both get the same value.



**NOTE:** The 10-port 10-Gigabit OSE PIC does not support DEI rewriting based on forwarding class and PLP at the VPLS ingress PE.

To configure these rewrite rules, include the `ieee-802.1` statement at the `[edit class-of-services routing-instance routing-instance-name rewrite-rules]` hierarchy level.

### Example: Applying an IEEE 802.1p Rewrite Rule to Dual VLAN Tags

Apply the `ieee8021p-rwrule1` rewrite rule to both inner and outer VLAN tags of Ethernet-tagged frames exiting the `ge-0/0/0.0` interface:

```
class-of-service {
  interfaces {
    ge-0/0/0 {
      unit 0 {
        rewrite-rules {
          ieee-802.1 ieee8021p-rwrule1 vlan-tag outer-and-inner;
        }
      }
    }
  }
}
```



## Applying IEEE 802.1ad Rewrite Rules to Dual VLAN Tags

### IN THIS SECTION

- [Example: Applying an IEEE 802.1ad Rewrite Rule to Dual VLAN Tags | 544](#)

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](#) | 547



## 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](#) | 546

## 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 532](#).

This section describes this configuration from a CoS perspective. The examples are not complete multicast or VPN configurations. For more information about multicast, see the [Junos OS Multicast Protocols User Guide](#). For more information about Layer 3 VPNs, see the [Junos OS VPNs Library for Routing Devices](#).

To configure the rewrite rules on the core-facing interface of the ingress PE, include the `rewrite-rules` statement at the `[edit class-of-service]` hierarchy level. You apply the rule to the proper ingress interface at the `[edit class-of-service interfaces]` hierarchy level to complete the configuration. This ingress DSCP rewrite is independent of classifiers placed on ingress traffic arriving on the customer-facing interface of the PE router.



The rewrite rules are applied to all unicast packets and multicast groups. You cannot configure different rewrite rules for different multicast groups. The use of DSCPv6 bits is not supported because IPv6 multicast is not supported. You can configure another rewrite rule for the EXP bits on MPLS CE-CE unicast traffic.

This example defines a rewrite rule called `dscp-rule` that establishes a value of `000000` for best-effort traffic. The rule is applied to the outgoing, core-facing PE interface `ge-2/3/0`.

```
[edit class-of-service]
rewrite-rules {
  dscp dscp-rule {
    forwarding-class best-effort {
      loss-priority low code-point 000000;
    }
  }
}

[edit class-of-service interfaces]
ge-2/3/0 {
  unit 0 {
    rewrite-rules {
      dscp dscp-rule;
    }
  }
}
```

## Applying Rewrite Rules to Output Logical Interfaces

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.

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.



**NOTE:** ACX 5448 supports rewrite on services such as:

- L2 (VLAN, VLAN-CCC - Priority Code Point (PCP))
- L3 (IPv4 and IPv6 - Differentiated Services Code Point (DSCP))
- MPLS (EXP - Experimental Bits)

For L2, rewrite information resides in the VLAN translation entry at the egress side. Without that entry, rewrite does not take place.

For VLAN-CCC (local cross-connect), the rewrite is supported by enabling VLAN translation entry in the same manner as that for L2.

## RELATED DOCUMENTATION

[Setting IPv6 DSCP and MPLS EXP Values Independently](#) | 546

[Configuring DSCP Values for IPv6 Packets Entering the MPLS Tunnel](#) | 547



# Rewriting MPLS and IPv4 Packet Headers

## IN THIS SECTION

- [Example: Rewriting MPLS and IPv4 Packet Headers | 555](#)
- [Example: Simultaneous DSCP and EXP Rewrite | 557](#)

You can apply a rewrite rule to MPLS and IPv4 packet headers simultaneously. This allows you to initialize MPLS EXP and IP precedence bits at LSP ingress. You can configure different rewrite rules depending on whether the traffic is VPN or non-VPN.



Devices running Junos OS Evolved do not support different rewrite rules for VPN and non-VPN traffic.

The default MPLS EXP rewrite rules are shown in [Table 55 on page 553](#).

**Table 55: Default MPLS EXP Rewrite Rules**

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



Table 55: Default MPLS EXP Rewrite Rules (*Continued*)

Forwarding Class	Loss Priority	MPLS EXP Rewrite Value
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-any`—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.



Junos OS Evolved supports only the `mpls-any` option.

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 535](#).

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.

## Rewriting the EXP Bits of All Three Labels of an Outgoing Packet

### IN THIS SECTION

- [Example: Rewriting the EXP Bits of All Three Labels of an Outgoing Packet | 559](#)

In interprovider, carrier-of-carrier, and complex traffic engineering scenarios, it is sometimes necessary to push three labels on the next hop, using a swap-push-push or triple-push operation.

By default, the top MPLS EXP label of an outgoing packet is not rewritten when you configure swap-push-push and triple-push operations. On M Series routers, you can rewrite the EXP bits of all three labels of an outgoing packet, thereby maintaining the CoS of an incoming MPLS or non-MPLS packet.

When the software performs a swap-push-push operation and no rewriting is configured, the EXP fields of all three labels are the same as in the old label. If there is EXP rewriting configured, the EXP bits of the bottom two labels are overwritten with the table entry. The EXP setting of the top label is retained even with rewriting.

To push three labels on all incoming MPLS packets, include the `exp-swap-push-push default` statement at the `[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules]
exp-swap-push-push default;
```



When the software performs a push-push-push operation and if no rewriting is configured, the EXP fields of the bottom two labels are zero. If EXP rewriting is configured, the EXP fields of the bottom two labels are rewritten with the table entry's rewrite value. The EXP field of the top label is inserted with the Qn+PLP value. This Qn reflects the final classification by a multifield classifier if one exists, regardless of whether rewriting is configured.



**NOTE:** The `exp-push-push-push` and `exp-swap-push-push` configuration on the egress interface does not rewrite the top label's EXP field with the Qn+PLP value on an IQ or IQ2 PIC.

To push three labels on incoming non-MPLS packets, include the `exp-push-push-push` default statement at the `[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules]` hierarchy level:

```
[edit class-of-service interfaces interface-name unit logical-unit-number rewrite-rules]
exp-push-push-push default;
```

These configurations apply the default MPLS EXP rewrite table, as described in ["Rewriting MPLS and IPv4 Packet Headers" on page 553](#). You can configure these operations and override the default MPLS EXP rewrite table with a custom table. For more information about writing and applying a custom rewrite table, see ["Configuring Rewrite Rules" on page 538](#) and ["Applying Rewrite Rules to Output Logical Interfaces" on page 550](#).



**NOTE:** With a three-label stack, if you do not include the `exp-swap-push-push` default or `exp-push-push-push` default statement in the configuration, the top label's EXP bits are set to zero.

## Example: Rewriting the EXP Bits of All Three Labels of an Outgoing Packet

Configure a swap-push-push operation, and override the default rewrite table with a custom table:

```
[edit class-of-service]
forwarding-classes {
    queue 0 be;
    queue 1 ef;
    queue 2 af;
    queue 3 nc;
}
interfaces {
    so-1/1/3 {
        unit 0 {
```



```

        rewrite-rules {
            exp exp_rew; # Apply custom rewrite table
            exp-swap-push-push default;
        }
    }
}

rewrite-rules {
    exp exp_rew {
        forwarding-class be {
            loss-priority low code-point 000;
            loss-priority high code-point 100;
        }
        forwarding-class ef {
            loss-priority low code-point 001;
            loss-priority high code-point 101;
        }
        forwarding-class af {
            loss-priority low code-point 010;
            loss-priority high code-point 110;
        }
        forwarding-class nc {
            loss-priority low code-point 011;
            loss-priority high code-point 111;
        }
    }
}

```

## RELATED DOCUMENTATION

[Rewriting MPLS and IPv4 Packet Headers | 553](#)

[Configuring Rewrite Rules | 538](#)

[Applying Rewrite Rules to Output Logical Interfaces | 550](#)

## 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

### IN THIS SECTION

- [Requirements | 564](#)
- [Overview | 564](#)
- [Configuration | 566](#)
- [Verification | 575](#)

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](http://www.juniper.net/books).

## Requirements

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

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

## Overview

IN THIS SECTION

Topology | 565

The purpose of this example is to demonstrate CoS rewriting at a network border to convey the traffics's CoS profile to the next-hop router, based on the forwarding class and packet loss priority (PLP) assigned to the traffic. CoS information rewriting is performed as the last step before a packet is transmitted onto the egress network.

In this example the rewriting is done when sending traffic from the host connected to Device R1 to the host connected to Device R2. The information required for rewriting the CoS parameters in the other direction is not included in this example. However, you can use the rewriting information in Device R1 (making changes for the interfaces used) and apply it to Device R2 to achieve bidirectional CoS rewriting.

Junos OS contains several default rewrite rules that might meet your requirements. You display them with the `show class-of-service rewrite-rule` command. [Table 56 on page 564](#) shows a partial list of the default rewrite rule mapping.

**Table 56: Partial list of the default rewrite rule mapping**

Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
expedited-forwarding	low	ef
expedited-forwarding	high	ef



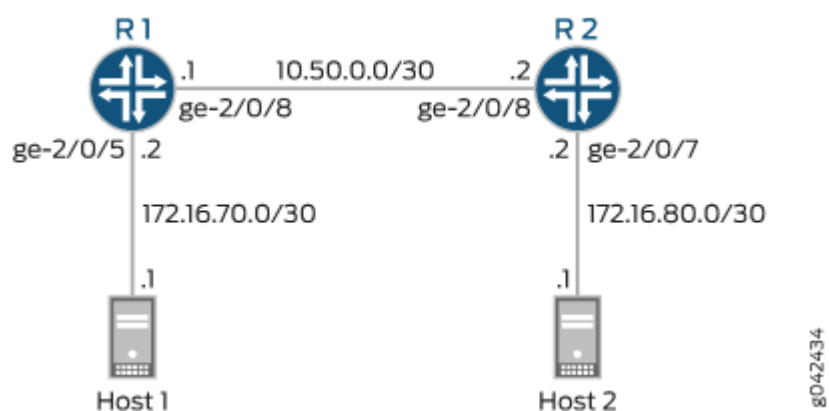
**Table 56: Partial list of the default rewrite rule mapping (*Continued*)**

Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
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 44 on page 565](#).

**Figure 44: Rewriting CoS Information at the Network Border to Enforce CoS Strategies Scenario**



This video explains the topics used in this example. We recommend that you watch the video before proceeding.



Video: [Learning Bytes CoS Remarking Video](#).

## Configuration

### IN THIS SECTION

- [Procedure](#) | 566

## Procedure

### CLI Quick Configuration

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

#### Device R1

```
set interfaces ge-2/0/5 description to-Host
set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
set interfaces ge-2/0/8 description to-R2
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces lo0 unit 0 description 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 [Junos OS CLI User Guide](#).

To configure Device R1:

1. Configure the device interfaces.

```
[edit ]
user@R1# set interfaces ge-2/0/5 description to-Host
user@R1# set interfaces ge-2/0/5 unit 0 family inet address 172.16.70.2/30
user@R1# set interfaces ge-2/0/5 unit 0 family inet filter input mf-classifier
user@R1# set interfaces ge-2/0/8 description to-R2
user@R1# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@R1# set interfaces lo0 unit 0 description loopback-interface
user@R1# set interfaces lo0 unit 0 family inet address 192.168.13.1/32
```

2. Configure the firewall parameters.

```
[edit ]
user@R1# set firewall family inet filter mf-classifier term BE-data from protocol tcp
user@R1# set firewall family inet filter mf-classifier term BE-data from port 80
user@R1# set firewall family inet filter mf-classifier term BE-data then count BE-data
user@R1# set firewall family inet filter mf-classifier term BE-data then forwarding-class BE-
data
user@R1# set firewall family inet filter mf-classifier term Prem-data from protocol tcp
user@R1# set firewall family inet filter mf-classifier term Prem-data from port 12345
user@R1# set firewall family inet filter mf-classifier term Prem-data then count Prem-data
user@R1# set firewall family inet filter mf-classifier term Prem-data then forwarding-class
Premium-data
user@R1# set firewall family inet filter mf-classifier term accept then accept
```

3. Configure the class-of-service parameters.

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



```

user@R1# set class-of-service interfaces ge-2/0/8 scheduler-map test-map
user@R1# set class-of-service interfaces ge-2/0/8 unit 0 rewrite-rules dscp IPv4-rewrite-table
user@R1# set class-of-service rewrite-rules dscp IPv4-rewrite-table forwarding-class BE-data
loss-priority low code-point be
user@R1# set class-of-service rewrite-rules dscp IPv4-rewrite-table forwarding-class Premium-
data loss-priority low code-point ef
user@R1# set class-of-service scheduler-maps test-map forwarding-class BE-data scheduler BE-
data
user@R1# set class-of-service scheduler-maps test-map forwarding-class Premium-data scheduler
Prem-data
user@R1# set class-of-service schedulers BE-data transmit-rate 1m
user@R1# set class-of-service schedulers BE-data buffer-size percent 25
user@R1# set class-of-service schedulers BE-data priority low
user@R1# set class-of-service schedulers Prem-data transmit-rate 1m
user@R1# set class-of-service schedulers Prem-data buffer-size percent 25
user@R1# set class-of-service schedulers Prem-data priority high

```

#### 4. Configure OSPF.

```

[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/5.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0

```

### Step-by-Step Procedure

To configure Device R2:

#### 1. Configure the device interface.

```

[edit ]
user@R1# set interfaces ge-2/0/7 description to-Host
user@R1# set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.1/30
user@R1# set interfaces ge-2/0/8 description to-R1
user@R1# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@R2# set interfaces ge-2/0/8 unit 0 family inet filter input mf-classifier
user@R1# set interfaces unit 0 description looback-interface
user@R1# set interfaces unit 0 family inet address 192.168.14.1/32

```



## 2. Configure the firewall parameters.

```
[edit ]
user@R2# set firewall family inet filter mf-classifier term BE-data from dscp be
user@R2# set firewall family inet filter mf-classifier term BE-data then count BE-data
user@R2# set firewall family inet filter mf-classifier term Premium-data from dscp ef
user@R2# set firewall family inet filter mf-classifier term Premium-data then count Premium-
data
user@R2# set firewall family inet filter mf-classifier term accept then accept
```

## 3. Configure OSPF.

```
[edit protocols ospf]
user@R1# set area 0.0.0.0 interface ge-2/0/7.0 passive
user@R1# set area 0.0.0.0 interface lo0.0 passive
user@R1# set area 0.0.0.0 interface ge-2/0/8.0
```

## Results

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

```
user@R1 show interfaces
  ge-2/0/5 {
    description to-Host;
    unit 0 {
      family inet {
        filter {
          input mf-classifier;
        }
        address 172.16.70.2/30;
      }
    }
  }
  ge-2/0/8 {
    description to-R2;
    unit 0 {
      family inet {
```



```

        address 10.50.0.1/30;
    }
}
lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
}

```

```

user@R1 show firewall
family inet {
    filter mf-classifier {
        term BE-data {
            from {
                protocol tcp;
                port 80;
            }
            then {
                count BE-data;
                forwarding-class BE-data;
            }
        }
        term Prem-data {
            from {
                protocol tcp;
                port 12345;
            }
            then {
                count Prem-data;
                forwarding-class Premium-data;
            }
        }
        term accept {
            then accept;
        }
    }
}

```



```

    }
}

```

```

user@R1 show class-of-service
forwarding-classes {
    queue 0 BE-data;
    queue 1 Premium-data;
    queue 2 voice;
    queue 3 NC;
}
interfaces {
    ge-2/0/8 {
        scheduler-map test-map;
        unit 0 {
            rewrite-rules {
                dscp IPv4-rewrite-table;
            }
        }
    }
}
rewrite-rules {
    dscp IPv4-rewrite-table {
        forwarding-class BE-data {
            loss-priority low code-point be;
        }
        forwarding-class Premium-data {
            loss-priority low code-point ef;
        }
    }
}
scheduler-maps {
    test-map {
        forwarding-class BE-data scheduler BE-data;
        forwarding-class Premium-data scheduler Prem-data;
    }
}
schedulers {
    BE-data {
        transmit-rate 1m;
        buffer-size percent 25;
        priority low;
    }
}

```



```

    }
    Prem-data {
        transmit-rate 1m;
        buffer-size percent 25;
        priority high;
    }
}

```

```

user@R1# show protocols ospf
area 0.0.0.0 {
    interface ge-2/0/5.0 {
        passive;
    }
    interface lo0.0 {
        passive;
    }
    interface ge-2/0/8.0;
}

```

If you are done configuring Device R1, enter `commit` from configuration mode.

```

user@R2# show interfaces
ge-2/0/7 {
    unit 0 {
        description to-Host;
        family inet {
            address 172.16.80.2;
        }
    }
}
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            filter {
                input mf-classifier;
            }
            address 10.50.0.2/30;
        }
    }
}
}

```



```

lo0 {
    unit 0 {
        description looback-interface;
        family inet {
            address 192.168.14.1/32;
        }
    }
}

```

```

user@R2# show firewall
family inet {
    filter mf-classifier {
        term BE-data {
            from {
                dscp be;
            }
            then count BE-data;
        }
        term Premium-data {
            from {
                dscp ef;
            }
            then count Premium-data;
        }
        term accept {
            then accept;
        }
    }
}

```

```

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

```



```
interface ge-2/0/8.0;  
}
```

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

## Verification

### IN THIS SECTION

- [Clearing the Firewall Counters | 575](#)
- [Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Monitoring the Results | 575](#)

Confirm that the configuration is working properly.

### Clearing the Firewall Counters

#### Purpose

Confirm that the firewall counters are cleared.

#### Action

On Devices R1 and R2, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@R1> clear firewall all  
user@R2> clear firewall all
```

### Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Monitoring the Results

#### Purpose

Send traffic from the host connected to Device 1 into the network so that it can be monitored by the firewall on Device R1 and Device R2.



## Action

1. Use a traffic generator to send 20 TCP packets with a source port of 80 into the network.

The `-s` flag sets the source port. The `-k` flag causes the source port to remain steady at 80 instead of incrementing. The `-c` flag sets the number of packets to 20. The `-d` flag sets the packet size.

```
[User@host]# hping 172.16.80.1 -c 20 -s 80 -k -d 300
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.9 ms
.
.
.
--- 172.16.80.1 hping statistic ---
20 packets transmitted, 20 packets received, 0% packet loss
round-trip min/avg/max = 0.9/9501.4/19002.4 ms
```

2. Use a traffic generator to send 20 TCP packets with a source port of 12345 into the network.

```
[User@host]# hping 172.16.80.1 -c 20 -s 12345 -k -d 300
HPING 172.16.80.1 (eth1 172.16.80.1): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=172.16.80.1 ttl=62 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.3 ms
.
.
.
--- 172.16.80.1 hping statistic ---
20 packets transmitted, 20 packets received, 0% packet loss
round-trip min/avg/max = 0.3/9501.5/19002.7 ms
```

3. On Device R1, check the firewall counters by using the `show firewall` command.

```
user@R1> show firewall
Filter: mf-classifier
Counters:


| Name      | Bytes | Packets |
|-----------|-------|---------|
| BE-data   | 800   | 20      |
| Prem-data | 800   | 20      |


```



4. On Device R2, check the firewall counters using the `show firewall` command.

```
user@R2> show firewall
Filter: mf-classifier
Counters:
Name                               Bytes      Packets
BE-data                            800         20
Premium-data                        800         20
```

## Meaning

Device R1 correctly set the code point for TCP packets to port 12345 to bf. Device R1 correctly set the code point for TCP packets to port 80 to ef. Device R2 correctly recognized the code point for TCP packets to port 12345 as bf. Device R2 correctly recognized the code point for TCP packets to port 80 as ef.

## RELATED DOCUMENTATION

[Example: Configuring and Applying Scheduler Maps | 351](#)

## Example: Remarking Diffserv Code Points to MPLS EXPs to Carry CoS Profiles Across a Service Provider's L3VPN MPLS Network

### IN THIS SECTION

- [Requirements | 578](#)
- [Overview | 578](#)
- [Configuration | 581](#)
- [Verification | 607](#)

This example is an introduction in how to rewrite (remark) DSCP class-of-service (CoS) code point values at the network border of a customer network and a service provider's MPLS network while maintaining



the original CoS profile of the traffic so that the traffic can be remarked with the original DSCP code points when it exits the MPLS network.

Requirements

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

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

Overview

IN THIS SECTION

Topology | 580

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 46 on page 581](#).

Junos OS contains several DSCP default rewrite rules that might meet your requirements. You display them with the `show class-of-service rewrite-rule` command. A partial set of the default rewrite DSCP code point rule mappings is shown in the following table.

You can also define your own custom rewrite-rules table, or use a mixture of the default rewrite-rules and a custom table that you create. This example uses default rewrite-rules.

Table 57: Default rewrite rules

Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
expedited-forwarding	low	ef
expedited-forwarding	high	ef
assured-forwarding	low	af11
assured-forwarding	high	af12 (DSCP/DSCP IPv6/EXP)



**Table 57: Default rewrite rules (Continued)**

Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
best-effort	low	be
best-effort	high	be
network-control	low	nc1/cs6
network-control	high	nc2/cs7

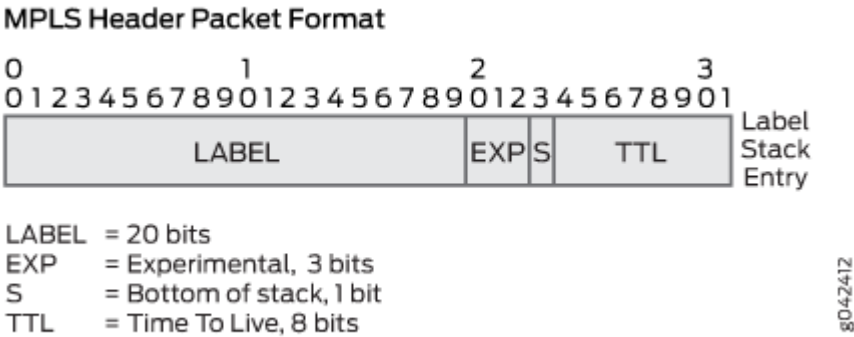
Junos OS uses the values shown in the following table for MPLS CoS in the EXP fields of the MPLS header.

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 45 on page 580 shows the MPLS packet structure.



Figure 45: 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 46 on page 581](#). A full explanation of the tasks required to configure an L3VPN network is not included in this example. If you require more information regarding configuring an L3VPN network, refer to the *Layer 3 VPNs User Guide for Routing Devices* available at <http://juniper.net/documentation>.

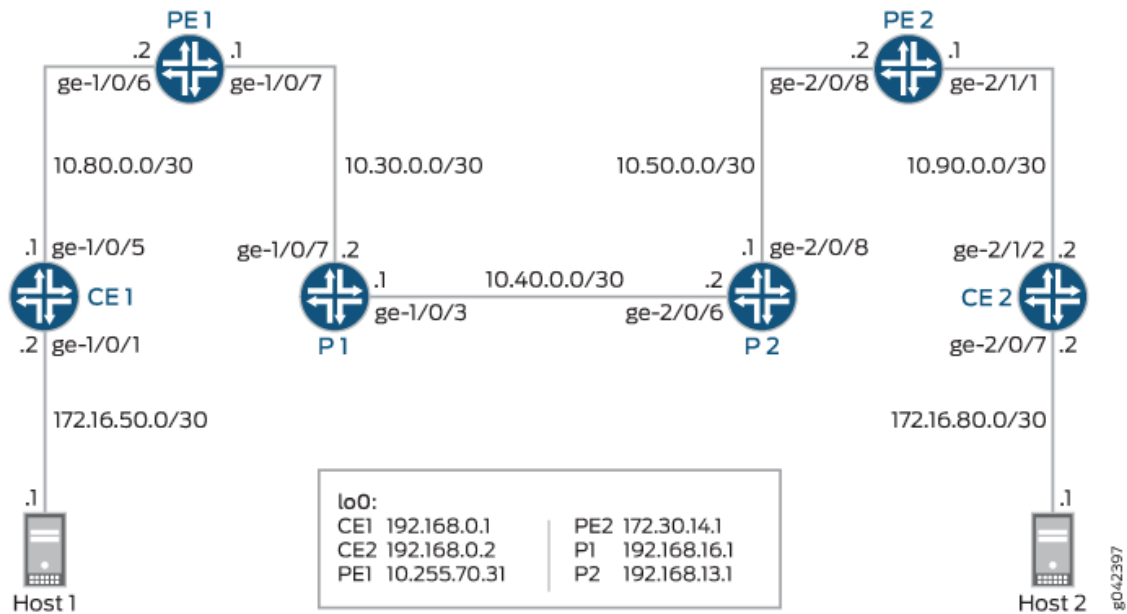
A thorough explanation of the required CoS rewriting and the underlying algorithms used in this example is beyond the scope of this document. For more information, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at [www.juniper.net/books](http://www.juniper.net/books).

### Topology

This example uses the topology in [Figure 46 on page 581](#).



Figure 46: Rewriting CoS Information at the Network Border to Transit an MPLS Network Scenario



## Configuration

### IN THIS SECTION

- Procedure | 581

## Procedure

### CLI Quick Configuration

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

#### Device CE1

```
set interfaces ge-1/0/1 unit 0 description to-host
set interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
set interfaces ge-1/0/1 unit 0 family inet filter input ip-v4
```



```

set interfaces ge-1/0/5 unit 0 description to_Provider
set interfaces ge-1/0/5 unit 0 family inet address 10.80.0.1/30
set interfaces lo0 unit 1 description loopback-interface
set interfaces lo0 unit 1 family inet address 192.168.0.1/32
set protocols bgp group to_Provider type external
set protocols bgp group to_Provider export send-direct
set protocols bgp group to_Provider peer-as 64511
set protocols bgp group to_Provider neighbor 10.80.0.2
set policy-options policy-statement send-direct from protocol direct
set policy-options policy-statement send-direct then accept
set routing-options router-id 192.168.0.1
set routing-options autonomous-system 64510
set firewall family inet filter ip-v4 term tcp80 from port 80
set firewall family inet filter ip-v4 term tcp80 then dscp ef
set firewall family inet filter ip-v4 term 12345 from port 12345
set firewall family inet filter ip-v4 term 12345 then dscp be
set firewall family inet filter ip-v4 term accept then accept

```

## Device PE1

```

set interfaces ge-1/0/6 description to_vpna
set interfaces ge-1/0/6 unit 0 family inet address 10.80.0.2/30
set interfaces ge-1/0/7 description to_P1
set interfaces ge-1/0/7 unit 0 family inet address 10.30.0.1/30
set interfaces ge-1/0/7 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 10.255.70.31/32
set routing-options router-id 10.255.70.31
set routing-options autonomous-system 64511
set protocols mpls interface ge-1/0/7.0
set protocols bgp group to_PE2 type internal
set protocols bgp group to_PE2 local-address 10.255.70.31
set protocols bgp group to_PE2 family inet-vpn unicast
set protocols bgp group to_PE2 neighbor 172.30.14.1
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/7.0
set protocols ldp interface ge-1/0/7.0
set protocols ldp interface lo0.0
set routing-instances vpna instance-type vrf
set routing-instances vpna interface ge-1/0/6.0
set routing-instances vpna route-distinguisher 64511:1
set routing-instances vpna vrf-target target:64511:1

```



```

set routing-instances vpna protocols bgp group to_vpna type external
set routing-instances vpna protocols bgp group to_vpna peer-as 64510
set routing-instances vpna protocols bgp group to_vpna neighbor 10.80.0.1
set class-of-service classifiers dscp 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-1/0/6 unit 0 classifiers dscp dscp4
set class-of-service interfaces ge-1/0/6 unit 0 rewrite-rules dscp dscp4-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 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 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 [Junos OS CLI User Guide](#).

To configure Device CE1:

1. Configure the device interfaces.

```

[edit ]
user@CE1# set interfaces ge-1/0/1 unit 0 description to-host
user@CE1# set interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
user@CE1# set interfaces ge-1/0/1 unit 0 family inet filter input ip-v4
user@CE1# set interfaces ge-1/0/5 unit 0 description to_Provider
user@CE1# set interfaces ge-1/0/5 unit 0 family inet address 10.80.0.1/30
user@CE1# set interfaces lo0 unit 1 description loopback-interface
user@CE1# set interfaces lo0 unit 1 family inet address 192.168.0.1/32

```

2. Configure the BGP parameters

```

[edit ]
user@CE1# set protocols bgp group to_Provider type external
user@CE1# set protocols bgp group to_Provider export send-direct
user@CE1# set protocols bgp group to_Provider peer-as 64511
user@CE1# set protocols bgp group to_Provider neighbor 10.80.0.2

```



### 3. Configure the policy option parameters.

```
[edit ]
user@CE1# set policy-options policy-statement send-direct from protocol direct
user@CE1# set policy-options policy-statement send-direct then accept
```

### 4. Configure the routing option parameters.

```
[edit ]
user@CE1# set routing-options router-id 192.168.0.1
user@CE1# set routing-options autonomous-system 64510
```

### 5. Configure the DSCP code point rewrite parameters.

```
[edit ]
user@CE1# set firewall family inet filter ip-v4 term tcp80 from port 80
user@CE1# set firewall family inet filter ip-v4 term tcp80 then dscp ef
user@CE1# set firewall family inet filter ip-v4 term 12345 from port 12345
user@CE1# set firewall family inet filter ip-v4 term 12345 then dscp be
user@CE1# set firewall family inet filter ip-v4 term accept then accept
```

## Step-by-Step Procedure

To configure Device PE1:

### 1. Configure the device interfaces.

```
[edit ]
user@PE1# set interfaces ge-1/0/6 description to_vpna
user@PE1# set interfaces ge-1/0/6 unit 0 family inet address 10.80.0.2/30
user@PE1# set interfaces ge-1/0/7 description to_P1
user@PE1# set interfaces ge-1/0/7 unit 0 family inet address 10.30.0.1/30
user@PE1# set interfaces ge-1/0/7 unit 0 family mpls
user@PE1# set interfaces lo0 unit 0 description loopback-interface
user@PE1# set interfaces lo0 unit 0 family inet address 10.255.70.31/32
```



## 2. Configure the routing option parameters.

```
[edit ]
user@PE1# set routing-options router-id 10.255.70.31
user@PE1# set routing-options autonomous-system 64511
```

## 3. Configure the protocol parameters.

```
user@PE1# set protocols mpls interface ge-1/0/7.0
user@PE1# set protocols bgp group to_PE2 type internal
user@PE1# set protocols bgp group to_PE2 local-address 10.255.70.31
user@PE1# set protocols bgp group to_PE2 family inet-vpn unicast
user@PE1# set protocols bgp group to_PE2 neighbor 172.30.14.1
user@PE1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@PE1# set protocols ospf area 0.0.0.0 interface ge-1/0/7.0
user@PE1# set protocols ldp interface ge-1/0/7.0
user@PE1# set protocols ldp interface lo0.0
```

## 4. Configure the routing instance parameters.

```
[edit ]
user@PE1# set routing-instances vpna instance-type vrf
user@PE1# set routing-instances vpna interface ge-1/0/6.0
user@PE1# set routing-instances vpna route-distinguisher 64511:1
user@PE1# set routing-instances vpna vrf-target target:64511:1
user@PE1# set routing-instances vpna protocols bgp group to_vpna type external
user@PE1# set routing-instances vpna protocols bgp group to_vpna peer-as 64510
user@PE1# set routing-instances vpna protocols bgp group to_vpna neighbor 10.80.0.1
```

## 5. Configure the class-of-service parameters that perform the DSCP code point to MPLS EXP rewriting.

```
user@PE1# set class-of-service classifiers dscp dscpv4 forwarding-class expedited-forwarding
loss-priority low code-points ef
user@PE1# set class-of-service classifiers dscp dscpv4 forwarding-class best-effort loss-
priority low code-points be
user@PE1# set class-of-service classifiers exp exp-in forwarding-class expedited-forwarding
loss-priority low code-points 010
user@PE1# set class-of-service classifiers exp exp-in forwarding-class best-effort loss-
priority low code-points 000
```



```

user@PE1# set class-of-service interfaces ge-1/0/6 unit 0 classifiers dscp dscpv4
user@PE1# set class-of-service interfaces ge-1/0/6 unit 0 rewrite-rules dscp dscpv4-rw
user@PE1# set class-of-service interfaces ge-1/0/7 unit 0 classifiers exp exp-in
user@PE1# set class-of-service interfaces ge-1/0/7 unit 0 rewrite-rules exp exp-out
user@PE1# set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class expedited-
forwarding loss-priority low code-point ef
user@PE1# set class-of-service rewrite-rules dscp dscpv4-rw forwarding-class best-effort loss-
priority low code-point be
user@PE1# set class-of-service rewrite-rules exp exp-out forwarding-class expedited-
forwarding loss-priority low code-point 010
user@PE1# set class-of-service rewrite-rules exp exp-out forwarding-class best-effort loss-
priority low code-point 000

```

## Step-by-Step Procedure

To configure Device P1:

1. Configure the device interfaces.

```

[edit ]
user@P1# set interfaces ge-1/0/3 description to_P2
user@P1# set interfaces ge-1/0/3 unit 0 family inet address 10.40.0.1/30
user@P1# set interfaces ge-1/0/3 unit 0 family mpls
user@P1# set interfaces ge-1/0/7 description to_PE1
user@P1# set interfaces ge-1/0/7 unit 0 family inet address 10.30.0.2/30
user@P1# set interfaces ge-1/0/7 unit 0 family mpls
user@P1# set interfaces lo0 unit 0 description loopback-interface
user@P1# set interfaces lo0 unit 0 family inet address 192.168.16.1/32

```

2. Configure the routing option parameters.

```

[edit ]
user@P1# set routing-options router-id 10.255.187.32

```

3. Configure the protocol parameters.

```

[edit ]
user@P1# set protocols mpls interface ge-1/0/7.0
user@P1# set protocols mpls interface ge-1/0/3.0
user@P1# set protocols ospf area 0.0.0.0 interface ge-1/0/3.0

```



```

user@P1# set protocols ospf area 0.0.0.0 interface ge-1/0/7.0
user@P1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@P1# set protocols ldp interface ge-1/0/3.0
user@P1# set protocols ldp interface ge-1/0/7.0
user@P1# set protocols ldp interface lo0.0

```

## Step-by-Step Procedure

To configure Device P2:

1. Configure the device interfaces.

```

[edit ]
user@P2# set interfaces ge-2/0/6 description to_P1
user@P2# set interfaces ge-2/0/6 unit 0 family inet address 10.40.0.2/30
user@P2# set interfaces ge-2/0/6 unit 0 family mpls
user@P2# set interfaces ge-2/0/8 description to_PE2
user@P2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@P2# set interfaces ge-2/0/8 unit 0 family mpls
user@P2# set interfaces lo0 unit 0 description loopback-interface
user@P2# set interfaces lo0 unit 0 family inet address 192.168.13.1/32

```

2. Configure the routing option parameters.

```

[edit ]
user@P2# set routing-options router-id 192.168.187.3

```

3. Configure the protocol parameters.

```

[edit ]
user@P2# set protocols mpls interface ge-2/0/6.0
user@P2# set protocols mpls interface ge-2/0/8.0
user@P2# set protocols ospf area 0.0.0.0 interface ge-2/0/6.0
user@P2# set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
user@P2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@P2# set protocols ldp interface ge-2/0/6.0
user@P2# set protocols ldp interface ge-2/0/8.0
user@P2# set protocols ldp interface lo0.0

```



## Step-by-Step Procedure

To configure Device PE2:

1. Configure the device interfaces.

```
[edit ]
user@PE2# set interfaces ge-2/0/8 description to-R1
user@PE2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@PE2# set interfaces ge-2/0/8 unit 0 family mpls
user@PE2# set interfaces ge-2/1/1 unit 0 description to-vpna
user@PE2# set interfaces ge-2/1/1 unit 0 family inet address 10.90.0.1/30
user@PE2# set interfaces lo0 unit 0 description loopback-interface
user@PE2# set interfaces lo0 unit 0 family inet address 172.30.14.1/32
```

2. Configure the routing option parameters.

```
[edit ]
user@PE2# set routing-options router-id 172.30.14.1
user@PE2# set routing-options autonomous-system 64511
```

3. Configure the protocol parameters.

```
[edit ]
user@PE2# set protocols mpls interface ge-2/0/8.0
user@PE2# set protocols bgp group to_PE2 type internal
user@PE2# set protocols bgp group to_PE2 local-address 172.30.14.1
user@PE2# set protocols bgp group to_PE2 family inet-vpn unicast
user@PE2# set protocols bgp group to_PE2 neighbor 10.255.70.31
user@PE2# set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
user@PE2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@PE2# set protocols ldp interface ge-2/0/8.0
user@PE2# set protocols ldp interface lo0.0
```

4. Configure the routing instance parameters.

```
[edit ]
user@PE2# set routing-instances vpna instance-type vrf
user@PE2# set routing-instances vpna interface ge-2/1/1.0
```



```

user@PE2# set routing-instances vpna route-distinguisher 64511:1
user@PE2# set routing-instances vpna vrf-target target:64511:1
user@PE2# set routing-instances vpna protocols bgp group to_vpna type external
user@PE2# set routing-instances vpna protocols bgp group to_vpna peer-as 64512
user@PE2# set routing-instances vpna protocols bgp group to_vpna neighbor 10.90.0.2

```

5. Configure the class-of-service parameters that perform the DSCP code point to MPLS EXP rewriting.

```

[edit ]
user@PE2# set class-of-service classifiers dscp 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 dscpv4 {
        forwarding-class expedited-forwarding {
            loss-priority low code-points ef;
        }
        forwarding-class best-effort {

```



```

        loss-priority low code-points be;
    }
}
exp exp-in {
    forwarding-class expedited-forwarding {
        loss-priority low code-points 010;
    }
    forwarding-class best-effort {
        loss-priority low code-points 000;
    }
}
}
interfaces {
    ge-1/0/6 {
        unit 0 {
            classifiers {
                dscp dscpv4;
            }
            rewrite-rules {
                dscp dscpv4-rw;
            }
        }
    }
    ge-1/0/7 {
        unit 0 {
            classifiers {
                exp exp-in;
            }
            rewrite-rules {
                exp exp-out;
            }
        }
    }
}
rewrite-rules {
    dscp dscpv4-rw {
        forwarding-class expedited-forwarding {
            loss-priority low code-point ef;
        }
        forwarding-class best-effort {
            loss-priority low code-point be;
        }
    }
}

```



```

exp exp-out {
    forwarding-class expedited-forwarding {
        loss-priority low code-point 010;
    }
    forwarding-class best-effort {
        loss-priority low code-point 000;
    }
}
}

```

If you are done configuring Device PE1, enter `commit` from configuration mode.

```

user@P1# show interfaces
ge-1/0/3 {
    description to_P2;
    unit 0 {
        family inet {
            address 10.40.0.1/30;
        }
        family mpls;
    }
}
ge-1/0/7 {
    description to_PE1;
    unit 0 {
        family inet {
            address 10.30.0.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.16.1/32;
        }
    }
}

```



```

    }
}

```

```

user@P1# show protocols
mpls {
    interface ge-1/0/7.0;
    interface ge-1/0/3.0;
}
ospf {
    area 0.0.0.0 {
        interface ge-1/0/3.0;
        interface ge-1/0/7.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/3.0;
    interface ge-1/0/7.0;
    interface lo0.0;
}

```

```

user@P1# show routing-options
router-id 10.255.187.32;

```

If you are done configuring Device P1, enter `commit` from configuration mode.

```

user@P2# show interfaces
ge-2/0/6 {
    description to_P1;
    unit 0 {
        family inet {
            address 10.40.0.2/30;
        }
        family mpls;
    }
}
ge-2/0/8 {

```



```

description to_PE2;
unit 0 {
    family inet {
        address 10.50.0.1/30;
    }
    family mpls;
}
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}
}

```

```

user@P2# show protocols
mpls {
    interface ge-2/0/6.0;
    interface ge-2/0/8.0;
}
ospf {
    area 0.0.0.0 {
        interface ge-2/0/6.0;
        interface ge-2/0/8.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-2/0/6.0;
    interface ge-2/0/8.0;
    interface lo0.0;
}

```

```

user@P2# show routing-options
router-id 192.168.187.3;

```



If you are done configuring Device P2, enter `commit` from configuration mode.

```
user@PE2# show interfaces
```

```
ge-2/0/8 {
  description to-R1;
  unit 0 {
    family inet {
      address 10.50.0.2/30;
    }
    family mpls;
  }
}
ge-2/1/1 {
  unit 0 {
    description to-vpna;
    family inet {
      address 10.90.0.1/30;
    }
  }
}
lo0 {
  unit 0 {
    description loopback-interface;
    family inet {
      address 172.30.14.1/32;
    }
  }
}
```

```
user@PE2# show protocols
```

```
mpls {
  interface ge-2/0/8.0;
}
bgp {
  group to_PE1 {
    type internal;
    local-address 172.30.14.1;
    family inet-vpn {
```



```

        unicast;
    }
    neighbor 10.255.70.31;
}
}
ospf {
    area 0.0.0.0 {
        interface ge-2/0/8.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-2/0/8.0;
    interface lo0.0;
}

```

```

user@PE2# show routing-options
router-id 172.30.14.1;
autonomous-system 64511;

```

```

user@PE2# show routing-instances
vpna {
    instance-type vrf;
    interface ge-2/1/1.0;
    route-distinguisher 64511:1;
    vrf-target target:64511:1;
    protocols {
        bgp {
            group to_vpna {
                type external;
                peer-as 64512;
                neighbor 10.90.0.2;
            }
        }
    }
}

```



```

    }
}

```

user@PE2# **show class-of-service**

```

classifiers {
  dscp 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 dscpv4-rw {
    forwarding-class expedited-forwarding {
      loss-priority low code-point ef;
    }
    forwarding-class best-effort {
      loss-priority low code-point be;
    }
  }
}
exp exp-out {
  forwarding-class expedited-forwarding {
    loss-priority low code-point 010;
  }
  forwarding-class best-effort {
    loss-priority low code-point 000;
  }
}
}
}

```

If you are done configuring Device PE2, enter `commit` from configuration mode.

```

user@CE2# show interfaces
ge-2/0/7 {
  unit 0 {
    description to-host;
    family inet {
      filter {
        input ip-v4;
      }
      address 172.16.80.2/30;
    }
  }
}
ge-2/1/2 {
  unit 0 {
    description to-Provider;
    family inet {
      address 10.90.0.2/30;
    }
  }
}

```



```

    }
}
lo0 {
    unit 1 {
        description loopback-interface;
        family inet {
            address 192.168.0.2/32;
        }
    }
}
}

```

```

user@CE2# show protocols
bgp {
    group to_Provider {
        type external;
        export send-direct;
        peer-as 64511;
        neighbor 10.90.0.1;
    }
}

```

```

user@CE2# show policy-options
policy-statement send-direct {
    from protocol direct;
    then accept;
}

```

```

user@CE2# show routing-options
router-id 192.168.0.2;
autonomous-system 64512;

```

```

user@CE2# show firewall
family inet {
    filter ip-v4 {
        term tcp80 {
            from {
                port 80;
            }
        }
    }
}

```



```
    }  
    then dscp ef;  
  }  
  term 12345 {  
    from {  
      port 12345;  
    }  
    then dscp be;  
  }  
  term accept {  
    then accept;  
  }  
}  
}
```

If you are done configuring Device CE2, enter `commit` from configuration mode.

## Verification

### IN THIS SECTION

- [Clearing the Firewall Counters | 607](#)
- [Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Monitoring the Results | 608](#)

Confirm that the configuration is working properly by verifying that the DSCP code points are maintained from CE1 to CE2.

### Clearing the Firewall Counters

#### Purpose

Confirm that the firewall counters are cleared.



## Action

On Device CE2, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@CE2> clear firewall all
```

## Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Monitoring the Results

### Purpose

Send traffic into the network from the host connected to Device CE1 so that it that can be monitored at Device CE2.

### Action

A different firewall is required on interface `ge-2/0/7` to count the traffic that is being transmitted outbound to the destination. The following commands apply the firewall filter that counts the marked traffic as it is transmitted to the destination.



**NOTE:** To capture traffic at Device CE1, apply this command `set interfaces ge-1/0/1 unit 0 family inet filter output count`, followed by the commands below.



**NOTE:** To capture traffic at Device CE2, apply this command `set interfaces ge-2/0/7 unit 0 family inet filter output count`, followed by the commands below.

```
set firewall family inet filter count term be from dscp be
set firewall family inet filter count term be then count be
set firewall family inet filter count term ef from dscp ef
set firewall family inet filter count term ef then count ef
set firewall family inet filter count term accept then accept
set interfaces ge-2/0/7 unit 0 family inet filter output count
```

When you are done testing, you can leave the counting filter in place, or remove it.



1. On host 1 use a traffic generator to send 20 TCP packets with a source port of 80 into the network, and repeat the task using a source port of 12345.

```
[user@host]# hping 172.16.80.1 -s 80 -k -c 20
[user@host]# hping 172.16.80.1 -s 12345 -k -c 20
```

2. On Device CE2, check the firewall counters by using the show firewall command.

```
user@CE2> show firewall

Filter: __CE2/ip-v4

Filter: __CE2/count
Counters:

```

Name	Bytes	Packets
be	800	20
ef	800	20

## Meaning

The code point for TCP packets to port 12345 is maintained as be. The code point for TCP packets to port 80 is maintained as ef.

## RELATED DOCUMENTATION

[Example: Configuring and Applying Scheduler Maps | 351](#)

## Example: Remarking Diffserv Code Points to 802.1P PCPs to Carry CoS Profiles Across a Service Provider's VPLS Network

### IN THIS SECTION

- [Requirements | 610](#)
- [Overview | 610](#)



- Configuration | 614
- Verification | 638

This configuration example explains how to implement class-of-service (CoS) capabilities over a Virtual Private LAN Service (VPLS) network.

## Requirements

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

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

## Overview

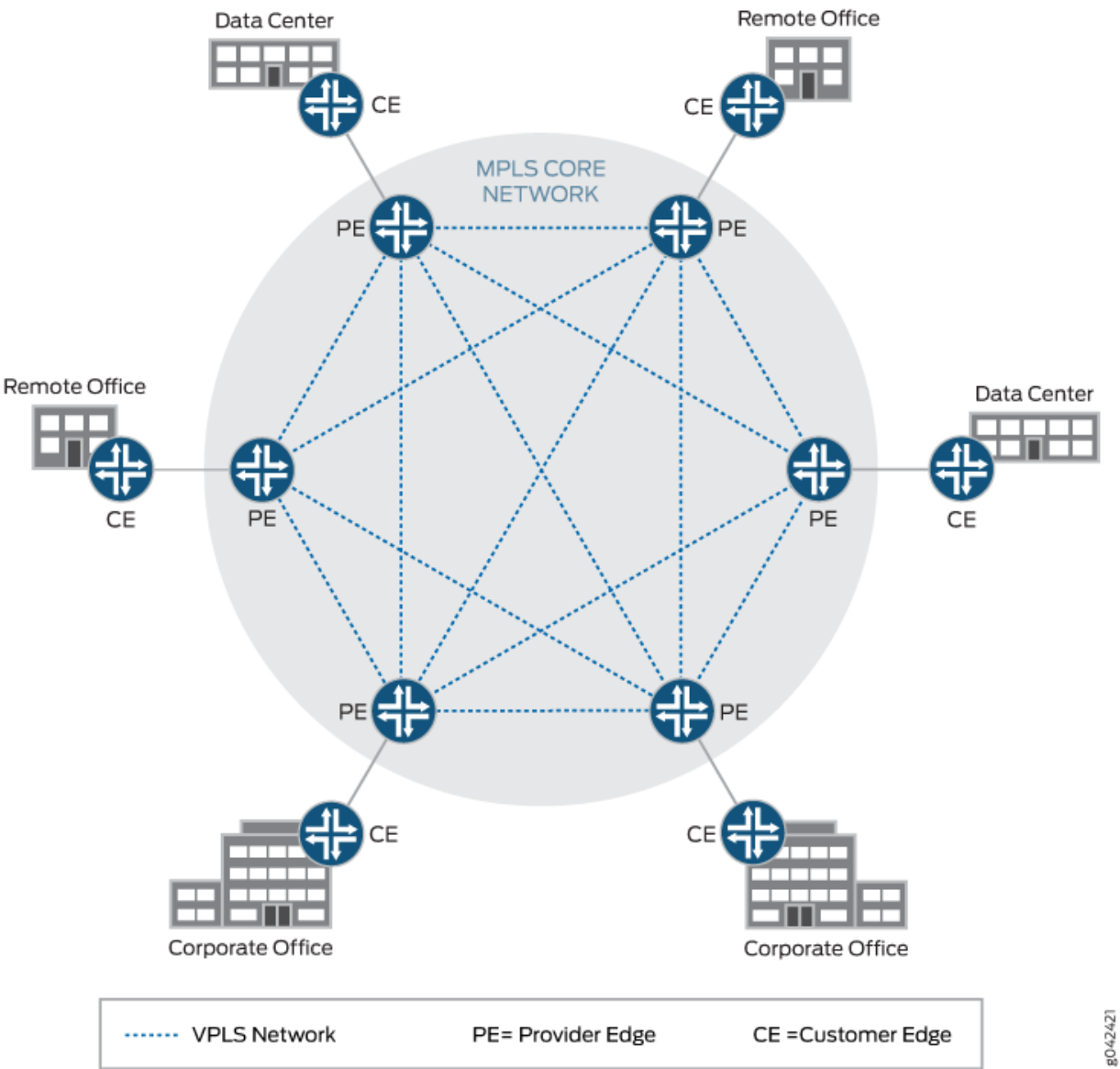
### IN THIS SECTION

- Topology | 613

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 47 on page 611](#) for an example of a typical VPLS topology.



Figure 47: Typical VPLS Topology



8042421

Junos OS contains several DiffServ code point (DSCP) default rewrite rules that might meet your requirements. You display them with the `show class-of-service rewrite-rule` command. A partial set of the default rewrite DSCP rule mappings is shown in the following table.

You can also define your own custom rewrite-rules table, or use a mixture of the default rewrite-rules and a custom table that you create. This example uses default rewrite-rules.



**Table 58: Default rewrite rules**

Map from Forwarding Class	PLP Value	MAP to DSCP/DSCP IPv6/EXP/IP Code Point Aliases
expedited-forwarding	low	ef
expedited-forwarding	high	ef
assured-forwarding	low	af11
assured-forwarding	high	af12 (DSCP/DSCP IPv6/EXP)
best-effort	low	be
best-effort	high	be
network-control	low	nc1/cs6
network-control	high	nc2/cs7

Junos OS uses the values shown in the following table for MPLS CoS in the EXP fields of the MPLS header.

Forwarding Class	Loss Priority	EXP Code Point
best-effort	low	000
best-effort	high	001
expedited-forwarding	low	010
expedited-forwarding	high	011
assured-forwarding	low	100



(Continued)

Forwarding Class	Loss Priority	EXP Code Point
assured-forwarding	high	101
network-control	low	110
network-control	high	111



**NOTE:** In addition to providing the necessary information to complete the purpose of this example, this example also includes all of the commands required to recreate the VPLS network as shown in [Figure 48 on page 614](#). A full explanation of the tasks required to configure a VPLS network is not included in this example. If you need more information regarding configuring a VPLS network, see the *VPLS User Guide for Routing Devices* at <http://juniper.net/documentation> and RFC 4761 at <http://tools.ietf.org/html/rfc4761>.

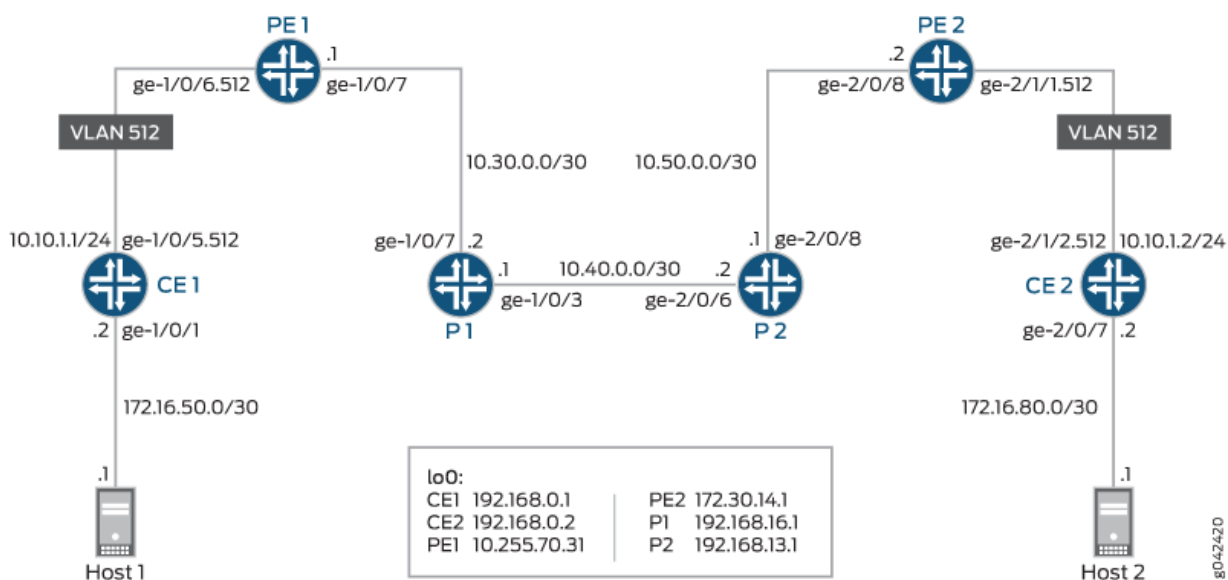
A thorough explanation of the required CoS tasks and the underlying algorithms used in this example is beyond the scope of this document. For more information, refer to *QOS-Enabled Networks—Tools and Foundations* by Miguel Barreiros and Peter Lundqvist. This book is available at many online booksellers and at [www.juniper.net/books](http://www.juniper.net/books).

## Topology

This example uses the topology in [Figure 48 on page 614](#).



Figure 48: VPLS with CoS Scenario



## Configuration

### IN THIS SECTION

- Procedure | 614

## Procedure

### CLI Quick Configuration

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

#### Device CE1

```
set interfaces ge-1/0/1 unit 0 description to-Host1
set interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
set interfaces ge-1/0/1 unit 0 family inet filter input ip-v4
set interfaces ge-1/0/5 vlan-tagging
```



```

set interfaces ge-1/0/5 unit 512 description to_PE1
set interfaces ge-1/0/5 unit 512 vlan-id 512
set interfaces ge-1/0/5 unit 512 family inet address 10.10.1.1/24
set interfaces lo0 unit 1 description loopback-interface
set interfaces lo0 unit 1 family inet address 192.168.0.1/32
set protocols ospf area 0.0.0.0 interface ge-1/0/5.512
set protocols ospf area 0.0.0.0 interface ge-1/0/1.0 passive
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set firewall family inet filter ip-v4 term tcp80 from port 80
set firewall family inet filter ip-v4 term tcp80 then dscp ef
set firewall family inet filter ip-v4 term 12345 from port 12345
set firewall family inet filter ip-v4 term 12345 then dscp be
set firewall family inet filter ip-v4 term accept then accept
set class-of-service classifiers ieee-802.1 dscp1 forwarding-class expedited-forwarding loss-
priority low code-points ef
set class-of-service classifiers ieee-802.1 dscp1 forwarding-class best-effort loss-priority low
code-points be
set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class expedited-forwarding
loss-priority low code-point 010
set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class best-effort loss-
priority low code-point 000
set class-of-service interfaces ge-1/0/5 unit 512 classifiers ieee-802.1 dscp1
set class-of-service interfaces ge-1/0/5 unit 512 rewrite-rules ieee-802.1 ieee1-c2

```

## Device PE1

```

set interfaces ge-1/0/6 vlan-tagging
set interfaces ge-1/0/6 encapsulation vlan-vpls
set interfaces ge-1/0/6 unit 512 description to_vpls
set interfaces ge-1/0/6 unit 512 encapsulation vlan-vpls
set interfaces ge-1/0/6 unit 512 vlan-id 512
set interfaces ge-1/0/9 description to_P1
set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.1/30
set interfaces ge-1/0/9 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 10.255.70.31/32
set protocols mpls interface ge-1/0/9.0
set protocols bgp group to_PE2 type internal
set protocols bgp group to_PE2 local-address 10.255.70.31
set protocols bgp group to_PE2 family l2vpn signaling
set protocols bgp group to_PE2 neighbor 172.30.14.1
set protocols ospf traffic-engineering

```



```

set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
set protocols ldp interface ge-1/0/9.0
set protocols ldp interface lo0.0
set routing-options router-id 10.255.70.31
set routing-options autonomous-system 64511
set routing-instances vpls_a instance-type vpls
set routing-instances vpls_a interface ge-1/0/6.512
set routing-instances vpls_a route-distinguisher 64511:1
set routing-instances vpls_a vrf-target target:64511:1
set routing-instances vpls_a protocols vpls no-tunnel-services
set routing-instances vpls_a protocols vpls site 1 site-identifier 1
set routing-instances vpls_a protocols vpls site 1 interface ge-1/0/6.512

```

## Device P1

```

set interfaces ge-1/0/3 description to_P2
set interfaces ge-1/0/3 unit 0 family inet address 10.40.0.1/30
set interfaces ge-1/0/3 unit 0 family mpls
set interfaces ge-1/0/9 description to_PE1
set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.2/30
set interfaces ge-1/0/9 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.16.1/32
set protocols mpls interface ge-1/0/9.0
set protocols mpls interface ge-1/0/3.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-1/0/3.0
set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-1/0/3.0
set protocols ldp interface ge-1/0/9.0
set protocols ldp interface lo0.0
set routing-options router-id 192.168.16.1

```

## Device P2

```

set interfaces ge-2/0/6 description to_P1
set interfaces ge-2/0/6 unit 0 family inet address 10.40.0.2/30
set interfaces ge-2/0/6 unit 0 family mpls
set interfaces ge-2/0/8 description to_PE2

```



```

set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 192.168.13.1/32
set protocols mpls interface ge-2/0/6.0
set protocols mpls interface ge-2/0/8.0
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-2/0/6.0
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-2/0/6.0
set protocols ldp interface ge-2/0/8.0
set protocols ldp interface lo0.0
set routing-options router-id 192.168.13.1

```

## Device PE2

```

set interfaces ge-2/0/8 description to-R1
set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
set interfaces ge-2/0/8 unit 0 family mpls
set interfaces ge-2/1/1 vlan-tagging
set interfaces ge-2/1/1 encapsulation vlan-vpls
set interfaces ge-2/1/1 unit 512 description to_vpls
set interfaces ge-2/1/1 unit 512 encapsulation vlan-vpls
set interfaces ge-2/1/1 unit 512 vlan-id 512
set interfaces lo0 unit 0 description loopback-interface
set interfaces lo0 unit 0 family inet address 172.30.14.1/32
set protocols mpls interface ge-2/0/8.0
set protocols bgp group to_PE1 type internal
set protocols bgp group to_PE1 local-address 172.30.14.1
set protocols bgp group to_PE1 family l2vpn signaling
set protocols bgp group to_PE1 neighbor 10.255.70.31
set protocols ospf traffic-engineering
set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
set protocols ospf area 0.0.0.0 interface lo0.0 passive
set protocols ldp interface ge-2/0/8.0
set protocols ldp interface lo0.0
set routing-options router-id 172.30.14.1
set routing-options autonomous-system 64511
set routing-instances vpls_a instance-type vpls
set routing-instances vpls_a interface ge-2/1/1.512
set routing-instances vpls_a route-distinguisher 64511:1

```



```

set routing-instances vpls_a vrf-target target:64511:1
set routing-instances vpls_a protocols vpls no-tunnel-services
set routing-instances vpls_a protocols vpls site 2 site-identifier 2
set routing-instances vpls_a protocols vpls site 2 interface ge-2/1/1.512

```

## Device CE2

```

set interfaces ge-2/0/7 unit 0 description to-Host2
set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
set interfaces ge-2/0/7 unit 0 family inet filter input ip-v4
set interfaces ge-2/1/2 vlan-tagging
set interfaces ge-2/1/2 unit 512 description to-PE2
set interfaces ge-2/1/2 unit 512 vlan-id 512
set interfaces ge-2/1/2 unit 512 family inet address 10.10.1.2/24
set interfaces lo0 unit 1 description loopback-interface
set interfaces lo0 unit 1 family inet address 192.168.0.2/32
set protocols ospf area 0.0.0.0 interface lo0.1 passive
set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
set protocols ospf area 0.0.0.0 interface ge-2/1/2.512
set firewall family inet filter ip-v4 term tcp80 from port 80
set firewall family inet filter ip-v4 term tcp80 then dscp ef
set firewall family inet filter ip-v4 term 12345 from port 12345
set firewall family inet filter ip-v4 term 12345 then dscp be
set firewall family inet filter ip-v4 term accept then accept
set class-of-service classifiers ieee-802.1 dscp1 forwarding-class expedited-forwarding loss-
priority low code-points ef
set class-of-service classifiers ieee-802.1 dscp1 forwarding-class best-effort loss-priority low
code-points be
set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class expedited-forwarding
loss-priority low code-point 010
set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class best-effort loss-
priority low code-point 000
set class-of-service interfaces ge-2/1/2 unit 512 rewrite-rules ieee-802.1 ieee1-c2
set class-of-service interfaces ge-2/1/2 unit 512 classifiers ieee-802.1 dscp1

```

## Step-by-Step Procedure

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

To configure Device CE1:



1. Configure the device interfaces.

```
[edit ]
user@CE1# set interfaces ge-1/0/1 unit 0 description to-Host1
user@CE1# set interfaces ge-1/0/1 unit 0 family inet address 172.16.50.2/30
user@CE1# set interfaces ge-1/0/1 unit 0 family inet filter input ip-v4
user@CE1# set interfaces lo0 unit 1 description loopback-interface
user@CE1# set interfaces lo0 unit 1 family inet address 192.168.0.1/32
```

2. Configure the VLAN parameters.

```
[edit ]
user@CE1# set interfaces ge-1/0/5 vlan-tagging
user@CE1# set interfaces ge-1/0/5 unit 512 description to_PE1
user@CE1# set interfaces ge-1/0/5 unit 512 vlan-id 512
user@CE1# set interfaces ge-1/0/5 unit 512 family inet address 10.10.1.1/24
```

3. Configure the class-of-service parameters.

```
[edit ]
user@CE1# set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class expedited-
forwarding loss-priority low code-point 010
user@CE1# set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class best-effort
loss-priority low code-point 000
user@CE1# set class-of-service classifiers ieee-802.1 dscp1 forwarding-class expedited-
forwarding loss-priority low code-points ef
user@CE1# set class-of-service classifiers ieee-802.1 dscp1 forwarding-class best-effort loss-
priority low code-points be
user@CE1# set class-of-service interfaces ge-1/0/5 unit 512 rewrite-rules ieee-802.1 ieee1-c2
user@CE1# set class-of-service interfaces ge-1/0/5 unit 512 classifiers ieee-802.1 dscp1
```

4. Configure the protocol parameters.

```
[edit ]
user@CE1# set protocols ospf area 0.0.0.0 interface ge-1/0/5.512
user@CE1# set protocols ospf area 0.0.0.0 interface ge-1/0/1.0 passive
user@CE1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
```



5. Configure the firewall DSCP rewrite parameters.

```
[edit ]
user@CE1# set firewall family inet filter ip-v4 term tcp80 from port 80
user@CE1# set firewall family inet filter ip-v4 term tcp80 then dscp ef
user@CE1# set firewall family inet filter ip-v4 term 12345 from port 12345
user@CE1# set firewall family inet filter ip-v4 term 12345 then dscp be
user@CE1# set firewall family inet filter ip-v4 term accept then accept
```

## Step-by-Step Procedure

To configure Device PE1:

1. Configure the device interfaces.

```
[edit ]
user@PE1# set interfaces ge-1/0/9 description to_P1
user@PE1# set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.1/30
user@PE1# set interfaces ge-1/0/9 unit 0 family mpls
user@PE1# set interfaces lo0 unit 0 description loopback-interface
user@PE1# set interfaces lo0 unit 0 family inet address 10.255.70.31/32
```

2. Configure the VLAN parameters.

```
[edit ]
user@PE1# set interfaces ge-1/0/6 vlan-tagging
user@PE1# set interfaces ge-1/0/6 encapsulation vlan-vpls
user@PE1# set interfaces ge-1/0/6 unit 512 description to_vpls
user@PE1# set interfaces ge-1/0/6 unit 512 encapsulation vlan-vpls
user@PE1# set interfaces ge-1/0/6 unit 512 vlan-id 512
```

3. Configure the protocol parameters.

```
[edit ]
user@PE1# set protocols mpls interface ge-1/0/9.0
user@PE1# set protocols bgp group to_PE2 type internal
user@PE1# set protocols bgp group to_PE2 local-address 10.255.70.31
user@PE1# set protocols bgp group to_PE2 family l2vpn signaling
user@PE1# set protocols bgp group to_PE2 neighbor 172.30.14.1
```



```

user@PE1# set protocols ospf traffic-engineering
user@PE1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@PE1# set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
user@PE1# set protocols ldp interface ge-1/0/9.0
user@PE1# set protocols ldp interface lo0.0

```

#### 4. Configure the routing option parameters.

```

[edit ]
user@PE1# set routing-options router-id 10.255.70.31
user@PE1# set routing-options autonomous-system 64511

```

#### 5. Configure the routing instance parameters.

```

[edit ]
user@PE1# set routing-instances vpls_a instance-type vpls
user@PE1# set routing-instances vpls_a interface ge-1/0/6.512
user@PE1# set routing-instances vpls_a route-distinguisher 64511:1
user@PE1# set routing-instances vpls_a vrf-target target:64511:1
user@PE1# set routing-instances vpls_a protocols vpls no-tunnel-services
user@PE1# set routing-instances vpls_a protocols vpls site 1 site-identifier 1
user@PE1# set routing-instances vpls_a protocols vpls site 1 interface ge-1/0/6.512

```

### Step-by-Step Procedure

To configure Device P1:

#### 1. Configure the device interfaces.

```

[edit ]
user@P1# set interfaces ge-1/0/3 description to_P2
user@P1# set interfaces ge-1/0/3 unit 0 family inet address 10.40.0.1/30
user@P1# set interfaces ge-1/0/3 unit 0 family mpls
user@P1# set interfaces ge-1/0/9 description to_PE1
user@P1# set interfaces ge-1/0/9 unit 0 family inet address 10.30.0.2/30
user@P1# set interfaces ge-1/0/9 unit 0 family mpls
user@P1# set interfaces lo0 unit 0 description loopback-interface
user@P1# set interfaces lo0 unit 0 family inet address 192.168.16.1/32

```



## 2. Configure the protocol parameters.

```
[edit ]
user@P1# set protocols mpls interface ge-1/0/9.0
user@P1# set protocols mpls interface ge-1/0/3.0
user@P1# set protocols ospf traffic-engineering
user@P1# set protocols ospf area 0.0.0.0 interface ge-1/0/3.0
user@P1# set protocols ospf area 0.0.0.0 interface ge-1/0/9.0
user@P1# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@P1# set protocols ldp interface ge-1/0/3.0
user@P1# set protocols ldp interface ge-1/0/9.0
user@P1# set protocols ldp interface lo0.0
```

## 3. Configure the routing options parameter.

```
[edit ]
user@P1# set routing-options router-id 192.168.16.1
```

## Step-by-Step Procedure

To configure Device P2:

### 1. Configure the device interfaces.

```
[edit ]
user@P2# set interfaces ge-2/0/6 description to_P1
user@P2# set interfaces ge-2/0/6 unit 0 family inet address 10.40.0.2/30
user@P2# set interfaces ge-2/0/6 unit 0 family mpls
user@P2# set interfaces ge-2/0/8 description to_PE2
user@P2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.1/30
user@P2# set interfaces ge-2/0/8 unit 0 family mpls
user@P2# set interfaces lo0 unit 0 description loopback-interface
user@P2# set interfaces lo0 unit 0 family inet address 192.168.13.1/32
```

### 2. Configure the protocol parameters.

```
[edit ]
user@P2# set protocols mpls interface ge-2/0/6.0
user@P2# set protocols mpls interface ge-2/0/8.0
```



```

user@P2# set protocols ospf traffic-engineering
user@P2# set protocols ospf area 0.0.0.0 interface ge-2/0/6.0
user@P2# set protocols ospf area 0.0.0.0 interface ge-2/0/8.0
user@P2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@P2# set protocols ldp interface ge-2/0/6.0
user@P2# set protocols ldp interface ge-2/0/8.0
user@P2# set protocols ldp interface lo0.0

```

### 3. Configure the routing option parameter.

```

[edit ]
user@P2# set routing-options router-id 192.168.13.1

```

## Step-by-Step Procedure

To configure Device PE2:

### 1. Configure the device interfaces.

```

[edit ]
user@PE2# set interfaces ge-2/0/8 description to-R1
user@PE2# set interfaces ge-2/0/8 unit 0 family inet address 10.50.0.2/30
user@PE2# set interfaces ge-2/0/8 unit 0 family mpls
user@PE2# set interfaces lo0 unit 0 description loopback-interface
user@PE2# set interfaces lo0 unit 0 family inet address 172.30.14.1/32

```

### 2. Configure the VLAN parameters.

```

[edit ]
user@PE2# set interfaces ge-2/1/1 vlan-tagging
user@PE2# set interfaces ge-2/1/1 encapsulation vlan-vpls
user@PE2# set interfaces ge-2/1/1 unit 512 description to_vpls
user@PE2# set interfaces ge-2/1/1 unit 512 encapsulation vlan-vpls
user@PE2# set interfaces ge-2/1/1 unit 512 vlan-id 512

```

### 3. Configure the protocol parameters.

```

[edit ]
user@PE2# set protocols mpls interface ge-2/0/8.0

```



```

user@PE2# set protocols bgp group to_PE1 type internal
user@PE2# set protocols bgp group to_PE1 local-address 172.30.14.1
user@PE2# 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@PE2# set routing-instances vpls_a instance-type vpls
user@PE2# set routing-instances vpls_a interface ge-2/1/1.512
user@PE2# set routing-instances vpls_a route-distinguisher 64511:1
user@PE2# set routing-instances vpls_a vrf-target target:64511:1
user@PE2# set routing-instances vpls_a protocols vpls no-tunnel-services
user@PE2# set routing-instances vpls_a protocols vpls site 2 site-identifier 2
user@PE2# set routing-instances vpls_a protocols vpls site 2 interface ge-2/1/1.512

```

### Step-by-Step Procedure

To configure Device CE2:

#### 1. Configure the device interfaces.

```

[edit ]
user@CE2# set interfaces ge-2/0/7 unit 0 description to-Host2
user@CE2# set interfaces ge-2/0/7 unit 0 family inet address 172.16.80.2/30
user@CE2# set interfaces ge-2/0/7 unit 0 family inet filter input ip-v4
user@CE2# set interfaces lo0 unit 1 description loopback-interface
user@CE2# set interfaces lo0 unit 1 family inet address 192.168.0.2/32

```



## 2. Configure the VLAN parameters

```
[edit ]
user@CE2# set interfaces ge-2/1/2 vlan-tagging
user@CE2# set interfaces ge-2/1/2 unit 512 description to-PE2
user@CE2# set interfaces ge-2/1/2 unit 512 vlan-id 512
user@CE2# set interfaces ge-2/1/2 unit 512 family inet address 10.10.1.2/24
```

## 3. Configure the class-of-service parameters.

```
[edit ]
user@CE2# set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class expedited-
forwarding loss-priority low code-point 010
user@CE2# set class-of-service rewrite-rules ieee-802.1 ieee1-c2 forwarding-class best-effort
loss-priority low code-point 000
user@CE2# set class-of-service classifiers ieee-802.1 dscp1 forwarding-class expedited-
forwarding loss-priority low code-points ef
user@CE2# set class-of-service classifiers ieee-802.1 dscp1 forwarding-class best-effort loss-
priority low code-points be
user@CE2# set class-of-service interfaces ge-2/1/2 unit 512 rewrite-rules ieee-802.1 ieee1-c2
user@CE2# set class-of-service interfaces ge-2/1/2 unit 512 classifiers ieee-802.1 dscp1
```

## 4. Configure the protocol parameters.

```
[edit ]
user@CE2# set protocols ospf area 0.0.0.0 interface lo0.0 passive
user@CE2# set protocols ospf area 0.0.0.0 interface ge-2/0/7.0 passive
user@CE2# set protocols ospf area 0.0.0.0 interface ge-2/1/2.512
```

## 5. Configure the firewall DSCP rewrite parameters.

```
[edit ]
user@CE2# set firewall family inet filter ip-v4 term tcp80 from port 80
user@CE2# set firewall family inet filter ip-v4 term tcp80 then dscp ef
user@CE2# set firewall family inet filter ip-v4 term 12345 from port 12345
user@CE2# set firewall family inet filter ip-v4 term 12345 then dscp be
user@CE2# set firewall family inet filter ip-v4 term accept then accept
```



## Results

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

```
user@CE1# show interfaces
ge-1/0/1 {
  unit 0 {
    description to-Host1;
    family inet {
      filter {
        input ip-v4;
      }
      address 172.16.50.2/30;
    }
  }
}
ge-1/0/5 {
  vlan-tagging;
  unit 512 {
    description to_PE1;
    vlan-id 512;
    family inet {
      address 10.10.1.1/24;
    }
  }
}
lo0 {
  unit 1 {
    description loopback-interface;
    family inet {
      address 192.168.0.1/32;
    }
  }
}
```

```
user@CE1# show class-of-service
classifiers {
  ieee-802.1 dscp1 {
```



```

        forwarding-class expedited-forwarding {
            loss-priority low code-points ef;
        }
        forwarding-class best-effort {
            loss-priority low code-points be;
        }
    }
}
interfaces {
    ge-1/0/5 {
        unit 512 {
            classifiers {
                ieee-802.1 dscp1;
            }
            rewrite-rules {
                ieee-802.1 ieee1-c2;
            }
        }
    }
}
rewrite-rules {
    ieee-802.1 ieee1-c2 {
        forwarding-class expedited-forwarding {
            loss-priority low code-point 010;
        }
        forwarding-class best-effort {
            loss-priority low code-point 000;
        }
    }
}
}

```

user@CE1# **show protocols**

```

ospf {
    area 0.0.0.0 {
        interface ge-1/0/5.512;
        interface ge-1/0/1.0 {
            passive;
        }
        interface lo0.1 {
            passive;
        }
    }
}

```



```

    }
}

```

```

user@CE1# show firewall
family inet {
    filter ip-v4 {
        term tcp80 {
            from {
                port 80;
            }
            then dscp ef;
        }
        term 12345 {
            from {
                port 12345;
            }
            then dscp be;
        }
        term accept {
            then accept;
        }
    }
}

```

If you are done configuring Device CE1, enter `commit` from configuration mode.

```

user@PE1# show interfaces
ge-1/0/6 {
    vlan-tagging;
    encapsulation vlan-vpls;
    unit 512 {
        description to_vpls;
        encapsulation vlan-vpls;
        vlan-id 512;
    }
}
ge-1/0/9 {
    description to_P1;
    unit 0 {
        family inet {
            address 10.30.0.1/30;

```



```

    }
    family mpls;
  }
}
lo0 {
  unit 0 {
    description loopback-interface;
    family inet {
      address 10.255.70.31/32;
    }
  }
}
}

```

```

user@PE1# show protocols
mpls {
  interface ge-1/0/9.0;
}
bgp {
  group to_PE2 {
    type internal;
    local-address 10.255.70.31;
    family l2vpn {
      signaling;
    }
    neighbor 172.30.14.1;
  }
}
ospf {
  traffic-engineering;
  area 0.0.0.0 {
    interface lo0.0 {
      passive;
    }
    interface ge-1/0/9.0;
  }
}
ldp {
  interface ge-1/0/9.0;
}

```



```

    interface lo0.0;
}

```

```

user@PE1# show routing-options
router-id 10.255.70.31;
autonomous-system 64511;

```

```

user@PE1# show routing-instances
vpls_a {
    instance-type vpls;
    interface ge-1/0/6.512;
    route-distinguisher 64511:1;
    vrf-target target:64511:1;
    protocols {
        vpls {
            no-tunnel-services;
            site 1 {
                site-identifier 1;
                interface ge-1/0/6.512;
            }
        }
    }
}

```

If you are done configuring Device PE1, enter `commit` from configuration mode.

```

user@P1# show interfaces
ge-1/0/3 {
    description to_P2;
    unit 0 {
        family inet {
            address 10.40.0.1/30;
        }
        family mpls;
    }
}
ge-1/0/9 {
    description to_PE1;
    unit 0 {

```



```

        family inet {
            address 10.30.0.2/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.16.1/32;
        }
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.16.1/32;
        }
    }
}
}

```

```

user@P1# show protocols
mpls {
    interface ge-1/0/9.0;
    interface ge-1/0/3.0;
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-1/0/3.0;
        interface ge-1/0/9.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-1/0/3.0;
    interface ge-1/0/9.0;
}

```



```

    interface lo0.0;
}

```

```

user@P1# show routing-options
router-id 192.168.16.1;

```

If you are done configuring Device P1, enter `commit` from configuration mode.

```

user@P2# show interfaces
ge-2/0/6 {
    description to_P1;
    unit 0 {
        family inet {
            address 10.40.0.2/30;
        }
        family mpls;
    }
}
ge-2/0/8 {
    description to_PE2;
    unit 0 {
        family inet {
            address 10.50.0.1/30;
        }
        family mpls;
    }
}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 192.168.13.1/32;
        }
    }
}

```

```

user@P2# show protocols
mpls {
    interface ge-2/0/6.0;
}

```



```

        interface ge-2/0/8.0;
    }
    ospf {
        traffic-engineering;
        area 0.0.0.0 {
            interface ge-2/0/6.0;
            interface ge-2/0/8.0;
            interface lo0.0 {
                passive;
            }
        }
    }
}
ldp {
    interface ge-2/0/6.0;
    interface ge-2/0/8.0;
    interface lo0.0;
}

```

```

user@P2# show routing-options
router-id 192.168.13.1;

```

If you are done configuring Device P2, enter `commit` from configuration mode.

```

user@PE2# show interfaces
ge-2/0/8 {
    description to-R1;
    unit 0 {
        family inet {
            address 10.50.0.2/30;
        }
        family mpls;
    }
}
ge-2/1/1 {
    vlan-tagging;
    encapsulation vlan-vpls;
    unit 512 {
        description to_vpls;
        encapsulation vlan-vpls;
        vlan-id 512;
    }
}

```



```

}
lo0 {
    unit 0 {
        description loopback-interface;
        family inet {
            address 172.30.14.1/32;
        }
    }
}

```

```

user@PE2# show protocols
mpls {
    interface ge-2/0/8.0;
}
bgp {
    group to_PE1 {
        type internal;
        local-address 172.30.14.1;
        family l2vpn {
            signaling;
        }
        neighbor 10.255.70.31;
    }
}
ospf {
    traffic-engineering;
    area 0.0.0.0 {
        interface ge-2/0/8.0;
        interface lo0.0 {
            passive;
        }
    }
}
ldp {
    interface ge-2/0/8.0;
}

```



```

    interface lo0.0;
}

```

```

user@PE2# show routing-options
router-id 172.30.14.1;
autonomous-system 64511;

```

```

user@PE2# show routing-instances
vpls_a {
    instance-type vpls;
    interface ge-2/1/1.512;
    route-distinguisher 64511:1;
    vrf-target target:64511:1;
    protocols {
        vpls {
            no-tunnel-services;
            site 2 {
                site-identifier 2;
                interface ge-2/1/1.512;
            }
        }
    }
}

```

If you are done configuring Device PE2, enter `commit` from configuration mode.

```

user@CE2# show interfaces
ge-2/0/7 {
    unit 0 {
        description to-Host2;
        family inet {
            filter {
                input ip-v4;
            }
            address 172.16.80.2/30;
        }
    }
}
ge-2/1/2 {

```



```

vlan-tagging;
unit 512 {
    description to-PE2;
    vlan-id 512;
    family inet {
        address 10.10.1.2/24;
    }
}
}
lo0 {
    unit 1 {
        description loopback-interface;
        family inet {
            address 192.168.0.2/32;
        }
    }
}
user@CE2# show class-of-service
classifiers {
    ieee-802.1 dscp1 {
        forwarding-class expedited-forwarding {
            loss-priority low code-points ef;
        }
        forwarding-class best-effort {
            loss-priority low code-points be;
        }
    }
}
interfaces {
    ge-2/1/2 {
        unit 512 {
            classifiers {
                ieee-802.1 dscp1;
            }
            rewrite-rules {
                ieee-802.1 ieee1-c2;
            }
        }
    }
}
rewrite-rules {
    ieee-802.1 ieee1-c2 {
        forwarding-class expedited-forwarding {

```



```

        loss-priority low code-point 010;
    }
    forwarding-class best-effort {
        loss-priority low code-point 000;
    }
}
}
user@CE2# show protocols
ospf {
    area 0.0.0.0 {
        interface lo0.1 {
            passive;
        }
        interface ge-2/0/7.0 {
            passive;
        }
        interface ge-2/1/2.512;
    }
}
user@CE2# show firewall
family inet {
    filter ip-v4 {
        term tcp80 {
            from {
                port 80;
            }
            then dscp ef;
        }
        term 12345 {
            from {
                port 12345;
            }
            then dscp be;
        }
        term accept {
            then accept;
        }
    }
}
}

```

If you are done configuring Device CE2, enter `commit` from configuration mode.



## Verification

### IN THIS SECTION

- [Clearing the Firewall Counters | 638](#)
- [Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Verifying the Results | 638](#)

Confirm that the configuration is working properly by verifying that the DSCP aliases are maintained from Device CE1 to Device CE2.

### Clearing the Firewall Counters

#### Purpose

Confirm that the firewall counters are cleared.

#### Action

On Device CE2, run the `clear firewall all` command to reset the firewall counters to 0.

```
user@CE2> clear firewall all
```

### Sending Traffic into the Network from TCP HTTP Ports 80 and 12345 and Verifying the Results

#### Purpose

Send traffic into the network that can be verified at Device CE2.

#### Action

Configure a new firewall on Device CE2 if you want to verify that the traffic that is being transmitted to Device Host2 from Device Host1 still has the correct DSCP aliases. The following commands create and apply the firewall filter that displays the traffic counts for each code point alias:

```
user@CE2# set firewall family inet filter count term be from dscp be
user@CE2# set firewall family inet filter count term be then count be
```



```

user@CE2# set firewall family inet filter count term ef from dscp ef
user@CE2# set firewall family inet filter count term ef then count ef
user@CE2# set firewall family inet filter count term accept then accept
user@CE2# set interfaces ge-2/0/7 unit 0 family inet filter output count

```

When you are done configuring Device CE2, enter `commit` from configuration mode.

When you are done testing, you can leave the counting filter in place, or remove it.

1. On Device Host1 use a traffic generator to send 20 TCP packets with a source port of 80 into the network.

The `-s` flag sets the source port. The `-k` flag causes the source port to remain steady instead of incrementing. The `-c` flag sets the number of packets to 20.

Repeat the task using a source port of 12345.

```

[user@host1]# hping 172.16.80.1 -s 80 -k -c 20
[user@host1]# hping 172.16.80.1 -s 12345 -k -c 20

```

2. On Device CE2, display the firewall counters by using the `show firewall` command.

```

user@CE2> show firewall
show firewall

Filter: __CE2/count
Counters:

```

Name	Bytes	Packets
be	800	20
ef	800	20

## Meaning

The code point aliases set by Device CE1 are maintained across the VPLS backbone and appear intact at Device CE2.

## RELATED DOCUMENTATION

[Example: Configuring and Applying Scheduler Maps](#) | 351



## Assigning Rewrite Rules on a Per-Customer Basis Using Policy Maps

### SUMMARY

This topic describes the purpose and configuration of policy maps. Policy maps enable you to assign rewrite rules on a per-customer basis.

### IN THIS SECTION

- [Policy Maps Overview | 640](#)
- [Configuring Policy Maps | 642](#)
- [Platform-Specific Policy Map Behavior | 645](#)

### Policy Maps Overview

Most commonly, packet marking (that is, setting rewrite rules) in Junos uses the forwarding class and loss priority determined through a behavior aggregate (BA) classifier or multifield classifier. The forwarding class and loss priority are also used to decide queuing behavior. This approach does not allow you to directly assign rewrite rules to each customer because of the limited number of combinations of forwarding class and loss priority. When a new customer is added, setting rewrite rules using this approach requires changes to the configuration on the core interfaces, which is undesirable as one mistake can affect traffic from all customers.

An alternative packet marking scheme, called *policy map*, enables you to define rewrite rules on a per-customer basis (that is, for each customer). The policy map makes it possible to use any packet field to identify a given flow and specify a rewrite value for that flow.

A policy map is defined at the `[edit class-of-service policy-map]` hierarchy level. Depending on your platform and software release, the policy map can define the following types of packet marking:

- IPv4 Precedence with the following options:
  - `proto-ip` – Mark the packet for IPv4 to IPv4 traffic.
  - `proto-mpls` – Mark the packet for an IPv4 packet entering an MPLS tunnel.
  - `proto-all` – Enable IP header marking.
- IPv6 Precedence with the following options:
  - `proto-ip` – Mark the packet for IPv6 to IPv6 traffic.
  - `proto-mpls` – Mark the packet for an IPv6 packet entering an MPLS tunnel.
- IPv4 DSCP with the following options:
  - `proto-ip` – Mark the packet for IPv4 to IPv4 traffic.



- `proto-mpls` – Mark the packet for an IPv4 packet entering an MPLS tunnel.
- `proto-all` – Enable IP header marking.
- IPv6 DSCP with the following options:
  - `proto-ip` – Mark the packet for IPv6 to IPv6 traffic.
  - `proto-mpls` – Mark the packet for an IPv6 packet entering an MPLS tunnel.
  - `proto-all` – Enable IPv6 header marking.
- MPLS EXP with the following options:
  - `all-label` – Mark all labels.
  - `outer-label` – Mark only the outer label.
- IEEE 802.1p with the following options:
  - `outer` – Mark only the outer VLAN header.
  - `outer-and-inner` – Mark both the outer and inner VLAN headers.
- IEEE 802.1ad with the following options:
  - `outer` – Mark only the outer VLAN header.
  - `outer-and-inner` – Mark both the outer and inner VLAN headers.



**NOTE:** To enable the policy map feature:

- On most devices, enable `enhanced-ip`, `enhanced-ethernet`, or `enhanced-mode` under `[edit chassis network-services]`.
- On PTX10002-36QDD routers, enable `policy-map-marking` on each egress logical interface that you want to do policy map marking. You enable `policy-map-marking` at the `[edit class-of-service interfaces interface-name unit unit-number]` hierarchy level.



**NOTE:** Policy maps have the following configuration restrictions:

- When configuring both `proto-ip` and `proto-mpls` options for `inet-precedence`, `inet6-precedence`, `dscp`, or `dscp-ipv6`, you must configure both options with the same code point or code point alias.



- You cannot configure `inet-precedence` and `dscp` in the same policy map.
- You cannot configure `inet6-precedence` and `dscp-ipv6` in the same policy map.
- In case of MPLS SWAP/PUSH operation, only the new labels are marked on all label-switching routers (LSRs), except the penultimate hop case where if it exposes the next label in the stack, then the exposed label is marked. Therefore, with the penultimate hop, the service label is changed.
- You cannot configure `ieee-802.1` and `ieee-802.1ad` in the same policy map.
- You cannot configure both `outer` and `outer-and-inner` options for `ieee-802.1` and `ieee-802.1ad` code points in the same policy map.
- For IEEE 802.1ad with the `outer-and-inner` option, the discard eligibility (DE) bit is marked only for the outer VLAN header. For the inner VLAN header, only the three CoS Bits are marked.

You assign a policy map to a customer through a firewall action on an ingress or egress firewall filter (where the match conditions identify the customer). Alternatively, you can assign a policy map to an ingress interface or a routing instance. You can assign multiple policy maps to a customer, one for each of the customer's traffic flows. Also, you can assign a single policy map to multiple customers.

A policy map is executed on a packet just before it is queued, so it overrides any other packet-marking scheme that was previously applied to the packet.

## Configuring Policy Maps

To assign rewrite rules on a per-customer basis:

### 1. Configure a policy map.

```
[edit class-of-service policy-map policy-map-name]
user@host# set inet-precedence proto-ip code-point [alias | bits];
user@host# set inet-precedence proto-mpls code-point [alias | bits]
user@host# set inet-precedence proto-all code-point [alias | bits]
user@host# set inet6-precedence proto-ip code-point [alias | bits];
user@host# set inet6-precedence proto-mpls code-point [alias | bits]
user@host# set dscp proto-ip code-point [alias | bits]
user@host# set dscp proto-mpls code-point [alias | bits]
user@host# set dscp proto-all code-point [alias | bits]
user@host# set dscp-ipv6 proto-ip code-point [alias | bits]
user@host# set dscp-ipv6 proto-mpls code-point [alias | bits]
user@host# set dscp-ipv6 proto-all code-point [alias | bits]
```



```

user@host# set exp all-label code-point [alias | bits]
user@host# set exp outer-label code-point [alias | bits]
user@host# set ieee-802.1 outer code-point [alias | bits]
user@host# set ieee-802.1 outer-and-inner code-point [alias | bits]
user@host# set ieee-802.1ad outer code-point [alias | bits]
user@host# set ieee-802.1ad outer-and-inner code-point [alias | bits]

```

For example:

```

[edit class-of-service]
user@host# set policy-map pm1 dscp proto-ip code-point 111000
user@host# set policy-map pm1 ieee-802.1 outer code-point 001
user@host# show
policy-map {
    pm1 {
        dscp proto-ip code-point 111000;
        ieee-802.1 outer code-point 001;
    }
}

```

## 2. Apply the policy map.

- Apply the policy map to an ingress or egress firewall filter.

```

[edit firewall family protocol-family-name filter filter-name]
user@host# set term term-name from match-conditions
user@host# set term term-name then policy-map policy-map-name

```

For example:

```

[edit firewall family inet filter f1]
user@host# set term t1 from address 10.2.2.0/24
user@host# set term t1 then policy-map pm1
user@host# show
term t1 {
    from {
        address {
            10.2.2.0/24;
        }
    }
}

```



```
then policy-map pm1;
}
```



**NOTE:** In this example, every IPv4 packet arriving from IP address 10.2.2.0/24 is assigned a DSCP value of 111000.

- Alternatively, apply the policy map to a routing instance.

```
[edit class-of-service]
user@host# set routing-instances routing-instance-name policy-map policy-map-name
```

For example:

```
[edit class-of-service]
user@host# set routing-instances r1 policy-map pm1
user@host# show
routing-instances {
  r1 {
    policy-map pm1;
  }
}
```



**NOTE:** In this example, every IPv4 packet in routing instance r1 is assigned a DSCP value of 111000.

- Alternatively, apply the policy map directly to an *ingress* interface.

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

For example:

```
[edit class-of-service]
user@host# set interfaces xe-4/0/0 unit 0 policy-map pm1
user@host# show
```



```
xe-4/0/0 {
    unit 0 {
        policy-map pm1;
    }
}
```



**NOTE:** In this example, every IPv4 packet arriving on interface xe-4/0/0 unit 0 is assigned a DSCP value of 111000.

Platform-Specific Policy Map Behavior

Use [Feature Explorer](#) to confirm platform and release support for [policy maps](#).

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

Platform	Difference
PTX10002-36QDD	<ul style="list-style-type: none"><li>• Enable policy-map-marking on each egress logical interface that you want to do policy map marking. You enable policy-map-marking at the [edit class-of-service interfaces <i>interface-name</i> unit <i>unit-number</i>] hierarchy level.</li><li>• INET-Precedence option marks both IPv4 and IPv6 packets.</li><li>• Due to hardware limitations, you can't create a policy map with both Layer 2 and Layer 3 packet marking types.</li><li>• Enabling policy-map action on an egress filter interface does not have any impact on rewrite values.</li><li>• You can also use policy maps to pass meta data between firewall filters. That is, a policy map value set in an ingress filter can be matched on in an egress filter.</li></ul>



## RELATED DOCUMENTATION

*policy-map (Class of Service)*

*show class-of-service policy-map*

## Host Outbound Traffic IEEE802.1p Rewrite

### SUMMARY

This topic describes how to rewrite the IEEE 802.1p field in the Ethernet frame header for host outbound traffic, that is, traffic generated by the Routing Engine. You can set a global value for the priority code point that applies to all host outbound traffic. Additionally, or alternatively, you can specify that rewrite rules are applied to all host outbound traffic on egress logical interfaces. These are rules that you have previously configured to set the IEEE 802.1p field for data traffic on those interfaces.

### IN THIS SECTION

- [Configuring a Global Default IEEE 802.1p Value for All Host Outbound Traffic | 646](#)
- [Apply Egress Interface Rewrite Rules to the IEEE 802.1p Field for All Host Outbound Traffic on an Interface | 647](#)
- [Platform-Specific Host Outbound Traffic Rewrite Behavior | 649](#)

To configure the IEEE 802.1p field settings:

1. (Optional) Specify a global default value for the IEEE 802.1p field for all host outbound traffic.
2. (Optional) Specify that the IEEE 802.1p rewrite rules for the egress logical interfaces are applied to all host outbound traffic on those interfaces.

## Configuring a Global Default IEEE 802.1p Value for All Host Outbound Traffic

### SUMMARY

This task describes how to configure a global default value for the IEEE 802.1p field for all host outbound traffic.

To configure a global default value for the IEEE 802.1p field:



1. Specify the value:

```
[edit class-of-service host-outbound-traffic ieee-802.1]
user@host# set default value
```

2. For example, specify that a value of 010 is applied to all host outbound traffic:

```
[edit class-of-service host-outbound-traffic ieee-802.1]
user@host# set default 010
```

## Apply Egress Interface Rewrite Rules to the IEEE 802.1p Field for All Host Outbound Traffic on an Interface

---

### SUMMARY

This topic describes how to apply rewrite rules for egress logical interfaces to the IEEE 802.1p field for all host outbound traffic on those interfaces.

---

This task requires separately configured rewrite rules that map packet loss priority information to the code point value in the 802.1p field for data traffic on egress logical interfaces. See ["Rewriting Packet Headers to Ensure Forwarding Behavior" on page 532](#).

To configure the rewrite rules:

1. Configure the CoS rewrite rules to map the forwarding class to the desired value for the 802.1p field. See ["Configuring Rewrite Rules" on page 538](#).
2. Associate the rewrite rules to the desired egress logical interfaces. See ["Applying Rewrite Rules to Output Logical Interfaces" on page 550](#).
3. (Optional) Configure the forwarding class for host outbound traffic. Do not configure this forwarding class if you want to use the default forwarding class assignment (input classification). See ["Overriding the Input Classification" on page 330](#).

To configure the rewrite rules to apply to the host outbound traffic IEEE 802.1p field:





**NOTE:** Enabling IEEE 802.1p rewrite rules for host outbound traffic without creating any corresponding IEEE 802.1p rewrite rules sets the IEEE 802.1p code point to be automatically 000 for all host generated traffic that exits that interface.

Configure the rewrite rules:

```
[edit class-of-service host-outbound-traffic ieee-802.1]
user@host# set rewrite-rules
```

For example:

```
[edit class-of-service]
rewrite-rules {
  ieee-802.1 rewrite_foo {
    forwarding-class network-control {
      loss-priority low code-point 101;
    }
  }
}
interfaces {
  ge-1/0/0 {
    unit 100 {
      rewrite-rules {
        ieee-802.1 rewrite_foo vlan-tag outer-and-inner;
      }
    }
  }
}
host-outbound-traffic {
  forwarding-class network-control;
}
host-outbound-traffic {
  ieee-802.1 {
    rewrite-rules;
  }
}
```



Platform-Specific Host Outbound Traffic Rewrite Behavior

Use [Feature Explorer](#) to confirm platform and release support for [802.1p rewrite rules for host outbound traffic](#).

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

Platform	Difference
ACX7000	ACX7000 routers support only the default option to configure a global default value for the IEEE 802.1p field for all host outbound traffic.

RELATED DOCUMENTATION

| [Rewriting Packet Headers to Ensure Forwarding Behavior](#) | 532



# Altering Class of Service Values in Packets Exiting the Network Using IPv6 DiffServ

## IN THIS CHAPTER

- [Resources for CoS with DiffServ for IPv6 | 651](#)
- [System Requirements for CoS with DiffServ for IPv6 | 651](#)
- [Terms and Acronyms for CoS with DiffServ for IPv6 | 652](#)
- [Default DSCP Mappings | 652](#)
- [Default Forwarding Classes | 655](#)
- [Juniper Networks Default Forwarding Classes | 659](#)
- [Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)
- [Configuring a Firewall Filter for an MF Classifier on Customer Interfaces | 662](#)
- [Applying the Firewall Filter to Customer Interfaces | 663](#)
- [Assigning Forwarding Classes to Output Queues | 664](#)
- [Configuring Rewrite Rules | 665](#)
- [DSCP IPv6 Rewrites and Forwarding Class Maps | 666](#)
- [Applying Rewrite Rules to an Interface | 667](#)
- [Configuring RED Drop Profiles | 667](#)
- [Configuring BA Classifiers | 668](#)
- [Applying a BA Classifier to an Interface | 669](#)
- [Configuring a Scheduler | 670](#)
- [Configuring Scheduler Maps | 671](#)
- [Applying a Scheduler Map to an Interface | 671](#)
- [Example: Configuring DiffServ for IPv6 | 672](#)



## 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 | 651](#)

[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)

## 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 | 661](#)

[Example: Configuring DiffServ for IPv6 | 672](#)



## Terms and Acronyms for CoS with DiffServ for IPv6

IN THIS SECTION

- classifier | 652
- class of service | 652
- Differentiated Services (DiffServ) | 652
- Differentiated Services code point (DSCP) | 652

**classifier**

A method of reading a sequence of bits in a packet header or label and determining the packet's forwarding class.

**class of service**

A set of forwarding class parameters that define different treatment for different traffic flows.

**Differentiated Services (DiffServ)**

A standards-based method of associating CoS parameters with traffic flows and their forwarding classes.

**Differentiated Services code point (DSCP)**

Values for a 6-bit field defined for IPv4 and IPv6 packet headers that can be used to enforce CoS distinctions in routers.

## Default DSCP Mappings

Table 59 on page 652 shows the mapping of DiffServ service class meanings (aliases) to DSCPs.

Table 59: Default DSCP Mappings

DiffServ Service Class Alias	IPv4 and IPv6 DSCP Mapping
ef	101110



**Table 59: Default DSCP Mappings** *(Continued)*

DiffServ Service Class Alias	IPv4 and IPv6 DSCP Mapping
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 59: Default DSCP Mappings (*Continued*)

DiffServ Service Class Alias	IPv4 and IPv6 DSCP Mapping
cs3	011000
cs4	100000
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

<a href="#">System Requirements for CoS with DiffServ for IPv6   651</a>
<a href="#">Roadmap for Configuring CoS with IPv6 DiffServ   661</a>
<a href="#">Example: Configuring DiffServ for IPv6   672</a>

Default Forwarding Classes

Table 60 on page 655 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 60: 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



**Table 60: Default Behavior Aggregate Classification (Continued)**

DSCP and DSCP IPv6	Forwarding Class	PLP
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 61 on page 657](#) shows the router forwarding classes associated with each well-known DSCP code point and the resources assigned to their output queues.



**Table 61: Forwarding Classes and Queues Classification**

DCSP 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



**Table 61: Forwarding Classes and Queues Classification (Continued)**

DCSP Alias	DSCP Bits	Forwarding Class	PLP	Queue
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 62 on page 658](#) shows the resources assigned to the four forwarding classes in this example.

**Table 62: Resources Assigned to Queues**

Queue	Forwarding Class	Transmit Rate	Buffer Size	Priority
0	be (data)	40%	40%	Low
1	ef (financial)	10%	10%	High
2	af (audiovisual)	45%	45%	High (with RED)
3	nc (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

<a href="#">System Requirements for CoS with DiffServ for IPv6   651</a>
<a href="#">Roadmap for Configuring CoS with IPv6 DiffServ   661</a>
<a href="#">Example: Configuring DiffServ for IPv6   672</a>

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 63 on page 659 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 63: 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 | 2](#)

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)

[Example: Configuring DiffServ for IPv6 | 672](#)

## 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 662](#).

Next, apply the MF classifier to the appropriate interface. See ["Applying the Firewall Filter to Customer Interfaces" on page 663](#).

- Assign the forwarding classes established by the MF classifier to output queues. See ["Assigning Forwarding Classes to Output Queues" on page 664](#).
- 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 665](#).

Next, apply the rewrite rules to the appropriate interface. See ["Applying Rewrite Rules to an Interface" on page 667](#).

- 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 668](#).

Next, apply the BA classifier to the appropriate interface. See ["Applying a BA Classifier to an Interface" on page 669](#).

- 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 667](#).
- Configure schedulers to assign resources, priorities, and drop profiles to output queues. See ["Configuring a Scheduler" on page 670](#).
- Configure a scheduler map to assign a forwarding class to a scheduler. See ["Configuring Scheduler Maps" on page 671](#).



Next, apply the scheduler map to the appropriate interface. See ["Applying a Scheduler Map to an Interface" on page 671](#).

## RELATED DOCUMENTATION

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

[Example: Configuring DiffServ for IPv6 | 672](#)

## 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 | 663](#)

[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

[Example: Configuring DiffServ for IPv6 | 672](#)

## 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 | 662](#)

[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

[Example: Configuring DiffServ for IPv6 | 672](#)

## 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 | 661](#)



---

[System Requirements for CoS with DiffServ for IPv6 | 651](#)[Example: Configuring DiffServ for IPv6 | 672](#)

---

## 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 | 667](#)[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)[System Requirements for CoS with DiffServ for IPv6 | 651](#)[Example: Configuring DiffServ for IPv6 | 672](#)

---



## 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](#) | 305



## 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 | 665](#)

---

[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)

---

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

---

[Example: Configuring DiffServ for IPv6 | 672](#)

## 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 | 661](#)

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

[Example: Configuring DiffServ for IPv6 | 672](#)

## 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 | 669](#)

[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

[Example: Configuring DiffServ for IPv6 | 672](#)

## 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 | 668](#)

[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

[Example: Configuring DiffServ for IPv6 | 672](#)



## 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 | 661](#)

---

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

---

[Example: Configuring DiffServ for IPv6 | 672](#)



## 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 | 671](#)

---

[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)

---

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

---

[Example: Configuring DiffServ for IPv6 | 672](#)

## 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 | 671
- Roadmap for Configuring CoS with IPv6 DiffServ | 661
- System Requirements for CoS with DiffServ for IPv6 | 651
- Example: Configuring DiffServ for IPv6 | 672

Example: Configuring DiffServ for IPv6

IN THIS SECTION

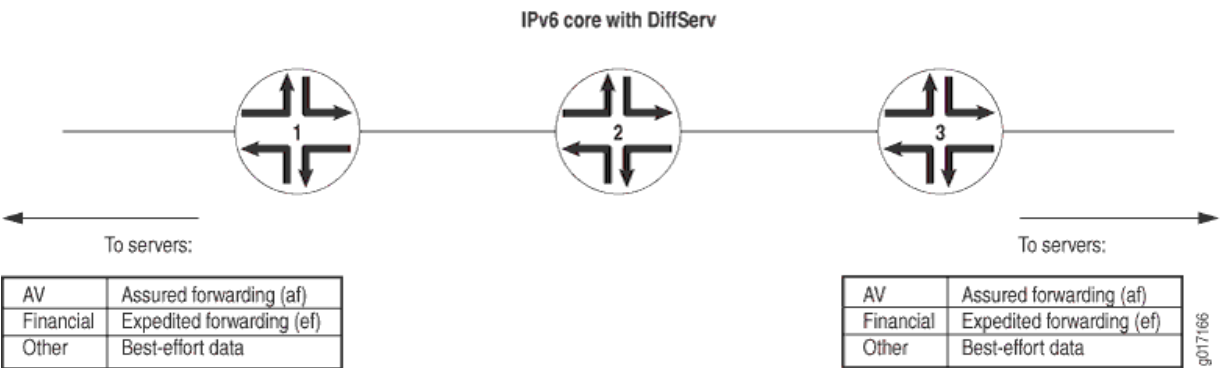
- Configuration | 672
- Verification | 685

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 49 on page 672 shows the topology of the three routers and links that are used as a case study in this chapter.

Figure 49: 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 50: IPv6 DiffServ Configuration**

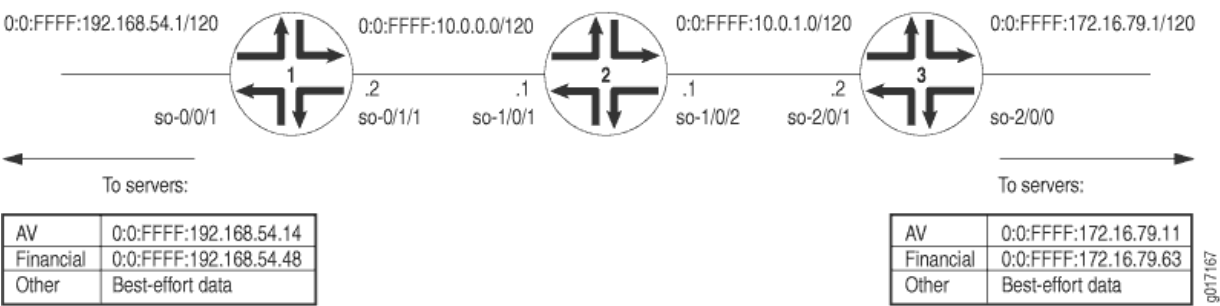


Figure 50 on page 673 shows the complete topology for IPv6 DiffServ, complete with interfaces and IPv6 addresses. The IPv4-mapped IPv6 address format described in RFC 5952 is used.

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

## Router 2

```
[edit]
class-of-service {
  classifiers { # Router 2 classifiers.
    dscp-ipv6 IPv6-classifier {
      import default; # Uses the DSCP default map.
      forwarding-class be-DATA-class {
        loss-priority high code-points 000001;
      }
      forwarding-class ef-FIN-class {
        loss-priority high code-points 101111;
      }
      forwarding-class af-AV-class {
        loss-priority high code-points 001100;
```



```

    }
    forwarding-class nc-CONTROL-class {
        loss-priority high code-points 110001;
    }
}
drop-profiles {      # Router 2 drop profiles.
    af-AV-normal {
        interpolate {
            fill-level [95 100];
            drop-probability [0 100];
        }
    }
    af-AV-with-PLP {
        interpolate {
            fill-level [60 70 80 90 95];
            drop-probability [80 90 95 97 100];
        }
    }
}
forwarding-classes {    # Router 2 forwarding classes.
    queue 0 be-DATA-class;
    queue 1 ef-FIN-class;
    queue 2 af-AV-class;
    queue 3 nc-CONTROL-class;
}
interfaces {            # Router 2 class-of-service interfaces.
    so-1/0/1 { # Connected to R1.
        scheduler-map diffserv-cos-map;
        unit 0 {
            classifiers {
                dscp-ipv6 IPv6-classifier;
            }
            rewrite-rules {
                dscp-ipv6 rewrite-IPv6-dscp;
            }
        }
    }
    so-1/0/2 { # Connected to R3.
        scheduler-map diffserv-cos-map;
        unit 0 {
            classifiers {
                dscp-ipv6 IPv6-classifier;
            }
        }
    }
}

```



```

    }
    rewrite-rules {
        dscp-ipv6 rewrite-IPv6-dscp;
    }
}

rewrite-rules rewrite-IPv6-dscps {    # Router 2 rewrite rules.
    forwarding-class be-DATA-class {
        loss-priority low code points 000000;
        loss-priority high code points 000001;
    }
    forwarding-class ef-FIN-class {
        loss-priority low code points 101110;
        loss-priority high code points 101111;
    }
    forwarding-class af-AV-class {
        loss-priority low code points 001010;
        loss-priority high code points 001100;
    }
    forwarding-class nc-CONTROL-class {
        loss-priority low code points 110000;
        loss-priority high code points 110001;
    }
}

scheduler-maps {                    # Router 2 scheduler maps.
    diffserv-cos-map {
        forwarding-class be-DATA-class scheduler be-DATA-scheduler;
        forwarding-class ef-FIN-class scheduler ef-FIN-scheduler;
        forwarding-class af-AV-class scheduler af-AV-scheduler;
        forwarding-class nc-CONTROL-class scheduler nc-CONTROL-scheduler;
    }
}

schedulers {                        # Router 2 schedulers.
    be-DATA-scheduler {
        transmit-rate percent 40;
        buffer-size percent 40;
        priority low;
    }

    ef-FIN-scheduler {
        transmit-rate percent 10;
        buffer-size percent 10;
        priority high;
    }
}

```



```

    }

    af-AV-scheduler {
        transmit-rate percent 45;
        buffer-size percent 45;
        priority high;
        drop-profile-map loss-priority low protocol any drop-profile af-AV-normal;
        drop-profile-map loss-priority high protocol any drop-profile af-AV-with-PLP;
    }

    nc-CONTROL-scheduler {
        transmit-rate percent 5;
        buffer-size percent 5;
        priority low;
    }
}

interfaces {
    # R2 interfaces.
    so-1/0/1 { # Connected to R1.
        unit 0 {
            family inet {
                address 10.0.0.1/24;
            }
            family inet6 {
                address 0:0:FFFF:10.0.0.1/120;
            }
        }
    }
    so-1/0/2 { # Connected to R3.
        unit 0 {
            family inet {
                address 10.0.1.1/24;
            }
            family inet6 {
                address 0:0:FFFF:10.0.1.1/120;
            }
        }
    }
}

```

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



**Router 1**

```

[edit]
class-of-service {
  classifiers {          # Router 1 classifiers.
    dscp-ipv6 IPv6-classifier {
      import default; # Uses the DSCP default map.
      forwarding-class be-DATA-class {
        loss-priority high code-points 000001;
      }
      forwarding-class ef-FIN-class {
        loss-priority high code-points 101111;
      }
      forwarding-class af-AV-class {
        loss-priority high code-points 001100;
      }
      forwarding-class nc-CONTROL-class {
        loss-priority high code-points 110001;
      }
    }
  }
  drop-profiles {       # Router 1 drop profiles.
    af-AV-normal {
      interpolate {
        fill-level [95 100];
        drop-probability [0 100];
      }
    }
    af-AV-with-PLP {
      interpolate {
        fill-level [60 70 80 90 95];
        drop-probability [80 90 95 97 100];
      }
    }
  }
  forwarding-classes {  # Router 1 forwarding classes.
    queue 0 be-DATA-class;
    queue 1 ef-FIN-class;
    queue 2 af-AV-class;
    queue 3 nc-CONTROL-class;
  }
  interfaces {          # Router 1 class-of-service interfaces.

```



```

so-0/1/1 { # To servers.
    scheduler-map diffserv-cos-map;
    unit 0 {
        classifiers {
            dscp-ipv6 IPv6-classifier;
        }
        rewrite-rules {
            dscp-ipv6 rewrite-IPv6-dscp;
        }
    }
}

rewrite-rules rewrite-IPv6-dscps { # Router 1 rewrite rules.
    forwarding-class be-DATA-class {
        loss-priority low code points 000000;
        loss-priority high code points 000001;
    }
    forwarding-class ef-FIN-class {
        loss-priority low code points 101110;
        loss-priority high code points 101111;
    }
    forwarding-class af-AV-class {
        loss-priority low code points 001010;
        loss-priority high code points 001100;
    }
    forwarding-class nc-CONTROL-class {
        loss-priority low code points 110000;
        loss-priority high code points 110001;
    }
}

scheduler-maps { # Router 1 scheduler map.
    diffserv-cos-map {
        forwarding-class be-DATA-class scheduler be-DATA-scheduler;
        forwarding-class ef-FIN-class scheduler ef-FIN-scheduler;
        forwarding-class af-AV-class scheduler af-AV-scheduler;
        forwarding-class nc-CONTROL-class scheduler nc-CONTROL-scheduler;
    }
}

schedulers { # Router 1 schedulers.
    be-DATA-scheduler {
        transmit-rate percent 40;
        buffer-size percent 40;
        priority low;
    }
}

```



```

        ef-FIN-scheduler {
            transmit-rate percent 10;
            buffer-size percent 10;
            priority high;
        }

        af-AV-scheduler {
            transmit-rate percent 45;
            buffer-size percent 45;
            priority high;
            drop-profile-map loss-priority low protocol any drop-profile af-AV-normal;
            drop-profile-map loss-priority high protocol any drop-profile af-AV-with-PLP;
        }

        nc-CONTROL-scheduler {
            transmit-rate percent 5;
            buffer-size percent 5;
            priority low;
        }
    }
}

firewall {
    # Router 1 firewall policer and filter.
    policer ef-FIN-Policer-Profile {
        if-exceeding {
            bandwidth-percent 10;
            burst-size-limit 2k;
        }
        then loss-priority high;
    }
    family inet6 {
        filter mf-classifier {
            filter-specific;
            term AV {
                from {
                    destination-address {
                        0:0:FFFF:172.16.79.11;
                    }
                }
                then {
                    loss-priority low;
                    forwarding-class af-AV-class;
                }
            }
            term Finance {
                from {

```



```

        destination-address {
            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.
}
}
}
}
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 {
            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.
}
}
}
}
interfaces {
    # Router 3 interfaces.
    so-2/0/0 { # To servers.
        unit 0 {
            family inet {
                address 1172.16.79.1/24;
            }
            family inet6 {
                filter {
                    input mf-classifier;
                }
                address 0:0:FFFF:172.16.79.1/120;
            }
        }
    }
}

```



```

so-2/0/1 { # to R2
    unit 0 {
        family inet {
            address 10.0.1.2/24;
        }
        family inet6 {
            address 0:0:FFFF:10.0.1.2/120;
        }
    }
}
}
}
}
}

```

## Verification

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

- `show class-of-service classifier type dscp-ipv6`
- `show class-of-service rewrite-rule type dscp-ipv6`
- `show class-of-service interface`
- `show class-of-service forwarding-table classifier mapping`
- `show class-of-service forwarding-table rewrite-rule mapping`
- `show class-of-service scheduler-map scheduler-map-name`
- `show class-of-service forwarding-table scheduler-map`

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

### DiffServ Classifiers

```

user@R1> show class-of-service classifier type dscp-ipv6
Classifier: dscp-ipv6-default, Code point type: dscp-ipv6, Index: 4

```

Code point	Forwarding class	Loss priority
000000	be-DATA-class	low
000001	be-DATA-class	low
000010	be-DATA-class	low
000011	be-DATA-class	low
000100	be-DATA-class	low



000101	be-DATA-class	low
000110	be-DATA-class	low
000111	be-DATA-class	low
001000	be-DATA-class	low
001001	be-DATA-class	low
001010	af-AV-class	low
001011	be-DATA-class	low
001100	af-AV-class	high
001101	be-DATA-class	low
001110	af-AV-class	high
001111	be-DATA-class	low
010000	be-DATA-class	low
010001	be-DATA-class	low
010010	be-DATA-class	low
010011	be-DATA-class	low
010100	be-DATA-class	low
010101	be-DATA-class	low
010110	be-DATA-class	low
010111	be-DATA-class	low
011000	be-DATA-class	low
011001	be-DATA-class	low
011010	be-DATA-class	low
011011	be-DATA-class	low
011100	be-DATA-class	low
011101	be-DATA-class	low
011110	be-DATA-class	low
011111	be-DATA-class	low
100000	be-DATA-class	low
100001	be-DATA-class	low
100010	be-DATA-class	low
100011	be-DATA-class	low
100100	be-DATA-class	low
100101	be-DATA-class	low
100110	be-DATA-class	low
100111	be-DATA-class	low
101000	be-DATA-class	low
101001	be-DATA-class	low
101010	be-DATA-class	low
101011	be-DATA-class	low
101100	be-DATA-class	low
101101	be-DATA-class	low
101110	ef-FIN-class	low
101111	be-DATA-class	low



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

Classifier: IPv6-classifier, Code point type: dscp-ipv6, Index: 18301

Code point	Forwarding class	Loss priority
000000	be-DATA-class	low
000001	be-DATA-class	high
000010	be-DATA-class	low
000011	be-DATA-class	low
000100	be-DATA-class	low
000101	be-DATA-class	low
000110	be-DATA-class	low
000111	be-DATA-class	low
001000	be-DATA-class	low
001001	be-DATA-class	low
001010	af-AV-class	low
001011	be-DATA-class	low
001100	af-AV-class	high
001101	be-DATA-class	low
001110	af-AV-class	high
001111	be-DATA-class	low
010000	be-DATA-class	low
010001	be-DATA-class	low
010010	be-DATA-class	low
010011	be-DATA-class	low
010100	be-DATA-class	low
010101	be-DATA-class	low
010110	be-DATA-class	low
010111	be-DATA-class	low
011000	be-DATA-class	low



011001	be-DATA-class	low
011010	be-DATA-class	low
011011	be-DATA-class	low
011100	be-DATA-class	low
011101	be-DATA-class	low
011110	be-DATA-class	low
011111	be-DATA-class	low
100000	be-DATA-class	low
100001	be-DATA-class	low
100010	be-DATA-class	low
100011	be-DATA-class	low
100100	be-DATA-class	low
100101	be-DATA-class	low
100110	be-DATA-class	low
100111	be-DATA-class	low
101000	be-DATA-class	low
101001	be-DATA-class	low
101010	be-DATA-class	low
101011	be-DATA-class	low
101100	be-DATA-class	low
101101	be-DATA-class	low
101110	ef-FIN-class	low
101111	ef-FIN-class	high
110000	nc-CONTROL-class	low
110001	nc-CONTROL-class	high
110010	be-DATA-class	low
110011	be-DATA-class	low
110100	be-DATA-class	low
110101	be-DATA-class	low
110110	be-DATA-class	low
110111	be-DATA-class	low
111000	nc-CONTROL-class	low
111001	be-DATA-class	low
111010	be-DATA-class	low
111011	be-DATA-class	low
111100	be-DATA-class	low
111101	be-DATA-class	low
111110	be-DATA-class	low
111111	be-DATA-class	low



## Rewrite Rules

```

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

```

## Class-of-Service Interfaces

```

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

```



Logical interface: so-0/1/1.0, Index: 69

Object	Name	Type	Index
Rewrite	exp-default	exp	21
Classifier	exp-default	exp	5
Classifier	ipprec-compatibility	ip	8

## Classifier Mapping

```
user@R1> show class-of-service forwarding-table classifier mapping
```

Table Index/			
Interface	Index	Q num	Table type
so-0/0/1.0	68	18301	IPv6 DSCP
so-0/1/1.0	69	8	IPv4 precedence

## Rewrite Rule Mapping

```
user@R1> show class-of-service forwarding-table rewrite-rule mapping
```

Interface	Index	Table index	Type
so-0/1/1.0	68	58077	IPv6 DSCP

## Scheduler Map

```
user@R1> show class-of-service scheduler-map diffserv-cos-map
```

Scheduler map: diffserv-cos-map, Index: 1094596010

Scheduler: be-DATA-scheduler, Forwarding class: be-DATA-class, Index: 14343

Transmit rate: 40 percent, Rate Limit: none, Buffer size: 40 percent,

Priority: low

Drop profiles:

Loss priority	Protocol	Index	Name
Low	non-TCP	1	<default-drop-profile>
Low	TCP	1	<default-drop-profile>
High	non-TCP	1	<default-drop-profile>
High	TCP	1	<default-drop-profile>

Scheduler: ef-FIN-scheduler, Forwarding class: ef-FIN-class, Index: 21707

Transmit rate: 10 percent, Rate Limit: none, Buffer size: 10 percent,

Priority: high

Drop profiles:

Loss priority	Protocol	Index	Name
Low	non-TCP	1	<default-drop-profile>
Low	TCP	1	<default-drop-profile>



```

    High          non-TCP          1    <default-drop-profile>
    High          TCP              1    <default-drop-profile>
Scheduler: af-AV-scheduler, Forwarding class: af-AV-class, Index: 51704
  Transmit rate: 45 percent, Rate Limit: none, Buffer size: 45 percent,
  Priority: high
  Drop profiles:
    Loss priority  Protocol  Index  Name
    Low           non-TCP   61474  af-AV-normal
    Low           TCP       61474  af-AV-normal
    High          non-TCP   65199  af-AV-with-PLP
    High          TCP       65199  af-AV-with-PLP
Scheduler: nc-CONTROL-scheduler, Forwarding class: nc-CONTROL-class, Index: 50404
  Transmit rate: 5 percent, Rate Limit: none, Buffer size: 5 percent,
  Priority: low
  Drop profiles:
    Loss priority  Protocol  Index  Name
    Low           non-TCP   1      <default-drop-profile>
    Low           TCP       1      <default-drop-profile>
    High          non-TCP   1      <default-drop-profile>
    High          TCP       1      <default-drop-profile>

user@R1> show class-of-service forwarding-table scheduler-map
...
Interface: so-0/0/1 (Index: 141, Map index: -543019056, Map type: FINAL,
Num of queues: 4):
  Entry 0 (Scheduler index: 14343, Queue #: 0):
    Tx rate: 0 Kb (40%), Buffer size: 40 percent
  Priority low
    PLP high: 1, PLP low: 1, TCP PLP high: 1, TCP PLP low: 1
  Entry 1 (Scheduler index: 21707, Queue #: 1):
    Tx rate: 0 Kb (10%), Buffer size: 10 percent
  Priority high
    PLP high: 1, PLP low: 1, TCP PLP high: 1, TCP PLP low: 1
  Entry 2 (Scheduler index: 51704, Queue #: 2):
    Tx rate: 0 Kb (45%), Buffer size: 45 percent
  Priority high
    PLP high: 65199, PLP low: 61474, TCP PLP high: 65199, TCP PLP low: 61474
  Entry 3 (Scheduler index: 50404, Queue #: 3):
    Tx rate: 0 Kb (5%), Buffer size: 5 percent
  Priority low
    PLP high: 1, PLP low: 1, TCP PLP high: 1, TCP PLP low: 1
  ...

```



## RELATED DOCUMENTATION

[System Requirements for CoS with DiffServ for IPv6 | 651](#)

[Roadmap for Configuring CoS with IPv6 DiffServ | 661](#)



# 3

PART

## Configuring Platform-Specific Functionality

---

[Configuring Class of Service on ACX Series Universal Metro Routers | 694](#)

[Configuring Class of Service on MX Series 5G Universal Routing Platforms | 753](#)

[Configuring Class of Service on PTX Series Packet Transport Routers | 785](#)

---



# Configuring Class of Service on ACX Series Universal Metro Routers

## IN THIS CHAPTER

- [CoS on ACX Series Routers Features Overview | 694](#)
- [Understanding CoS CLI Configuration Statements on ACX Series Routers | 697](#)
- [DSCP Propagation and Default CoS on ACX Series Routers | 700](#)
- [Configuring CoS on ACX Series Routers | 704](#)
- [Classifiers and Rewrite Rules at the Global, Physical, and Logical Interface Levels Overview | 705](#)
- [Configuring Classifiers and Rewrite Rules at the Global and Physical Interface Levels | 707](#)
- [Applying DSCP and DSCP IPv6 Classifiers on ACX Series Routers | 709](#)
- [Schedulers Overview for ACX Series Routers | 710](#)
- [Shared and Dedicated Buffer Memory Pools on ACX Series Routers | 714](#)
- [CoS for PPP and MLPPP Interfaces on ACX Series Routers | 717](#)
- [CoS for NAT Services on ACX Series Routers | 730](#)
- [Hierarchical Class of Service in ACX Series Routers | 731](#)
- [Storm Control on ACX Series Routers Overview | 751](#)

## CoS on ACX Series Routers Features Overview

### IN THIS SECTION

- [CoS on ACX routers running Junos OS | 695](#)
- [CoS on ACX routers running Junos OS Evolved | 696](#)





**NOTE:** Unless otherwise noted in the following topics, CoS on ACX Series Universal Metro Routers functions the same as CoS on other routers, and information common CoS functionality can be found in the Configuring Class of Service section of [Class of Service User Guide \(Routers and EX9200 Switches\)](#).

## CoS on ACX routers running Junos OS

The following key CoS features are supported on ACX Series Universal Metro Routers:

- Physical interface-based classifiers at the `[edit class-of-service interfaces interfaces-name]` hierarchy level
- Fixed classification for all ingress packets traversing a *logical interface* to a single forwarding class. Fixed classification is supported on all interface types.
- EXP bits located in each MPLS label and used to encode the CoS value of a packet as it traverses a label-switched path (LSP). To configure global EXP bits, include the `exp` statement at the `[edit class-of-service system-defaults classifiers]` hierarchy level.
- *Rewrite rules* at the physical and logical interface levels including the following: IP type-of-service (ToS), DSCP, MPLS EXP bit value, and IEEE 802.1p bit value.
- Attachment of the following rewrite rules to the physical interface at the `[edit class-of-service interfaces interface-name rewrite-rules]` hierarchy level: IP ToS, DSCP, and IEEE 802.1p bit value.
- Rewrite rules for MPLS EXP bits on the logical interface at the `[edit class-of-service interfaces interface-name unit unit-number rewrite-rule]` hierarchy level.



**NOTE:** Fine-grained rewrite is not possible, even when you use multifield filters, because of the application-specific integrated circuit (ASIC) limitation.

Queuing and scheduling features include:

- Support for up to eight forwarding classes.
- Support for up to eight egress queues per port.
- Internal buffer of 2 MB with per-egress queue buffer management.
- Three weighted random early detection (WRED) curves for TCP and one WRED curve for non-TCP. There are two fill levels and two drop probabilities per WRED curve; the drop probability corresponding to the first fill must be zero.



- Strict-priority and weighted deficit round-robin scheduling.
- Multiple strict-priority queues per port.
- Per-queue committed information rate (CIR) and peak information rate (PIR).
- Per-physical-port shaping.
- Up to four levels of hierarchical scheduling, depending on the platform.

Queue statistics features include:

- Per-egress-queue enqueue statistics in packets, bytes, packets per second (pps), and bits per second (bps).
- Per-egress-queue transmit statistics in packets, bytes, pps, and bps.
- Per-egress-queue drop statistics in packets and pps.

## CoS on ACX routers running Junos OS Evolved

ACX7000 routers support the following CoS features:

- Behavior aggregate (BA) classifiers (Inet-Prec/DSCP/DSCP-v6/EXP/IEEE-802.1p/IEEE-802.1ad) at the logical interface level
- Fixed and multifield classifiers
- Maximum of eight forwarding classes
- Eight virtual output queues (VOQ) per port
- Eight scheduling priority levels
- Multiple strict-high priority queues with round robin (RR) scheduling
- Multiple low priority queues with weighted fair queue (WFQ) scheduling
- CIR and PIR for low priority queues
- PIR at the port level
- Rewrite rules (Inet-Prec/DSCP/DSCP-v6/EXP/IEEE-802.1p/IEEE-802.1ad) at the logical interface level
- Hierarchical class of service (HCoS) features including:
  - Four levels of hierarchical scheduling (physical interfaces, logical interface sets, logical interfaces, queues)



- Support for Layer 3 VPN, Layer 2 VPN, Layer 2 Circuit, VPLS, and EVPN services
- Per-queue buffer configuration
- HCoS on aggregated Ethernet (ae-) interfaces except on logical interface sets
- Six scheduling priority levels (low-latency, strict-high, high, medium-high, medium-low, low (low-high, low-medium, low))
- Host outbound traffic forwarding class assignment and global default IEEE 802.1p rewrite



**NOTE:** To apply CoS to an aggregated Ethernet (AE) interface, do not configure CoS on a physical interface and then add the physical interface to the AE interface. First configure the AE interface, then configure CoS on the AE interface.



**NOTE:** ACX routers running Junos OS Evolved do not support the following CLI show commands:

- show class-of-service forwarding-table
- show class-of-service routing-instance
- show class-of-service system-defaults

## RELATED DOCUMENTATION

[Understanding CoS CLI Configuration Statements on ACX Series Routers | 697](#)

[Configuring CoS on ACX Series Routers | 704](#)

[Hierarchical Class of Service in ACX Series Routers | 731](#)

## Understanding CoS CLI Configuration Statements on ACX Series Routers

ACX Series Universal Metro Routers have some statements or statement options supported on other platforms that are not supported or may not have effect on ACX Series devices.

The following CLI options are not applicable to ACX Series Universal Metro Routers:

```
[edit class-of-service schedulers scheduler-name priority-level]
low;
```



```
medium-low;
medium-high;
high;
```

Configure the strict-high-priority queue with unlimited transmission bandwidth so that all traffic receives precedence over any non strict-high priority queues.

At the **[edit class-of-service classifiers *type classifier-name*]** hierarchy level, the **dscp-ipv6** and **ieee-802.1ad** classifier types are not supported. For the dscp classifier type, only the outer tag is supported.

The following CLI stanza is not applicable to ACX Series Universal Metro Routers.

```
[edit class-of-service interfaces interface-name]
irb {
  unit logical-unit-number {
    classifiers {
      type (classifier-name | default);
    }
    rewrite-rules {
      dscp (rewrite-name | default);
      dscp-ipv6 (rewrite-name | default);
      exp (rewrite-name | default) protocol protocol-types;
      ieee-802.1 (rewrite-name | default) vlan-tag (outer | outer-and-inner);
      inet-precedence (rewrite-name | default);
    }
  }
}
```

The following CLI statements are not applicable to ACX Series Universal Metro Routers.

```
[edit class-of-service routing-instances routing-instance-name]
```

```
[edit class-of-service scheduler-map-chassis map-name]
```

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
input-shaping-rate (percent percentage | rate);
input-traffic-control-profile profiler-name shared-instance instance-name;
output-traffic-control-profile profile-name shared-instance instance-name;
```



```
per-session-scheduler;
scheduler-map map-name;
shaping-rate rate;
```

```
[edit class-of-service interfaces iinterface-name unit logical-unit-number]
  classifiers {
    type (classifier-name | default);
  }
  rewrite-rules {
    dscp (rewrite-name | default);
    dscp-ipv6 (rewrite-name | default);
    exp (rewrite-name | default) protocol protocol-types;
    exp-push-push-push default;
    exp-swap-push-push default;
    ieee-802.1 (rewrite-name | default) vlan-tag (outer | outer-and-inner);
    inet-precedence (rewrite-name | default);
  }
```

In the above stanza, **[edit class-of-service *interface-name* unit *logical-unit-number* rewrite-rule exp (*rewrite-name* | default)]** is supported. However, edit **[class-of-service *interface-name* unit *logical-unit-number* rewrite-rule exp protocol *protocol type*]** is not supported.

```
[edit class-of-service interfaces interface-name interface-set interface-set-name]
excess-bandwidth-share;
internal-node;
output-traffic-control-profile profile-name;
output-traffic-control-profile-remaining profile-name;
```

## RELATED DOCUMENTATION

[CoS on ACX Series Routers Features Overview | 694](#)

[Configuring CoS on ACX Series Routers | 704](#)



## DSCP Propagation and Default CoS on ACX Series Routers

### IN THIS SECTION

- [Benefits of DSCP Propagation and Default CoS Support for EVPN VXLANs | 700](#)
- [Default CoS and DSCP propagation on VXLAN interfaces | 700](#)
- [Limitations | 703](#)

Junos OS Evolved uses default CoS and rewrite functions on ACX Series routers. As a result, Differentiated Services code point (DSCP) can't propagate from the native payload to the VXLAN tunnel header.

Starting in Junos OS Evolved release 22.1R1, use our new command `set system packet-forwarding-options no-ip-tos-rewrite` to allow DSCP propagation for VXLAN traffic. When you configure or delete DSCP propagation, the `evo-pf` process restarts.

The new CLI command can handle IP code point propagation, but VLAN Priority Code Point (PCP) propagation is not supported.

### Benefits of DSCP Propagation and Default CoS Support for EVPN VXLANs

The `set system packet-forwarding-options no-ip-tos-rewrite` command enables the DSCP field from the incoming packet to propagate to the VXLAN encapsulated packet. This allows CoS to be applied to VXLAN encapsulated packets as they traverse the network.

### Default CoS and DSCP propagation on VXLAN interfaces

In previous releases, the DSCP information in the incoming (native) payload does not propagate to the VXLAN tunnel header. Therefore, the DSCP of the VXLAN packet is set to zero. With the new CLI command, the DSCP of the native payload propagates to the VXLAN tunnel header.

The Layer 3 classifier (`inet-precedence`, DSCP, DSCPv6) at the ingress interface classifies packets to a forwarding class and loss priority. With the new command, the Layer 3 classifier classifies packets using the DSCP of the native packet since the DSCP of the native payload is always retained.

The [Table 64 on page 701](#) provide a summary of the CoS, Layer 3, and L3 and L2 classifier behavior when you enable the CLI command.



**Table 64: CoS Behavior with CLI Command**

CoS Behavior with CLI Command	CoS Behavior without CLI Command
Disables L3 rewrite	Existing CoS function
Retains the DSCP of the native payload	Retains the DSCP of the native payload
Enables DSCP propagation for both VXLAN L2 and VXLAN L3, with the L3 classifier explicitly attached to the user-to-network (UNI) interface for an L2.	DSCP of the VXLAN packet is set to zero

### Layer 3 Classifier Behavior

**Table 65: Layer 3 Classifier Behavior**

Ingress Classifier Type	Egress Rewrite Type	Ingress Packet DSCP	Support for Rewrite as per Configured Rewrite
L3 (DSCP, DSCP IPv6, IP Precedence) on L2 or L3 IFL	L3	not applicable	Preserves only the configured rewrite
L3 (DSCP, DSCP IPv6, IP Precedence) on L2 or L3 IFL	IEEE 802.1p/ad, EXP	not applicable	Preserves only the configured rewrite  Does not allow rewrite or preserve the configured rewrite for IEEE 802.1p and IEEE 802.1pad/EXP
L2 (IEEE 802.1p/ad) on L2 IFL	IEEE 802.1p/ad	not applicable	Supports rewrite and preserves if DSCP is present
MPLS EXP	IEEE 802.1p/ad, MPLS EXP	not applicable	Yes



**Table 65: Layer 3 Classifier Behavior (Continued)**

Ingress Classifier Type	Egress Rewrite Type	Ingress Packet DSCP	Support for Rewrite as per Configured Rewrite
L3 (DSCP, DSCP IPv6, IP Precedence)	IEEE 802.1p/ad, EXP	0 to 7	<p>The rewrite configuration is preserved if the ingress DSCP matches the forwarding class id used in the rewrite rule. For IEEE 802.1p, EXP rewrite occurs per the rewrite rule IEEE 802.1p, EXP rewrite occurs as per the rewrite rule (forwarding class, loss priority to IEEE 802.1p/EXP).</p> <p>For IEEE 802.1p, the EXP bits are set to zero.</p>
L3 (DSCP, DSCP IPv6, IP Precedence)	IEEE 802.1p/ad, EXP	More than 7	For IEEE 802.1p, the EXP bits are set to zero

The [Table 66 on page 702](#) provides a summary of the L3 and L2 classifier behavior when you enable the CLI command.

**Table 66: L3 and L2 Classifier Behavior with CLI Command**

Input Packet	Type of Interface at UNI	Classifier	User Configuration	Ingress Remark	Egress Rewrite	VXLAN Encapsulated Packet DSCP	Description
L3 IPv4 and IPv6	IRB/L3 UNI	DSCP default	None	No	No	PCP/DEI of inner packet	None
L2 carrying IPv4 and IPv6	L2	PCP default	DSCP classifier	No	No	PCP/DEI of inner packet	User needs to configure DSCP classifier explicitly on L2 UNI.



Table 66: L3 and L2 Classifier Behavior with CLI Command *(Continued)*

Input Packet	Type of Interface at UNI	Classifier	User Configuration	Ingress Remark	Egress Rewrite	VXLAN Encapsulated Packet DSCP	Description
Pure L2 (no IP payload)	L2	PCP default	DSCP classifier	No	No	PCP /DEI of inner packet having VLAN tag  Example: PCP/DEI bits = 100/1 => DSCP = 1001 =9 (decimal); DSCP =0 with untagged packet; DSCP=0 with native tagging	User needs to configure DSCP classifier explicitly on L2 UNI; On QFX, DSCP=0 on VXLAN encapsulated packet irrespective of single tagged, untagged and native tagged packets at L2 UNI.

## Limitations

When you enable the new CLI command on an ACX Series router:

- The IP DSCP or IP Precedence rewrite is disabled in the system and network. As a result, you can't execute L3 rewrite (DSCP, DSCP IPv6, IP precedence).
- IEEE 802.1p code point bits in the packet header retain their original header. IEEE 802.1p propagation is not supported.

## RELATED DOCUMENTATION

[Configuring CoS on ACX Series Routers | 704](#)

[Applying DSCP and DSCP IPv6 Classifiers on ACX Series Routers | 709](#)



## Configuring CoS on ACX Series Routers

Physical interface-based classifiers are supported at the [edit class-of-service interfaces *interfaces-name*] hierarchy level. EXP bits are located in each MPLS label and used to encode the CoS value of a packet as it traverses an LSP. To configure global EXP bits, include the `exp` statement at the [edit class-of-service system-defaults classifiers] hierarchy level.

To configure CoS on ACX Series routers:

1. Configure the class of service.

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

2. Configure the rewrite rules.

```
[edit class-of-service]
user@host# edit rewrite-rules (dscp | inet-precedence) rewrite-name
user@host# edit forwarding-class class-name
user@host# set loss-priority low class-name code-points (alias / bits)
```

3. Configure behavior aggregate classifiers for DiffServ CoS.

```
[edit class-of-service]
user@host# edit classifiers (dscp | inet-precedence) classifier-name
user@host# edit forwarding-classes class-name
user@host# set loss-priority class-name code-points (alias / bits)
```

4. Configure expedited forwarding class classifiers.

```
[edit class-of-service classifiers]
user@host# edit forwarding-classes class-name
user@host# set loss-priority class-name code-points (alias / bits)
```

5. Define the forwarding-class mappings.

```
[edit class-of-service]
user@host# edit forwarding-classes class queue-number queue-number
```



## 6. Configure network control forwarding class classifiers.

```
[edit class-of-service]
user@host# edit forwarding-class class-name
user@host# set loss-priority low class-name code-points (alias / bits)
```

## 7. Apply the rewrite rules and classifiers to the interfaces.

```
[edit class-of-service interface interface-name unit unit-number]
user@host# set rewrite-rule (dscp | inet-precedence ) (rewrite-name| default)
user@host# set classifiers (dscp | inet-precedence ) classifier-name | default)
```

## 8. Set the global system default.

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

## RELATED DOCUMENTATION

[CoS on ACX Series Routers Features Overview | 694](#)

[Understanding CoS CLI Configuration Statements on ACX Series Routers | 697](#)

## Classifiers and Rewrite Rules at the Global, Physical, and Logical Interface Levels Overview

On most ACX Series Universal Metro Routers and EX Series switches, CoS supports classification and rewrite at the global level and physical interface levels.



**NOTE:** The ACX6360 router does not support rewrite rules or Layer 2 (IEEE802.1p and IEEE802.1ad) classifiers.



**NOTE:** ACX7100 routers support classification and rewrite rules of all types (Inet-Prec/DSCP/DSCP-v6/EXP/IEEE-802.1p/IEEE-802.1ad) at the logical interface level.

At a global level, you can define EXP classification.



At a physical interface level, you can define the following features:

- DSCP, DSCP-IPV6, and IPv4 precedence classifiers
- DSCP, DSCP-IPV6, and IPv4 precedence rewrites
- IEEE 802.1p and IEEE 802.1ad classifiers (inner and outer)
- IEEE 802.1p and IEEE 802.1ad rewrites (outer)

The IEEE 802.1ad classifier uses IEEE 802.1p and DEI bits together.



**NOTE:** You cannot configure both IEEE 802.1p and IEEE 802.1ad classifiers together at the physical interface level.

At a logical interface level, you can define the fixed classification and EXP rewrites.

To configure global EXP classifiers, include the **classifiers exp classifier-name** statement at the **[edit class-of-service system-defaults]** hierarchy level.

To configure classifiers or *rewrite rules* at the physical interface, include either the **classifiers** statement or the **rewrite-rules** statement at the **[edit class-of-service interfaces interface-name]** hierarchy level.

To configure fixed classifiers at the logical interface, include the **forwarding-class fc** or the **rewrite-rules** statement at the **[edit class-of-service interfaces interface-name unit number]** hierarchy level.

To configure EXP rewrite at the logical interface, include the **[edit class-of-service interfaces interface-name unit number rewrite-rules exp rewrite-rule]** statement.

To display classifiers configured under **system-defaults**, enter the **show class-of-service system-defaults** command.

To display classifiers and rewrite rules bound to physical interfaces, enter the **show class-of-service interfaces interface-name** command.

## RELATED DOCUMENTATION

[Configuring Classifiers and Rewrite Rules at the Global and Physical Interface Levels](#) | 707



## Configuring Classifiers and Rewrite Rules at the Global and Physical Interface Levels

On ACX Series Universal Metro Routers and EX Series switches, CoS supports classification and rewrite at the global and physical interface levels.

To configure the global EXP classifier, include the following statements at the **[edit class-of-service] system-defaults** hierarchy level.

```
[edit class-of-service]
{
  system-defaults {
    classifiers exp classifier-name
  }
}
```

CoS supports one global system default classifier of the EXP type, as shown in the following example:

```
[edit class-of-service]
{
  system-defaults {
    classifiers {
      exp exp-classf-core;
    }
  }
}
```

To configure classifiers and rewrite rules at the physical interface level, include the following statements at the **[edit class-of-service] interfaces** hierarchy level.

```
[edit class-of-service]
interfaces {
  interface-name
    classifiers dscp classifier-name
    classifiers inet-precedence classifier-name
    classifiers ieee-802.1 [vlan-tag (outer | inner)] classifier-name
    rewrite-rules dscp rewrite-name
    rewrite-rules inet-prec rewrite-name
```



```

        rewrite-rules ieee-802.1 rewrite-name
    }

```

The following example shows classifiers and rewrite rules configured on physical interfaces:

```

ge-0/1/0 {
    unit 0 {
        rewrite-rules {
            exp custom-exp;
        }
    }
    classifiers {
        dscp d1;
        ieee-802.1 ci;
    }
    rewrite-rules {
        dscp default;
    }
}

ge-0/1/2 {
    classifiers {
        ieee-802.1 ci;
    }
    rewrite-rules {
        ieee-802.1 ri;
    }
}

ge-0/1/3 {
    unit 0 {
        rewrite-rules {
            exp custom-exp2;
        }
    }
}

ge-0/1/7 {
    classifiers {
        dscp d1;
    }
}

ge-0/1/8 {
    classifiers {
        dscp d1;
    }
}

```



```
}
}
```

## RELATED DOCUMENTATION

[Classifiers and Rewrite Rules at the Global, Physical, and Logical Interface Levels Overview](#) | 705

## Applying DSCP and DSCP IPv6 Classifiers on ACX Series Routers

For ACX Series routers, you cannot apply separate DSCP and DSCP IPv6 classifiers for IPv4 and IPv6 packets on a physical interface. Instead, classifier assignment works as follows:

- If you assign a DSCP classifier only, IPv4 and IPv6 packets are classified using the DSCP classifier.
- If you assign a DSCP IPv6 classifier only, IPv4 and IPv6 packets are classified using the DSCP IPv6 classifier.
- If you assign an IP precedence classifier only, IPv4 and IPv6 packets are classified using the IP precedence classifier. In this case, the lower three bits of the DSCP field are ignored because IPv4 precedence mapping requires the upper three bits only.
- You can assign either DSCP, DSCP IPv6, or IPv4 precedence classifier types on a physical interface.
- You can configure either DSCP, DSCP IPv6, or IPv4 precedence rewrite rules on a physical interface. The rewrite rule applies to both IPv4 and IPv6 packets.
- If you assign either the DSCP or the IP precedence classifier in conjunction with the DSCP IPv6 classifier, the commit fails.

## RELATED DOCUMENTATION

[Default DSCP and DSCP IPv6 Classifiers](#) | 49



## Schedulers Overview for ACX Series Routers

### IN THIS SECTION

- [Scheduling on ACX Routers Running Junos OS | 710](#)
- [Scheduling on ACX Routers Running Junos OS Evolved | 711](#)

You use *schedulers* to define the properties of output queues. These properties include the amount of interface bandwidth assigned to the queue, the size of the memory buffer allocated for storing packets, the priority of the queue, and the random early detection (RED) drop profiles associated with the queue.

You associate the schedulers with forwarding classes by means of *scheduler maps*. You can then associate each scheduler map with an interface, thereby configuring the hardware queues, packet schedulers, and RED processes that operate according to this mapping.

In ACX Series routers, you can configure more than one strict-priority queue per port. The hardware services the queues in the descending order of queue numbers marked as strict priority. All the strict-priority queues are given preferential treatment by the scheduler as long as their shaping rates (or peak information rates) are not met. Unlike MX Series routers, the ACX Series routers configured with queues as *strict-high* at the [edit class-of-service schedulers *scheduler-name* priority *strict-high*] statement hierarchy, the service is based on queue number and not based on sharing the *strict-high* queues.

### Scheduling on ACX Routers Running Junos OS

Unlike other ACX Series routers, ACX5048 and ACX5096 router supports CIR among strict-priority queues. There is no implicit queue number-based priority among the strict-priority queues. Unlike other ACX Series routers, ACX5048 and ACX5096 router supports configuring drop profiles for loss-priority low, medium-high, and high for non-TCP protocols as well.



**NOTE:** The options `buffer-partition multicast percent <0-100>` at the [edit class-of-service schedulers *scheduler-name* buffer-size] hierarchy level and `multicast <0-100>` at the [edit class-of-service schedulers *scheduler-name* shared-buffer-maximum] hierarchy level are supported only on ACX5048 and ACX5096 routers. For more information, see ["Shared and Dedicated Buffer Memory Pools on ACX Series Routers"](#) on page 714.

ACX5448 routers support port-based queueing, scheduling, and shaping. You can configure up to eight queues (virtual output queues) per physical interface (port). Scheduling properties can be applied at both



physical as well as logical interface levels. The egress scheduler supports two priority levels (strict-high and low). Multiple strict-high priority queues and multiple low (default) priority queues can be configured.

By default a port on an ACX5448 router gets a dedicated buffer of 100 microseconds and shared buffer from DRAM. Delay buffer controls the latency of the queue during congestion and maximum number of packets that can be held in a queue. Default buffer size per port is 100 microseconds.



**NOTE:** On all ACX Series routers running Junos OS, the strict priority queues cannot have transmit-rate configured.



**NOTE:** On ACX710 routers, class-of-service commit changes, particularly for schedulers on a physical interface, cause very short (less than one second) pauses in forwarding packets. This can cause packet loss. For some protocols that use very short keepalive intervals, such as BFD, we recommend you disable or enlarge the keepalive interval before you apply scheduler changes. We also recommend you configure scheduling before you enable live traffic.



**NOTE:** On ACX 4000 routers, whenever the scheduling and shaping parameters of a port or any of its queues are changed, the entire scheduling configuration on the port is erased and the new configuration is applied. During this window, the traffic pattern does not adhere to user parameters. We recommend you configure scheduling before you enable live traffic.

## Scheduling on ACX Routers Running Junos OS Evolved

All ACX7000 routers run Junos OS Evolved.

### Scheduling Priorities

ACX7000 routers support eight scheduler priority levels for port-level scheduling and six levels for hierarchical scheduling. ACX7000 routers also support multiple queues with the same priority.

**Table 67: Scheduling Priority Support**

Priority	Port Scheduling	Hierarchical Scheduling
low-latency	Yes	Yes



Table 67: Scheduling Priority Support (*Continued*)

Priority	Port Scheduling	Hierarchical Scheduling
strict-high	Yes	Yes
high	Yes	Yes
medium-high	Yes	Yes
medium-low	Yes	Yes
low-high	Yes	No
low-medium	Yes	No
low	Yes	Yes



**NOTE:** ACX routers *do not* guarantee round-robin distribution between same priority queues.



**NOTE:** On all ACX routers running Junos OS Evolved, you can only configure transmit-rate on low priority queues.

## Low Latency Queuing (LLQ)

ACX7000 routers support low latency queuing (LLQ). LLQ enables delay-sensitive data to have preferential treatment over other traffic. A low-latency queue has the highest priority over any other priority queues, including strict-high queues, as well as a low delay scheduling profile.

For port scheduling of virtual output queues (VOQs), low latency VOQs receive their own dedicated egress queue. High priority VOQs receive a second dedicated egress queue, and low priority VOQs receive a third dedicated egress queue.

Due to the scheduling hierarchy of hierarchical class of service (HCoS), a hierarchical scheduling can use a maximum of two egress queues. Therefore for hierarchical scheduling of VOQs, low latency VOQs and



high priority VOQs receive a common dedicated egress queue, and low priority VOQs receive the second dedicated egress queue.



**NOTE:** We recommend the following when configuring low-latency VOQs:

- Use policers to normalize the burstiness of traffic before it reaches a low-latency VOQ.
- Configure a maximum of two low-latency VOQs on a physical or logical interface.
- Classify and schedule traffic (that is, reserve bandwidth) for low-latency VOQs so that there is no congestion for those queues.



**NOTE:** Low-latency queues receive the same buffers as other queues to efficiently use the limited hardware VOQ buffer profiles.

Change History Table

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

Release	Description
Junos OS Evolved 23.4R1	ACX7000 routers support six scheduler priority levels for port scheduling and hierarchical scheduling.
Junos OS Evolved 23.4R1	ACX7000 routers support low latency queuing (LLQ).
Junos OS Evolved 24.4R1	ACX7000 routers support eight priority levels for port scheduling.

RELATED DOCUMENTATION

[RED Drop Profiles for Congestion Management | 464](#)

[Shared and Dedicated Buffer Memory Pools on ACX Series Routers | 714](#)

[Virtual Output Queues on PTX Series Packet Transport Routers | 797](#)



## Shared and Dedicated Buffer Memory Pools on ACX Series Routers

The ACX5048 and ACX5096 router has 12 megabytes of Packet Forwarding Engine (PFE) wide common packet buffer memory that is used to store packets on interface queues. The buffer memory is divided into two pools, shared buffers and dedicated buffers or reserved buffers.

Shared buffers are a global memory pool that the router allocates dynamically to ports as needed, so the buffers are shared among the ports. To configure a maximum amount of shared buffer that the multicast packets can consume, include the `multicast percentage` CLI statement at the `[edit class-of-service schedulers scheduler-name shared-buffer maximum]` hierarchy level. The value that you can specify for `multicast percentage` CLI command can be from 0 through 100 percent. If the `multicast percentage` CLI statement is not added, then the value defined by the `shared-buffer maximum percent percentage` is used for multicast packets as well.

Dedicated buffers or reserved buffers are a memory pool divided equally among the router ports. Each port receives a minimum guaranteed amount of buffer space, dedicated to each port, not shared among ports. To configure a dedicated buffer for multicast packets, include the `buffer-partition multicast percentage` CLI statement at the `[edit class-of-service schedulers scheduler-name buffer-size]` hierarchy level. The value that you can specify for `buffer-partition multicast percentage` CLI command can be from 0 through 100 percent. If the `buffer-partition multicast percentage` CLI statement is not configured, then a default value of 25% is reserved for multicast packets.



**NOTE:** The total amount of actual queue buffer is defined using the `buffer-size` CLI command.

The ACX5448 router supports delay bandwidth buffer (DBB) for virtual output queues (VOQs). ACX5448 router supports an external DRAM memory, as well as an on-chip buffer (OCB) for storing packet data. A packet is either fully stored in the DRAM or fully stored in the OCB and can consume one or more buffers (upto 40 buffers) depending on the packet size versus the buffer size. A buffer contains a single packet or a part of a single packet.



**NOTE:** The ACX5448 router does not support buffering for IRB multicast traffic and therefore CLIs for configuring multicast is not supported.

The ACX5448 router does not support `buffer-partition multicast percent percentage` option for `buffer-size` and `multicast percentage` option for `shared-buffer-maximum`.

To configure shared and dedicated buffers, include the `multicast percentage` and `buffer-partition multicast percentage` CLI statements at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
schedulers {
```



```

scheduler-name {
    buffer-size (percent percentage | remainder | temporal microseconds | buffer-partition
multicast percent percentage
);
    shared-buffer maximum (percent percentage | multicast percentage);
}
}

```

The following is a sample configuration for shared and dedicated buffers in ACX5048 and ACX5096 routers:

```

[edit class-of-service]
schedulers schd1{
    buffer-size percent 80;
    buffer-partition {
        multicast {
            percent 30;
        }
    }
    shared-buffer {
        maximum {
            20;
            multicast {
                10;
            }
        }
    }
}
}

```

The port gets 50 microseconds worth of reserved buffer. For a 10 Gigabyte port without any shaper, this translates to 62500 Bytes.

In the above sample configuration, the total buffer size allotted for the queue is 80 percent.

Under buffer-partition, the multicast packets get 30 percent of the total buffer-size, which translates to about 24 percent of port-buffer. The unicast packets get the remaining 70 percent of 80 percent of the port buffer, which translates to 56 percent of port buffer.

Under shared-buffer, the multicast packets get up to 10 percent of the total shared buffer. Unicast packets use up to 20 percent of the total shared buffer.



The following is a sample configuration for shared and dedicated buffers in ACX5448 router:

```
[edit class-of-service]
schedulers schd1{
  buffer-size percent 80;
  shared-buffer {
    maximum {
      20;
    }
  }
}
```

The ACX5448 router has OCB size of 16MB and DRAM size of 6GB. The default buffer size per port is 100 microseconds. The total buffer size for 48X10GE + 4X100GE comes to 11MB. The ACX5448 router supports deep buffering of oversubscribed traffic using external DRAM to queue traffic to oversubscribed ports. The ACX5448 router uses the DRAM-Mix mode by default, which uses DRAM buffers during oversubscription cases. The ACX5448 router supports configuring buffer size (dedicated buffers) per egress queue, which is similar to ACX5000 line of routers.

The ACX6360 router has a 39 MB total switch buffer pool. By default, 15 percent of the total buffer pool is allocated to the dedicated buffer pool and the remaining is allocated to the shared buffer pool. If you configure the shared buffer pool as less than 100 percent of the available buffer pool, the remaining buffer space is added to the dedicated buffer pool. You can distribute the shared buffer pool among lossless, lossy, and multicast queues with the following configuration:

```
[edit class-of-service shared-buffer]
user@router# set egress percent 100
user@router# set egress buffer-partition lossless percent percent-value
user@router# set egress buffer-partition lossy percent percent-value
user@router# set egress buffer-partition multicast percent percent-value
```

## RELATED DOCUMENTATION

[Schedulers Overview for ACX Series Routers](#) | 710



## CoS for PPP and MLPPP Interfaces on ACX Series Routers

### IN THIS SECTION

- [Limitations That are Common for CoS on PPP and MLPPP Interfaces | 717](#)
- [Limitations for CoS on PPP Interfaces | 719](#)
- [Guidelines for Configuring CoS on PPP and MLPPP Interfaces | 719](#)
- [Limitations for CoS on MLPPP Interfaces | 720](#)
- [CoS Functionalities for IPv4 Over PPP Interfaces | 721](#)
- [CoS Functionalities for IPv4 Over MLPPP Interfaces | 724](#)

Junos CoS enables you to divide traffic into classes and offer various levels of throughput and packet loss when congestion occurs. This functionality allows packet loss to happen according to rules that you configure. The Junos CoS features provide a set of mechanisms that you can use to provide differentiated services when best-effort traffic delivery is insufficient.

CoS functionalities are supported on PPP and MLPPP interfaces. Up to four forwarding classes and four queues are supported per logical interface for PPP and MLPPP packets.



**NOTE:** CoS for PPP and MLPPP Interfaces is not applicable on ACX5048 and ACX5096 routers.

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

### Limitations That are Common for CoS on PPP and MLPPP Interfaces

The following restrictions apply for configuring CoS on PPP and MLPPP interfaces on ACX Series routers:

- For interfaces with PPP encapsulation, you can configure interfaces to support the IPv4, Internet Protocol Control Protocol (IPCP), PPP Challenge Handshake Authentication Protocol (CHAP), and Password Authentication Protocol (PAP) applications.



- Drop timeout, which defines a recovery method for any packets dropped by the member links in a link services or multilink bundle, is not applicable for ACX routers.
- Loss of traffic occurs during a change of CoS configuration; you cannot modify scheduling attributes instantaneously. The link moves to the down state for PPP, and the protocol is denoted as down for MLPPP interfaces.
- Scheduling and shaping capabilities are based on the CIR-EIR model and are not in accordance with the weighed fair queuing mode. The minimum transmit speed is 32 Kbps, and the minimum difference that can be supported between the transmit rate and shaping rate is also 32 Kbps.
- Buffer size is calculated in terms of packets using 256 bytes as the average packet size. For example, if you configure a 10 percent buffer size for TI interfaces, the buffer allocated as 1.536 Mbps \* (10/100) \* (0.1 sec) = 15360 bits. The following formula computes the configured queue length:

Queue length configured = Buffer/average packet size = (15360/256)/8 = 7.5 = 8 packets.

Because there are no shared buffers, the usage of "buffer-size" and "buffer-size exact" attributes result in the same behavior.

- Only two loss priority levels, namely low and high, are supported. Traffic that arrives from the Packet Forwarding Engine with a medium-high priority is treated as high priority traffic. Although you can configure the medium-high loss priority type when you configure the action for a firewall term, it is considered by the system as high priority traffic.
- A fixed, in-built mapping between forwarding class and queue number as follows is performed: Best-effort is queue 0, expedited-forwarding is queue 1, assured-forwarding is queue 2, and network-control is queue 3
- For WRED configurations, the difference between maximum fill-level and minimum-fill level is a number raised to the power of 2 in terms of number of packets ( $x^2$ ). Otherwise, the lower fill-level is tuned to turn the difference into a value raised to the power of 2. For example, queues contain a size of 64 packets. If the following configuration is performed:

```
fill-level 50 drop-probability 0;
fill-level 100 drop-probability 100;
```

- For the lower fill level, the minimum number of packets is 32. However, if you specify the fill-level to be 45 instead of 50, the lower fill level is 28. Because 64 - 28, which equals 36 is not a power of 2, the lower fill-level is internally adjusted to convert it to be a number exponentially raised to 2.
- When fragmentation-map is configured, the forwarding-class carrying the multi-class 0 traffic must be assigned the highest priority and the forwarding-class carrying the multi-class 3 traffic must be assigned the lowest priority. Such a configuration is necessary because of the NPU design.



## Limitations for CoS on PPP Interfaces

The following restrictions apply for configuring CoS on PPP interfaces on ACX Series routers:

- The distribution of excess rate between 2 or more queues that contain the same priority occurs on a first-come, first-served basis. For example, consider two Queues configured as follows:

Q1 : Transmit-rate = 10%, Shaping-rate = 20%

Q2 : Transmit-rate = 10%, Shaping-rate = 30% on same priority

The excess rate for Q1 =  $(20 - 10) = 10\%$

The excess rate for Q2 =  $(30 - 10) = 20\%$

- The excess rate distribution between Q1 and Q2 does not follow the same ratio but packets in these queues are served in first-come, first-served manner. The shaping rate continues to be honored in such a scenario.

## Guidelines for Configuring CoS on PPP and MLPPP Interfaces

Keep the following points in mind when you configure CoS on PPP and MLPPP interfaces:

- You can configure only the any option with the protocol statement at the [edit class-of-service schedulers *scheduler-name* drop-profile-map] hierarchy level to specify the protocol type for the specified scheduler. You cannot specify the TCP or non-TCP protocol types.
- CoS functionalities for fractional T1 and E1 interfaces are not supported. CoS is supported only for full T1 and E1 interfaces.
- Weighted fair queuing (WFQ) shaping and scheduling model is not supported. Instead of WFQ, CIR-EIR model is supported to handle shaping and scheduling requirements.
- Percentage-based rate configuration is not supported for MLPPP LSQ interfaces; only absolute rate configuration in bps is supported.
- Auto-adjustment of shaping and scheduling rates with the addition or deletion of T1/E1 links is not supported. All the limitations applicable for CoS on ACX routers apply for PPP interfaces.
- All the policer limitations on ACX routers for Gigabit Ethernet interfaces are valid for PPP interfaces. This restriction includes ingress and egress policers. Because these limitations are chassis-wide, they are also effective for PPP interfaces.
- All valid configurations specified for MLPPP interfaces with inet address families are also valid for MLPPP interfaces with MPLS address families. For example, EXP classifier as a global classifier is supported for ingress classification and EXP rewrite rule is supported for egress logical interfaces.



- PPP encapsulation is supported on ACX1000, ACX2000, ACX2100, and ACX4000 routers.
- A maximum of 1000 logical interfaces can be supported on an ACX router .
- A maximum of 280 PPP or MLPPP logical interfaces can support drop-profiles on a system. On each MIC, a maximum of 140 PPP or MLPPP interfaces are supported.

## Limitations for CoS on MLPPP Interfaces

The following restrictions apply for configuring CoS on MLPPP interfaces on ACX Series routers:

- Percentage-based configuration for scheduling and shaping parameters is not supported; only absolute rate configuration is supported. As a result, dynamic, swift readjustment of shaping and scheduling settings does not happen with the addition or deletion of T1/E1 links.
- Buffer size is calculated in terms of a single T1 or E1 link speed. Therefore, a temporal value, in microseconds, is used to compute the buffer size for a higher value of the buffer size. For the temporal setting, the queuing algorithm starts dropping packets when it queues more than a computed number of bytes. This maximum is computed by multiplying the transmission rate of the queue by the configured temporal value. The default queue size and percentage-based queue size are not based on the current bandwidth.
- If you configure a scheduler map without a fragmentation map, any scheduler-map configuration including the default settings are applied the same behavior as the exact transmission rate functionality. Priorities of traffic are not honored and no excess rates are provisioned. The forwarding class with no rate configuration receives the minimum fixed rate allocated to it, which is 32 Kbps.
- Support for oversubscription and priority is not available, which might cause inefficient bandwidth utilization. For example, consider a default scheduler map, with the best-effort queue configured with a rate that is equal to  $16 * T1 * (.95)$  transmit-rate exact.
- The network-control queue is configured with a rate that is equal to  $16 * T1 * (0.05)$  transmit rate exact. In such a scenario, the following behavior is observed for MLPPP bundle with a single T1 link:

Traffic that arrives as only best-effort type of traffic is provided with complete bandwidth capacity if no traffic is distributed to any other queue. Traffic that arrives on the only network-control queue is limited to a bandwidth of 1.2288 Mbps, even if no traffic is present on any other queue. When traffic arrives on both the best-effort and network-control queues, an equal division of traffic is done on both the queues because both the queues are within their minimum guarantee rate. Queues other than Best-Effort and Network-Control receive 32 Kbps of exact transmit bandwidth.

Queues other than best-effort and network-control are assigned 32 Kbps of exact transmit bandwidth.

Consider another example of a default scheduler map, and an MLPPP bundle with two T1 links. In such a scenario, the following behavior is observed for MLPPP bundle with two T1 links:



Traffic that arrives on only the best-effort queue obtains the entire bandwidth capacity if there is no traffic on any other queue. Traffic that arrives on only the network-control queue is limited to 1.2288 Mbps, even when no traffic is passing through any other queue. When traffic arrives on both the queues, it is marked at 1.2288 Mbps for the network-control queue and at 1.843200 Mbps for best-effort queue.

For a default scheduler-map with an MLPPP bundle that contains 16 T1 links, the traffic that arrives as only best-effort traffic receives a bandwidth that is equal to  $(0.95 * 16 * T1)$  capacity if there is no traffic on any other queue. Traffic that arrives as only network-control traffic is limited to 1.2288 Mbps even if no traffic on any other queue is observed. When traffic arrives on both the queues, it is tagged as 1.2288 Mbps for network-control and  $(0.95 * 16 * T1)$  Mbps for best-effort queues. If you configure a scheduler-map with a fragmentation map, two or more classes when configured with same priority receive only the transmit-rate served for them and function as the traffic defined for exact transmit-rate functionality.

Support for oversubscription between two multi-classes on the same priority is not provided. The queue corresponding to which there is no-multiclass entry is moved to the disabled state. Only one-to-one mapping between forwarding-class to multi-class is supported. One forwarding class can send traffic corresponding to only one multi-class.

## CoS Functionalities for IPv4 Over PPP Interfaces

The following CoS capabilities are supported on PPP interfaces for IPv4 traffic:

- Ingress Classification can be either specified as fixed classifiers or behavior aggregate (BA) classifiers. Fixed classifiers map all traffic on an interface to the forwarding class. The forwarding class determines the output queue. To configure a fixed classifier, include the `forwarding-class class-name` statement at the `[edit class-of-service interface interface-name unit logical-unit-number]` hierarchy level.
- BA classification, or CoS value traffic classification, refers to a method of packet classification that uses a CoS configuration to set the forwarding class of a packet based on the CoS value in the IP packet header. The CoS value examined for BA classification purposes can be the Differentiated Services code point (DSCP) value, or the IP precedence value, or EXP bits. The default classifier is based on the IP precedence value.
- To configure the global EXP classifier, include the following statements at the `[edit class-of-service system-defaults]` hierarchy level.

```
[edit class-of-service]
{
  system-defaults
  {
    classifiers exp classifier-name
```



```

    }
}

```

CoS supports one global system default classifier of the EXP type, as shown in the following example:

```

[edit class-of-service]
{
  system-defaults {
    classifiers {
      exp exp-classf-core;
    }
  }
}

```

- All packets that are received on a logical interface in the ingress direction can be classified to a single forwarding class using fixed classification.
- BA Classification based on the following packet fields is supported at the logical interface level:
  - IPv4 - inet-precedence
  - DSCP

The following is the configuration stanza for defining BA classifiers:

```

[edit class-of-service]
  classifiers {
    (inet-precedence|dscp ) classifier-name
    {
      forwarding-class class-name loss-priority [low | high] code-points
    }
    ([ aliases ] | [ dscp-bits ]);
  }

```

- Queuing and scheduling functionalities comprise the following parameters:
  - Transmit rate per queue
  - Shaping rate per queue
  - WRED
  - Forwarding classes. A maximum of 4 forwarding classes can be defined for PPP interfaces.



- The four priorities supported for logical interface-level queuing are strict-high, medium-high, medium-low, or low. The transmit rate per queue (CIR) is the minimum committed rate can be specified for each queue. The shaping rate per queue (PIR) is the maximum transmit rate can be specified for each queue. For WRED, the default behavior is to enable tail drops. The drop profile configuration enables WRED and enables different drop behavior based on the drop precedence to be entered. Loss priority settings help determine which packets are dropped from the network during periods of congestion. The software supports multiple packet loss priority (PLP) designations: low and high
- Buffer-size can be specific in percentage and temporal configuration. This size is turned into the number of packets per queue, with 256 bytes treated as the average packet size. The following settings can be configured at the queue level:
  - Guaranteed transmit rate (CIR)
  - Shaping rate (PIR)
  - Drop profile
  - Buffer size

```
[edit class-of-service]
  schedulers scheduler_name {
    transmit-rate (percent number | rate);
    shaping-rate (percent number | rate);
    buffer-size (percent | temporal)
    drop-profile-map map_name;
  }
```

Only 4 forwarding classes and only 4 queues per logical interface are supported. Also, logical interface-based shaping is not supported.

- Packet and byte counters for transmitted and dropped packets are available per queue. These statistical details are displayed using the show interfaces queue command. Aggregation to provide port-level statistics, if needed, is also supported by the system. The logical interface-level statistics are correctly available for egress direction and are displayed in the output, but the statistics pertaining to dropped packets are not considered because of hardware limitations. The following configuration stanza defines the rewrite rules:

```
[edit class-of-service]
  rewrite-rules {
    (inet-precedence | dscp | exp) rewrite-name
```



```

    {
        forwarding-class class-name loss-priority [low | high] code-point value;
    }
}

```

Each of the rewrite rules can be attached to an interface by using the following configuration syntax:

```
[edit class-of-service interfaces interface-name]
```

```

rewrite-rules {
    dscp (rewrite-name | default) protocol (inet-both | inet-outer | mpls);
    exp (rewrite-name | default) protocol protocol-types;
    inet-precedence (rewrite-name | default) protocol (inet-both | inet-outer | mpls);
}

```

All of the firewall features supported on ACX routers are applicable for PPP interfaces for IPv4 packets.

- For rewrite rules, IPv4 or inet precedence and EXP rules are supported. EXP rewrite rules apply only to logical interfaces. You cannot apply EXP rewrite rules to physical interfaces. There are no default EXP rewrite rules. If you want to rewrite the EXP value in MPLS packets, you must configure EXP rewrite rules and apply them to logical interfaces. If no rewrite rules are applied, all MPLS labels that are pushed have a value of zero (0). The EXP value remains unchanged on MPLS labels that are swapped.

## CoS Functionalities for IPv4 Over MLPPP Interfaces

The following CoS capabilities are supported on MLPPP interfaces for IPv4 traffic:

- Ingress Classification can either be specified as fixed or BA classifiers.
- Fixed classification using forwarding classes and BA classification using IPv4 precedence value are supported.
- The following scheduling and queuing properties are supported:
  - Transmit rate per queue
  - Shaping rate per queue
  - WRED



- Forwarding classes
- Buffer size per queue
- Forwarding class to multilink class mapping. You use the `multilink-class` statement to map a forwarding class into a multiclass MLPPP (MCML).
- All of the firewall features supported on ACX routers are applicable for MLPPP interfaces for IPv4 packets
- For rewrite rules, IPv4 precedence rule is supported.

The following CoS capabilities are supported on MLPPP interfaces for MPLS packets:

- Ingress Classification can either be specified as fixed or BA classifiers. Fixed classification using forwarding classes and BA classification using EXP bits are supported.
- The following scheduling and queuing properties are supported:
  - Transmit rate per queue
  - Shaping rate per queue
  - WRED
  - Forwarding classes
  - Buffer size per queue
- Forwarding class to multilink class mapping. You use the `multilink-class` statement to map a forwarding class into a multiclass MLPPP (MCML).
- All of the firewall features supported on ACX routers are applicable for MLPPP interfaces for IPv4 packets
- For rewrite rules, the EXP rule is supported.

The following example illustrates an MLPPP CoS configuration set:

```
[edit]
class-of-service {
    classifiers {
        inet-precedence all-traffic-inet {
            forwarding-class assured-forwarding {
                loss-priority low code-points 101;
            }
            forwarding-class expedited-forwarding {
```



```

        loss-priority low code-points 000;
    }
}
drop-profiles {
    plp-low-red {
        fill-level 50 drop-probability 0;
        fill-level 100 drop-probability 100;
    }
    plp-high-red {
        fill-level 25 drop-probability 0;
        fill-level 50 drop-probability 100;
    }
}
forwarding-classes {
    queue 0 best-effort;
    queue 1 assured-forwarding;
    queue 2 expedited-forwarding;
    queue 3 network-control;
}

schedulers {
    evdo-mlppp-best-effort {
        transmit-rate 1M;
        buffer-size percent 80;
        priority medium-high;
    }
    evdo-mlppp-assured-forwarding {
        transmit-rate 500000;
        buffer-size percent 10;
        drop-profile-map loss-priority low protocol any drop-profile plp-low-red;
        drop-profile-map loss-priority high protocol any drop-profile plp-high-red;
        priority medium-low;
    }
    evdo-mlppp-expedited-forwarding {
        transmit-rate 300000;
        buffer-size percent 5;
        priority low;
    }
    evdo-mlppp-network-control {
        transmit-rate 200000;
        buffer-size percent 5;
        priority strict-high;
    }
}

```



```

    }
}

scheduler-maps {
    evdo-mlpp-cos-map {
        forwarding-class best-effort scheduler evdo-mlpp-best-effort;
        forwarding-class assured-forwarding scheduler evdo-mlpp-assured-forwarding;
        forwarding-class network-control scheduler evdo-mlpp-network-control;
        forwarding-class expedited-forwarding scheduler evdo-mlpp-expedited-forwarding;
    }
}

fragmentation-maps {
    frag-mlpp {
        forwarding-class {
            assured-forwarding {
                multilink-class 2;
            }
            expedited-forwarding {
                multilink-class 3;
            }
            best-effort {
                multilink-class 1;
            }
            network-control {
                multilink-class 0;
            }
        }
    }
}

interfaces {
    lsq-0/* {
        unit * {
            scheduler-map evdo-mlpp-cos-map;
            fragmentation-map frag-mlpp;
        }
    }
}

```





**NOTE:** In ACX Series routers, the forwarding class and queue mapping is fixed for PPP and MLPPP interfaces.

The following example illustrates an MLPPP firewall configuration set:

```
[edit]
firewall {
    filter classify-evdo-traffic-mlppp {
        interface-specific;
        term signalling {
            from {
                dscp ef;
            }
            then {
                count signalling-counter;
                forwarding-class network-control;
                accept;
            }
        }
        term user-speech {
            from {
                dscp af31;
            }
            then {
                policer user-speech-rate-limit;
                count user-speech-counter;
                forwarding-class network-control;
                accept;
            }
        }
        term ptt-mcs {
            from {
                dscp af11;
            }
            then {
                count ptt-mcs-counter;
                forwarding-class network-control;
                accept;
            }
        }
    }
}
```



```

term user-video {
    from {
        dscp af21;
    }
    then {
        count user-video-counter;
        loss-priority low;
        forwarding-class assured-forwarding;
        accept;
    }
}

term user-cmcs {
    from {
        dscp af12;
    }
    then {
        count user-cmcs-counter;
        loss-priority low;
        forwarding-class assured-forwarding;
        accept;
    }
}

term ran-rlp-retransmission {
    from {
        dscp af41;
    }
    then {
        count rlp-retransmit-counter;
        loss-priority high;
        forwarding-class assured-forwarding;
        accept;
    }
}

term user-best-effort {
    then {
        count user-be-counter;
        forwarding-class best-effort;
        accept;
    }
}
}

```



}

## CoS for NAT Services on ACX Series Routers

For packets that go through NAT services, the classification is done when the packet enters the router, and the resulting forwarding class and the packet loss priority are preserved when the packet goes through the inline services interface. When the packet exits the router through a physical interface, the queuing, scheduling, and rewrite rules configured on the egress physical interface are applied on the basis of the forwarding class and packet loss priority derived during ingress.



**NOTE:** CoS for NAT services is not applicable on ACX5048 and ACX5096 routers.

Queuing and scheduling are supported on the inline services interface to handle congestion at the inline services interface. Congestion at inline services interface is possible when the interface is oversubscribed. In this case, you would need to selectively drop packets based on the packet's forwarding class and packet loss priority. To drop packets based on the packet's forwarding class and packet loss priority, scheduler map configurations are supported on the inline services interface.



**NOTE:** In ACX Series routers, you cannot configure classification and rewrite policies on the inline services interface.

### RELATED DOCUMENTATION

*Network Address Translation Overview on ACX Series*

*Network Address Port Translation Overview*

*Enabling Inline Services Interface on ACX Series*

[Understanding Service Sets](#)

*Service Filters in ACX Series*

*Guidelines for Applying Service Filters*

[Service Filter Match Conditions for IPv4 Traffic](#)

[Service Filter Match Conditions for IPv4 Traffic](#)

*Network Address Translation Address Overload in ACX Series*

[Configuring Address Pools for Network Address Port Translation \(NAPT\) Overview](#)



## Hierarchical Class of Service in ACX Series Routers

### IN THIS SECTION

- [Hierarchical Scheduling on the Physical Interface | 731](#)
- [Traffic Control Profiles | 732](#)
- [Schedulers | 733](#)
- [Deep Buffer Control | 733](#)
- [Drop Profiles | 735](#)
- [Scheduler Maps | 735](#)
- [Applying the Traffic Control Profiles | 735](#)
- [HCoS on Aggregated Ethernet Interfaces | 736](#)
- [Subscriber Services | 737](#)

Scheduling properties can be applied at physical as well as logical interface and logical interface set levels. Service providers can support hierarchical class of service (HCoS) at multiple levels to meet the service level agreements and bandwidth allocations for subscribers.

### Hierarchical Scheduling on the Physical Interface

By default, the queuing mode on all the physical interfaces in ACX routers that support HCoS is 4 queues per physical interface (port). In the hierarchical scheduler mode, you can configure up to 3 levels (physical interface, logical interface, and queues) or 4 levels (including logical interfaces sets) of scheduling, depending on the platform.

You can enable hierarchical scheduling by including the `hierarchical-scheduler` CLI command under the `[edit interfaces interface-name]` hierarchy:

```
[edit]
interfaces xe-0/0/1 {
```



```
hierarchical-scheduler;
}
```



**NOTE:** If you change the physical interface queuing mode from default to hierarchical scheduler mode or vice-versa, the traffic flowing out of the physical interface during the mode change results in a transient loss of traffic data.

## Traffic Control Profiles

Traffic control profiles hold parameters for levels above the queue level of the scheduler hierarchy. You configure the scheduling and shaping on the scheduler nodes using traffic control profiles and scheduler maps for the queue level. The traffic control profile defines the following characteristics of a scheduler node:

- Scheduler-map
- Shaping rate
- Guaranteed rate

You attach traffic control profiles at the physical interface, logical interface set, and logical interface level. You define scheduling and shaping characteristics for the scheduler node using `shaping-rate` and `guaranteed-rate`. The following is a sample traffic control profile configuration:


```
[edit class-of-service traffic-control-profiles]
tcp-500m-shaping-rate {
    shaping-rate 500m;
}
tcp-vlan0 {
    shaping-rate 200m;
    guaranteed-rate 100m;
    scheduler-map tcp-map-vlan0; # Applies scheduler maps to customer VLANs.
}
tcp-vlan1 {
    shaping-rate 100m;
    guaranteed-rate 40m;
    scheduler-map tcp-map-vlan1; # Applies scheduler maps to customer VLANs.
}
```



# Schedulers

A scheduler defines scheduling and queuing characteristics of a queue and holds the information about the queues, the last level of the hierarchy. The following is a sample scheduler configuration:

```
[edit class-of-service schedulers]
sched-vlan0-q0 {
    priority low;
    transmit-rate 20m;
    buffer-size temporal 100ms;
    drop-profile loss-priority low dp-low;
    drop-profile loss-priority high dp-high;
}
sched-vlan1-q1 {
    priority strict-high;
    shaping-rate 20m;
}
```

 **NOTE:** We do not support transmit-rate with priority queues on ACX Series routers.

# Deep Buffer Control

With deep buffer control, also known as fine-grained buffer control, you can configure buffers on a per-queue basis.

On ACX routers running Junos OS Evolved, by default all queues on HQoS enabled logical interfaces have a dedicated buffer of 4KB and a shared buffer of 10msec. [Table 68 on page 733](#) and [Table 69 on page 734](#) show the guaranteed and shared per queue default buffers on ACX routers running Junos OS Evolved.

**Table 68: Guaranteed Buffer per Queue on HCoS-enabled Physical Ports**

Interface Speed (GB)	Minimum Guaranteed Buffer (KB)
1/10	125
25/40/50	625



**Table 68: Guaranteed Buffer per Queue on HCoS-enabled Physical Ports *(Continued)***

Interface Speed (GB)	Minimum Guaranteed Buffer (KB)
100	1250
200	2500
400	5000

**Table 69: Shared Buffer per Queue on HCoS-enabled Physical Ports**

Interface Speed (GB)	Shared buffer size (MB)	Shared buffer size (ms)
1	5	40
10	50	40
25	250	80
40	250	50
50	250	40
100	500	40
200	500	20
400	500	10

You can configure the shared and guaranteed buffers for each queue on HCoS-enabled ports and logical interfaces. See *buffer-size (Schedulers)*.



**NOTE:** Use the `temporal` option to configure shared buffers.



## Drop Profiles

Drop profiles allow you to specify queue specific behavior to drop packets based on WRED profile under congestion. The following is a sample drop profile configuration:

```
[edit class-of-service drop-profiles]
dp-low {
    fill-level 80 drop-probability 0;
    fill-level 100 drop-probability 100;
}
dp-high {
    fill-level 60 drop-probability 0;
    fill-level 80 drop-probability 100;
}
```

## Scheduler Maps

A scheduler map is referenced by traffic control profiles to define queues. The scheduler map establishes the number of queues over a scheduler node, associating a forwarding-class with a scheduler. The following is a sample scheduler map configuration:

```
[edit class-of-service scheduler-maps]
tcp-map-vlan0 {
    forwarding-class voice scheduler sched-vlan0-q0;
    forwarding-class video scheduler sched-vlan0-q1;
    forwarding-class data scheduler sched-vlan0-q2;
}
tcp-map-vlan1 {
    forwarding-class voice scheduler sched-vlan1-q0;
    forwarding-class video scheduler sched-vlan1-q1;
    forwarding-class data scheduler sched-vlan1-q2;
}
```

## Applying the Traffic Control Profiles

You can attach `output-traffic-control-profile` and `output-traffic-control-profile-remaining` at various levels of the scheduler hierarchy to achieve hierarchical class of service. On an interface set, `output-traffic-control-profile-remaining` is active if there is at least one member logical interface with an attached `output-traffic-control-profile` and at least one without.





**NOTE:** Although a shaping rate can be applied directly to the physical interface, hierarchical schedulers must use a traffic control profile to hold this parameter. All logical interfaces that are part of an interface set inherit the traffic control profile of the interface set.

The following is a sample configuration to apply traffic control profiles:

```
[edit class-of-service interfaces]
ge-1/0/0 {
  output-traffic-control-profile tcp-500m-shaping-rate;
  output-traffic-control-profile-remaining tcp-500m-shaping-rate;
  unit 0 {
    output-traffic-control-profile tcp-cvlan0;
  }
  unit 1 {
    output-traffic-control-profile tcp-cvlan1;
  }
}
interface-set svlan0 {
  output-traffic-control-profile tcp-svlan0;
}
interface-set svlan1 {
  output-traffic-control-profile tcp-svlan1;
}
```

## HCoS on Aggregated Ethernet Interfaces

Configuring HCoS on aggregated Ethernet (AE) interfaces is exactly the same as configuring HCoS on physical interfaces. Apply all HCoS configuration to the AE interface.



**NOTE:** You cannot apply HCoS configuration to a physical interface that is a member of an AE interface. To apply HCoS to an AE interface, do not configure HCoS on a physical interface and then add the physical interface to the AE interface. First configure the AE interface, then configure HCoS on the AE interface.

HCoS configuration on an AE interface is automatically applied to each member interface. How the system applies the configuration to each member interfaces (that is, whether the configuration is normalized or not) depends on the AE interface configuration mode:



- Link protection mode — system applies configuration directly to each member interface *without* normalization.
- Non-link protection replica mode — system applies configuration directly to each member interface *without* normalization.
- Non-link protection scale mode (default mode) — system *does* normalize the configuration before applying to each member interface.

HCoS configuration has the following requirements when the configuration must be normalized across member interfaces:

- transmit-rate at scheduler/queue level — both percent and absolute configuration allowed
- shaping-rate at scheduler/queue level — both percent and absolute configuration allowed
- guaranteed-rate at traffic control profile level — only absolute configuration allowed
- shaping-rate at traffic control profile level — only absolute configuration allowed
- shaping-rate at port level — only absolute configuration allowed

If the configuration must be normalized, rates configured as a percent require no normalization. If the configuration must be normalized, rates configured as absolute must be normalized. You can calculate the normalized value for each interface by dividing the absolute configured value by the number of member interfaces in the AE interface.

## Subscriber Services

Supported ACX routers support hierarchical class of service functionality for subscriber services such as Layer 3 VPN, Layer 2 VPN, Ethernet pseudowire (VPWS), and VPLS for logical interface instance on the AC (Access Port).



**NOTE:** Hierarchical class of service is not supported for Layer 2 bridging (bridge domain VLAN) service.

The following sections explain the hierarchical class of service configuration for subscriber services:

### Configuring hierarchical class of service for Layer 3 VPN Service

Supported ACX routers can be configured to provide Layer 3 VPN services to subscribers by connecting the UNI port to a CE device. The physical port can be configured to provide Layer 3 VPN services to multiple subscribers. You can schedule traffic for different Layer 3 VPN instances based on the SLA parameters agreed with the subscriber.



The following is a sample UNI and NNI logical interface configuration on the PE router providing the Layer 3 VPN service:

```
[edit interfaces]
xe-0/0/1 {
  description "NNI IFL";
  unit 0 {
    family inet {
      address 100.1.1.1/24;
    }
    family mpls;
  }
}
ge-0/0/1 {
  description "UNI IFL";
  hierarchical-scheduler;
  flexible-vlan-tagging;
  unit 0 {
    vlan-id 100;
    family inet {
      address 20.20.0.1/24;
    }
  }
  unit 1 {
    vlan-id 101;
    family inet {
      address 20.20.1.1/24;
    }
  }
  unit 2 {
    vlan-id 2;
    family inet {
      address 20.20.2.1/24;
    }
  }
  unit 3 {
    vlan-id 3;
    family inet {
      address 20.20.3.1/24;
    }
  }
  unit 4 {
```



```

        vlan-id 4;
        family inet {
            address 20.20.4.1/24;
        }
    }
    ...
}

```

Scheduling can be enabled on the interfaces to achieve hierarchical class of service support for traffic flowing from NNI towards UNI direction.

### Configuring hierarchical class of service for Layer 2 VPN (Ethernet Pseudowires) Service

Supported ACX routers can be configured to provide Layer 2 VPN services to subscribers based on Ethernet pseudowires where the UNI port is connected to a CE device. The physical port can be configured to provide Layer 2 VPN services to multiple subscribers. You can schedule traffic for different pseudowires based on the SLA parameters agreed with the subscriber. Hierarchical class of service can be enabled per UNI logical interface represented as the attachment point of the Ethernet pseudowire to achieve the functionality.

The following is a sample to configure the UNI logical interface on the PE router providing the Layer 2 VPN service based on Ethernet pseudowire:

```

[edit interfaces]
ge-0/0/1 {
    hierarchical-scheduler;
    vlan-tagging;
    unit 0 {
        encapsulation vlan-ccc;
        vlan-id 0;
    }
    unit 1 {
        encapsulation vlan-ccc;
        vlan-id 1;
    }
    unit 2 {
        encapsulation vlan-ccc;
        vlan-id 2;
    }
    unit 3 {
        encapsulation vlan-ccc;
        vlan-id 3;
    }
}

```



```

    }
    unit 4 {
        encapsulation vlan-ccc;
        vlan-id 4;
    }
}

```

You can enable scheduling on the interfaces to achieve hierarchical class of service for traffic flowing from NNI towards UNI direction.

### Configuring hierarchical class of service for VPLS Service

Supported ACX routers can be configured to provide Layer 2 VPN services to subscribers based on VPLS where the UNI port can be connected to a CE device. Subscriber network is attached to UNI logical interface at the PE router and have a VPLS instance. The same physical port can service multiple VPLS instances for various subscribers. The service provider can schedule traffic for different VPLS instances based on the SLA parameters agreed with the subscriber. You can enable hierarchical class of service per UNI logical interface representing the VPLS instance for the subscriber to achieve the functionality.

The following is a sample to configure the UNI logical interface on the PE router providing the VPLS service:

```

[edit interfaces]
ge-0/0/1 {
    hierarchical-scheduler;
    vlan-tagging;
    encapsulation vlan-vpls;
    unit 0 {
        encapsulation vlan-vpls;
        vlan-id 0;
    }
    unit 1 {
        encapsulation vlan-vpls;
        vlan-id 1;
    }
    unit 2 {
        encapsulation vlan-vpls;
        vlan-id 2;
    }
    unit 3 {
        encapsulation vlan-vpls;
    }
}

```



```

        vlan-id 3;
    }
    unit 4 {
        encapsulation vlan-vpls;
        vlan-id 4;
    }
}

```

Scheduling can be enabled on the interfaces to achieve hierarchical class of service for the traffic flowing from NNI towards UNI direction.

## Verifying the hierarchical class of service configurations

You can use the following CLI commands to verify the configuration:

- `show interfaces queue`—Shows physical interface aggregate, physical interface remaining, and logical interface traffic statistics to monitor the traffic received and transmitted. The following are some sample outputs of `show interfaces queue` CLI command:

```

user@host# run show interfaces queue ge-0/0/4.2
  Logical interface ge-0/0/4.2 (Index 555) (SNMP ifIndex 671)
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Burst size: 0
Queue: 0, Forwarding classes: 8q0
  Queued:
    Packets          :          1121476          7642 pps
    Bytes            :      587885024      32237416 bps
  Transmitted:
    Packets          :          594964          3160 pps
    Bytes            :      304621568      12946280 bps
  Tail-dropped packets : Not Available
  RL-dropped packets  :              0              0 pps
  RL-dropped bytes    :              0              0 bps
  Total-dropped packets:          526512          4482 pps
  Total-dropped bytes :      283263456      19291136 bps
  Queue Buffer Usage:
    Reserved buffer   :              0 pkts              0 bytes
    Shared buffer     :              0 pkts              0 bytes
Queue: 1, Forwarding classes: 8q1
  Queued:
    Packets          :          1121476          7642 pps

```



```

Bytes          :          587885154          32237416 bps
Transmitted:
Packets        :          594959          3160 pps
Bytes          :          304619008          12946280 bps
Tail-dropped packets : Not Available
RL-dropped packets :          0          0 pps
RL-dropped bytes  :          0          0 bps
Total-dropped packets:          526517          4482 pps
Total-dropped bytes :          283266146          19291136 bps
Queue Buffer Usage:
Reserved buffer   :          0 pkts          0 bytes
Shared buffer     :          0 pkts          0 bytes
Queue: 2, Forwarding classes: 8q2
Queued:
Packets          :          1119456          7321 pps
Bytes            :          595127702          31122568 bps
Transmitted:
Packets          :          274601          1500 pps
Bytes            :          140595712          6144000 bps
Tail-dropped packets : Not Available
RL-dropped packets :          0          0 pps
RL-dropped bytes  :          0          0 bps
Total-dropped packets:          844855          5821 pps
Total-dropped bytes :          454531990          24978568 bps
Queue Buffer Usage:
Reserved buffer   :          0 pkts          0 bytes
Shared buffer     :          0 pkts          0 bytes
Queue: 3, Forwarding classes: 8q3
Queued:
Packets          :          1119464          7303 pps
Bytes            :          595131980          31122568 bps
Transmitted:
Packets          :          274602          1500 pps
Bytes            :          140596224          6144000 bps
Tail-dropped packets : Not Available
RL-dropped packets :          0          0 pps
RL-dropped bytes  :          0          0 bps
Total-dropped packets:          844862          5803 pps
Total-dropped bytes :          454535756          24978568 bps
Queue Buffer Usage:
Reserved buffer   :          0 pkts          0 bytes
Shared buffer     :          0 pkts          0 bytes
Queue: 4, Forwarding classes: 8q4

```



## Queued:

Packets	:	1121476	7642 pps
Bytes	:	587885024	32237416 bps

## Transmitted:

Packets	:	594964	3160 pps
Bytes	:	304621568	12946280 bps

Tail-dropped packets : Not Available

RL-dropped packets	:	0	0 pps
--------------------	---	---	-------

RL-dropped bytes	:	0	0 bps
------------------	---	---	-------

Total-dropped packets:		526512	4482 pps
------------------------	--	--------	----------

Total-dropped bytes	:	283263456	19291136 bps
---------------------	---	-----------	--------------

## Queue Buffer Usage:

Reserved buffer	:	0 pkts	0 bytes
-----------------	---	--------	---------

Shared buffer	:	0 pkts	0 bytes
---------------	---	--------	---------

## Queue: 5, Forwarding classes: 8q5

## Queued:

Packets	:	1121476	7660 pps
Bytes	:	587885024	32310560 bps

## Transmitted:

Packets	:	594964	3178 pps
Bytes	:	304621568	13019424 bps

Tail-dropped packets : Not Available

RL-dropped packets	:	0	0 pps
--------------------	---	---	-------

RL-dropped bytes	:	0	0 bps
------------------	---	---	-------

Total-dropped packets:		526512	4482 pps
------------------------	--	--------	----------

Total-dropped bytes	:	283263456	19291136 bps
---------------------	---	-----------	--------------

## Queue Buffer Usage:

Reserved buffer	:	0 pkts	0 bytes
-----------------	---	--------	---------

Shared buffer	:	0 pkts	0 bytes
---------------	---	--------	---------

## Queue: 6, Forwarding classes: 8q6

## Queued:

Packets	:	1121476	7017 pps
Bytes	:	587190842	29535136 bps

## Transmitted:

Packets	:	621684	3589 pps
Bytes	:	318302208	14701712 bps

Tail-dropped packets : Not Available

RL-dropped packets	:	0	0 pps
--------------------	---	---	-------

RL-dropped bytes	:	0	0 bps
------------------	---	---	-------

Total-dropped packets:		499792	3428 pps
------------------------	--	--------	----------

Total-dropped bytes	:	268888634	14833424 bps
---------------------	---	-----------	--------------

## Queue Buffer Usage:

Reserved buffer	:	0 pkts	0 bytes
-----------------	---	--------	---------



```

    Shared buffer      :              0 pkts              0 bytes
Queue: 7, Forwarding classes: 8q7
  Queued:
    Packets           :          1121477          6481 pps
    Bytes              :          586036910        27137704 bps
  Transmitted:
    Packets           :          666066           3660 pps
    Bytes              :          341025792        14994280 bps
  Tail-dropped packets : Not Available
  RL-dropped packets   :              0              0 pps
  RL-dropped bytes     :              0              0 bps
  Total-dropped packets:          455411          2821 pps
  Total-dropped bytes  :          245011118        12143424 bps
Queue Buffer Usage:
  Reserved buffer      :              0 pkts              0 bytes
  Shared buffer        :              0 pkts              0 bytes

```

```

user@host# run show interfaces queue ge-0/0/4 aggregate
Physical interface: ge-0/0/4, Enabled, Physical link is Up
  Interface index: 648, SNMP ifIndex: 1763
  Description:   UNI side - connected to ixia 6/9
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Queue: 0, Forwarding classes: 8q0
  Queued:
    Packets           :          16762731          33205 pps
    Bytes              :          8738362264        139058280 bps
  Transmitted:
    Packets           :          8779233           13724 pps
    Bytes              :          4494967296        56220880 bps
  Tail-dropped packets : Not Available
  RL-dropped packets   :              0              0 pps
  RL-dropped bytes     :              0              0 bps
  Total-dropped packets:          7983498          19481 pps
  Total-dropped bytes  :          4243394968        82837400 bps
Queue Buffer Usage:
  Reserved buffer      :          32 pkts           6656 bytes
  Shared buffer        :          52 pkts          10816 bytes
Queue: 1, Forwarding classes: 8q1
  Queued:
    Packets           :          16762732          33237 pps

```



```

Bytes          :          8738359966          139190408 bps
Transmitted:
Packets        :          8779363          13756 pps
Bytes          :          4495033856          56353008 bps
Tail-dropped packets : Not Available
RL-dropped packets :          0          0 pps
RL-dropped bytes  :          0          0 bps
Total-dropped packets:          7983369          19481 pps
Total-dropped bytes :          4243326110          82837400 bps
Queue Buffer Usage:
Reserved buffer   :          32 pkts          6656 bytes
Shared buffer     :          43 pkts          8944 bytes
Queue: 2, Forwarding classes: 8q2
Queued:
Packets          :          16754769          30221 pps
Bytes            :          8826383526          127522040 bps
Transmitted:
Packets          :          4052168          6369 pps
Bytes            :          2074710016          26095472 bps
Tail-dropped packets : Not Available
RL-dropped packets :          0          0 pps
RL-dropped bytes  :          0          0 bps
Total-dropped packets:          12702601          23852 pps
Total-dropped bytes :          6751673510          101426568 bps
Queue Buffer Usage:
Reserved buffer   :          32 pkts          6656 bytes
Shared buffer     :          24232 pkts          5040256 bytes
Queue: 3, Forwarding classes: 8q3
Queued:
Packets          :          16754937          30328 pps
Bytes            :          8826406908          127965968 bps
Transmitted:
Packets          :          4052173          6378 pps
Bytes            :          2074646336          26134456 bps
Tail-dropped packets : Not Available
RL-dropped packets :          0          0 pps
RL-dropped bytes  :          0          0 bps
Total-dropped packets:          12702764          23950 pps
Total-dropped bytes :          6751760572          101831512 bps
Queue Buffer Usage:
Reserved buffer   :          32 pkts          6656 bytes
Shared buffer     :          24205 pkts          5034640 bytes
Queue: 4, Forwarding classes: 8q4

```



## Queued:

Packets	:	16762735	33406 pps
Bytes	:	8738360722	139886136 bps

## Transmitted:

Packets	:	8779404	13828 pps
Bytes	:	4495054848	56648672 bps

Tail-dropped packets : Not Available

RL-dropped packets	:	0	0 pps
--------------------	---	---	-------

RL-dropped bytes	:	0	0 bps
------------------	---	---	-------

Total-dropped packets:		7983331	19578 pps
------------------------	--	---------	-----------

Total-dropped bytes	:	4243305874	83237464 bps
---------------------	---	------------	--------------

## Queue Buffer Usage:

Reserved buffer	:	32 pkts	6656 bytes
-----------------	---	---------	------------

Shared buffer	:	46 pkts	9568 bytes
---------------	---	---------	------------

## Queue: 5, Forwarding classes: 8q5

## Queued:

Packets	:	16762734	33285 pps
Bytes	:	8738359924	139389112 bps

## Transmitted:

Packets	:	8779416	13788 pps
Bytes	:	4495060992	56485136 bps

Tail-dropped packets : Not Available

RL-dropped packets	:	0	0 pps
--------------------	---	---	-------

RL-dropped bytes	:	0	0 bps
------------------	---	---	-------

Total-dropped packets:		7983318	19497 pps
------------------------	--	---------	-----------

Total-dropped bytes	:	4243298932	82903976 bps
---------------------	---	------------	--------------

## Queue Buffer Usage:

Reserved buffer	:	32 pkts	6656 bytes
-----------------	---	---------	------------

Shared buffer	:	52 pkts	10816 bytes
---------------	---	---------	-------------

## Queue: 6, Forwarding classes: 8q6

## Queued:

Packets	:	16762732	27688 pps
Bytes	:	8721917186	115931832 bps

## Transmitted:

Packets	:	9618678	12111 pps
Bytes	:	4924763136	49614432 bps

Tail-dropped packets : Not Available

RL-dropped packets	:	0	0 pps
--------------------	---	---	-------

RL-dropped bytes	:	0	0 bps
------------------	---	---	-------

Total-dropped packets:		7144054	15577 pps
------------------------	--	---------	-----------

Total-dropped bytes	:	3797154050	66317400 bps
---------------------	---	------------	--------------

## Queue Buffer Usage:

Reserved buffer	:	24 pkts	4992 bytes
-----------------	---	---------	------------



```

    Shared buffer      :              0 pkts              0 bytes
Queue: 7, Forwarding classes: 8q7
  Queued:
    Packets           :          16762733          26045 pps
    Bytes              :          8710804790        108947832 bps
  Transmitted:
    Packets           :          10187546          11805 pps
    Bytes              :          5216023552        48359208 bps
  Tail-dropped packets : Not Available
  RL-dropped packets  :              0              0 pps
  RL-dropped bytes    :              0              0 bps
  Total-dropped packets:          6575187          14240 pps
  Total-dropped bytes :          3494781238        60588624 bps
Queue Buffer Usage:
  Reserved buffer     :              21 pkts          4368 bytes
  Shared buffer       :              0 pkts              0 bytes

```

```

user@host# run show interfaces queue ge-0/0/4 remaining-traffic
Physical interface: ge-0/0/4, Enabled, Physical link is Up
  Interface index: 648, SNMP ifIndex: 1763
  Description:   UNI side - connected to ixia 6/9
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Queue: 0, Forwarding classes: 8q0
  Queued:
    Packets           :          77501          6106 pps
    Bytes              :          41609646        26235344 bps
  Transmitted:
    Packets           :           3206           243 pps
    Bytes              :          1641472        999248 bps
  Tail-dropped packets : Not Available
  RL-dropped packets  :              0              0 pps
  RL-dropped bytes    :              0              0 bps
  Total-dropped packets:          74295          5863 pps
  Total-dropped bytes :          39968174        25236096 bps
Queue Buffer Usage:
  Reserved buffer     :              8 pkts          1664 bytes
  Shared buffer       :          1495 pkts        310960 bytes
Queue: 1, Forwarding classes: 8q1
  Queued:
    Packets           :          77489          6100 pps

```



```

Bytes          :          41444318          26107008 bps
Transmitted:
Packets        :          9330             732 pps
Bytes          :          4776960          3002344 bps
Tail-dropped packets : Not Available
RL-dropped packets :           0             0 pps
RL-dropped bytes  :           0             0 bps
Total-dropped packets:          68159          5368 pps
Total-dropped bytes :          36667358        23104664 bps
Queue Buffer Usage:
Reserved buffer   :           8 pkts          1664 bytes
Shared buffer     :          1435 pkts        298480 bytes
Queue: 2, Forwarding classes: 8q2
Queued:
Packets          :          77485          6099 pps
Bytes            :          41362188        26054080 bps
Transmitted:
Packets          :          12417           975 pps
Bytes            :          6357504        3996992 bps
Tail-dropped packets : Not Available
RL-dropped packets :           0             0 pps
RL-dropped bytes  :           0             0 bps
Total-dropped packets:          65068          5124 pps
Total-dropped bytes :          35004684        22057088 bps
Queue Buffer Usage:
Reserved buffer   :           8 pkts          1664 bytes
Shared buffer     :          1507 pkts        313456 bytes
Queue: 3, Forwarding classes: 8q3
Queued:
Packets          :          77547          6114 pps
Bytes            :          41609314        26271632 bps
Transmitted:
Packets          :           3261           243 pps
Bytes            :          1646862          999248 bps
Tail-dropped packets : Not Available
RL-dropped packets :           0             0 pps
RL-dropped bytes  :           0             0 bps
Total-dropped packets:          74286          5871 pps
Total-dropped bytes :          39962452        25272384 bps
Queue Buffer Usage:
Reserved buffer   :           8 pkts          1664 bytes
Shared buffer     :          1381 pkts        287248 bytes
Queue: 4, Forwarding classes: 8q4

```



## Queued:

Packets	:	77502	6105 pps
Bytes	:	41450894	26131200 bps

## Transmitted:

Packets	:	9349	732 pps
Bytes	:	4786688	3002344 bps

Tail-dropped packets : Not Available

RL-dropped packets	:	0	0 pps
--------------------	---	---	-------

RL-dropped bytes	:	0	0 bps
------------------	---	---	-------

Total-dropped packets:		68153	5373 pps
------------------------	--	-------	----------

Total-dropped bytes	:	36664206	23128856 bps
---------------------	---	----------	--------------

## Queue Buffer Usage:

Reserved buffer	:	8 pkts	1664 bytes
-----------------	---	--------	------------

Shared buffer	:	1366 pkts	284128 bytes
---------------	---	-----------	--------------

## Queue: 5, Forwarding classes: 8q5

## Queued:

Packets	:	77480	6094 pps
Bytes	:	41358904	26032304 bps

## Transmitted:

Packets	:	12444	975 pps
Bytes	:	6371328	3996992 bps

Tail-dropped packets : Not Available

RL-dropped packets	:	0	0 pps
--------------------	---	---	-------

RL-dropped bytes	:	0	0 bps
------------------	---	---	-------

Total-dropped packets:		65036	5119 pps
------------------------	--	-------	----------

Total-dropped bytes	:	34987576	22035312 bps
---------------------	---	----------	--------------

## Queue Buffer Usage:

Reserved buffer	:	8 pkts	1664 bytes
-----------------	---	--------	------------

Shared buffer	:	1552 pkts	322816 bytes
---------------	---	-----------	--------------

## Queue: 6, Forwarding classes: 8q6

## Queued:

Packets	:	77970	6099 pps
Bytes	:	41151002	25749384 bps

## Transmitted:

Packets	:	30585	2440 pps
Bytes	:	15659520	9997088 bps

Tail-dropped packets : Not Available

RL-dropped packets	:	0	0 pps
--------------------	---	---	-------

RL-dropped bytes	:	0	0 bps
------------------	---	---	-------

Total-dropped packets:		47385	3659 pps
------------------------	--	-------	----------

Total-dropped bytes	:	25491482	15752296 bps
---------------------	---	----------	--------------

## Queue Buffer Usage:

Reserved buffer	:	3 pkts	624 bytes
-----------------	---	--------	-----------



```

    Shared buffer      :              0 pkts              0 bytes
Queue: 7, Forwarding classes: 8q7
  Queued:
    Packets           :              77971              6099 pps
    Bytes              :             41151540            25749384 bps
  Transmitted:
    Packets           :              30585              2440 pps
    Bytes              :             15659520            9997088 bps
  Tail-dropped packets : Not Available
  RL-dropped packets   :                  0                  0 pps
  RL-dropped bytes     :                  0                  0 bps
  Total-dropped packets:              47386              3659 pps
  Total-dropped bytes  :             25492020            15752296 bps
Queue Buffer Usage:
  Reserved buffer      :                  3 pkts             624 bytes
  Shared buffer        :                  0 pkts              0 bytes

```

- **show class-of-service packet-buffer usage**—Shows the total buffer usage of the system. The following is a sample output of the **show class-of-service packet-buffer usage** CLI command:

```

user@host# run show class-of-service packet-buffer usage
  Egress:
    Total Buffer Bytes   : 10652.89 KB in use out of 12480.00 KB
    Total Buffer Pkts    : 52445      in use out of 61440
    Dedicated Buffer Bytes : 48.14 KB  in use out of 738.16 KB
    Dedicated Buffer Pkts : 237        in use out of 3634
    Shared Buffer Bytes   : 10604.75 KB in use out of 11741.84 KB
    Shared Buffer Pkts    : 52208      in use out of 57806

```

- For interface sets, you can run the following commands:
  - **show interfaces interface-set voq**—Show VOQ statistics for the interface-set.
  - **show interfaces interface-set detail**—Display detailed output for the interface set.
  - **show interfaces interface-set voq remaining-traffic**—Show remaining traffic queue statistics for the interface set.

You can use the syslog to view the log messages and error reports.



## Storm Control on ACX Series Routers Overview

A traffic storm is generated when messages are broadcast on a network and each message prompts a receiving node to respond by broadcasting its own messages on the network. This, in turn, prompts further responses, creating a snowball effect. The LAN is suddenly flooded with packets, creating unnecessary traffic that leads to poor network performance or even a complete loss of network service. Storm control enables the router to monitor traffic levels and to drop broadcast, unknown unicast, and multicast (BUM) packets if they exceed the configured limit.

Storm control is applied on the following traffic types:

- Layer 2 broadcast packets
- Layer 2 multicast packets
- Layer 2 unregistered multicast packets
- Layer 2 registered multicast packets
- Layer 2 unknown unicast packets

Storm control functions slightly differently on ACX Series routers compared to other Juniper Networks routers. On ACX Series routers, storm control is only applicable at the physical interface level. No event will be logged when a traffic storm hits an ACX Series router. Also interfaces will not be bound to any default profile. The default action is to drop the packets exceeding the configured bandwidth.

Storm control configuration is done in two steps. The first step is to create a storm control profile. Use the following configuration to create your storm control profile on an ACX Series router:

```
storm-control-profiles {
  foo {
    all {
      bandwidth [percentage | level] <x>;
      [no-unknown-unicast | no-broadcast | no-multicast | no-registered-multicast | no-
unregistered-multicast]
    }
  }
}
```

The second step in configuring storm control is to bind the profile to physical interface. The following configuration shows how to bind your storm control profile:

```
[edit interfaces]
  ge-0/0/0 {
```



```
ether-options {  
    ethernet-switch-profile {  
        storm-control foo;  
    }  
}  
}
```

## RELATED DOCUMENTATION

*Controlling Network Access Using Traffic Policing Overview*

[ACX Series Universal Metro Router Overview](#)

[ACX1000 and ACX1100 Routers Hardware and CLI Terminology Mapping](#)

*ACX2000 and ACX2100 Routers Hardware and CLI Terminology Mapping*



# Configuring Class of Service on MX Series 5G Universal Routing Platforms

## IN THIS CHAPTER

- [Junos CoS on MX Series 5G Universal Routing Platforms Overview | 753](#)
- [CoS Features and Limitations on MX Series Routers | 754](#)
- [Configuring and Applying IEEE 802.1ad Classifiers | 758](#)
- [Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | 760](#)
- [Example: Performing Output Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | 762](#)
- [CoS-Based Interface Counters for IPv4 or IPv6 Aggregate on Layer 2 | 781](#)
- [Enabling a Timestamp for Ingress and Egress Queue Packets | 783](#)

## Junos CoS on MX Series 5G Universal Routing Platforms Overview

The increased demand for sophisticated, media-rich services, the exponential growth of mobile sessions, and the emerging trend of cloud computing require a networking infrastructure that supports massive numbers of subscribers, service types and instances, and bandwidth. A number of features and methods have been developed to address these advanced network requirements, including Junos *class of service* (CoS). Junos CoS is a set of mechanisms that helps maintain specified service levels for your network by optimizing and prioritizing network traffic so that demand for resources can meet requirements. Use CoS mechanisms to control the allocation of network attributes such as available bandwidth, latency, *jitter*, packet drop, and bit rate errors so that resources are managed to levels acceptable to your network customers and applications.

### CoS on Juniper Networks MX Series 5G Universal Routing Platforms

MX Series routers are available in a variety of configurations with robust features, including options that provide the level and granularity of the CoS support needed in your network. The MX Series hardware options include several models of Modular Port Concentrators (MPCs), using several different Modular Interface Cards (MICs), and several models of Dense Port Concentrators (DPCs). The MPCs and DPCs provide varying degrees of CoS support.



The MPCs are next-generation line modules for advanced Ethernet services edge and broadband edge networks using high capacity, modular Gigabit Ethernet, 10-Gigabit Ethernet, and 100-Gigabit Ethernet hardware. The MPCs house Packet Forwarding Engines that deliver comprehensive Layer 3 routing (IPv4 and IPv6), Layer 2 switching, inline services, and advanced hierarchical class of service (H-CoS) per MX Series slot. The MPCs can also take advantage of the high performance Junos Trio chipset.

Key CoS features provided by the MPCs include extensive queue management, scheduler hierarchy, shaping, intelligent oversubscription, weighted round robin (WRR), random early detection (RED), and weighted random early detection (WRED).

The DPCs (DPCE-X, DPCE-R, and DPCE-Q) each provide multiple physical interfaces and Packet Forwarding Engines on a single board that performs packet processing and forwarding. Each Packet Forwarding Engine consists of one I-chip for Layer 3 processing and one network processor for Layer 2. DPCE-Qs offer enhanced queuing capabilities and the CoS features of WRR, RED, and WRED.

## RELATED DOCUMENTATION

[CoS Features and Limitations on MX Series Routers](#) | 754

## CoS Features and Limitations on MX Series Routers

All Layer 3 Junos OS CoS functions are supported on the MX Series routers. In addition, Layer 3 CoS capabilities, with the exception of traffic shaping, are supported on virtual LANs (VLANs) that span multiple ports.

MX Series routers can be equipped with Flexible PIC Concentrators (FPCs) and associated Physical Interface Cards (PICs), Dense Port Concentrators (DPCs), Modular Interface Cards (MICs), Modular Port Concentrators (MPCs), or MPCs with associated MICs. In all cases, the command-line interface (CLI) configuration syntax refers to FPCs, PICs, and ports (*type- fpc/ pic/ port*).



**NOTE:** The MX80 router is a single-board router with a built-in Routing Engine and one Packet Forwarding Engine, which can have up to four MICs attached to it. The Packet Forwarding Engine has two “pseudo” Flexible PIC Concentrators (FPC 0 and FPC1). Because there is no switching fabric, the single Packet Forwarding Engine takes care of both ingress and egress packet forwarding.

Fixed classification places all packets in the same forwarding class, or the usual multifield or behavior aggregate (BA) classifications can be used to treat packets differently. BA classification with firewall filters can be used for classification based on IP precedence, DSCP, IEEE, or other bits in the frame or packet header.



However, the MX Series routers can also employ multiple BA classifiers on the same *logical interface*. The logical interfaces do not have to employ the same type of BA classifier. For example, a single logical interface can use classifiers based on IP precedence as well as IEEE 802.1p. If the CoS bits of interest are on the inner VLAN tag of a dual-tagged VLAN interface, the classifier can examine either the inner or outer bits. (By default, the classification is done based on the outer VLAN tag.)

Internal fabric scheduling is based on only two queues: high and low priority. Strict-high priority queuing is also supported in the high-priority category.

Egress port scheduling supports up to eight queues per port using a form of round-robin queue servicing. The supported priority levels are strict-high, high, medium-high, medium-low, and low. The MX Series router architecture supports both early discard and tail drop on the queues.

All CoS features are supported at line rate.

The MX960 router allows fixed classification of traffic. All packets on a logical interface can be put into the same forwarding class. For example:

```
[edit class-of-service interfaces ge-1/0/0 unit 0]
user@host#set forwarding-class af
```

As on other routers, the MX Series routers allow BA classification, the classifying of packets into different forwarding classes (up to eight) based on a value in the packet header. However, MX Series routers allow a mixture of BA classifiers (IEEE 802.1p and others) for logical interfaces on the same port. In the following example, the IEEE classifier is applied to Layer 2 traffic and the Internet precedence classifier is applied to Layer 3 (IP) traffic.

```
[edit class-of-service interfaces ge-0/0/0 unit 0]
user@host#set classifiers ieee-802.1 DOT1P-BA-1
user@host#set classifiers inet-precedence IPPRCE-BA-1
```

The IEEE classifier can also perform BA classification based on the bits of either the inner, outer, or transparent VLAN tag on a dual-tagged logical interface, as shown in the following example:

```
[edit class-of-service interfaces ge-0/0/0]
user@host#set unit 0 classifiers ieee-802.1 DOT1-BA-1 vlan-tag inner
user@host#set unit 1 classifiers ieee-802.1 DOT1-BA-1 vlan-tag outer
user@host#set unit 2 classifiers ieee-802.1 DOT1-BA-1 vlan-tag transparent
```





**NOTE:** The example above does not apply to single-tagged packets. The following example shows how to configure the classifier on single-tagged interfaces:

```
[edit class-of-service interfaces ge-0/0/0]
user@host# set unit 0 classifiers ieee-802.1 DOT1-BA-1
```

The default action is based on the outer VLAN tag's IEEE precedence bits.

As on other routers, the BA classification can be overridden with a multifield classifier in the action part of a firewall filter.

Rewrites are handled as on other routers, but MX Series routers support classifications and rewrites for aggregated Ethernet (ae-) logical interfaces. MX Series routers also support the use of egress firewall filters for DSCP rewrites for IPv4 and IPv6 packets. For example:

```
[edit firewall family inet]
user@host# set term 1 from destination-address 198.51.100.100/32
user@host# set term 1 then dscp af21
user@host# set term 2 then accept
```

On MX Series routers, the 64 classifier limit is a theoretical upper limit. In practice, you can configure 63 classifiers. Three values are used internally by the default IP precedence, IPv6, and EXP classifiers. Two other classifiers are used for forwarding class and queue operations. This leaves 58 classifiers for configuration purposes. If you configure Differentiated Services code point (DSCP) rewrites for MPLS, the maximum number of classifiers you can configure is less than 58.

On MX Series routers, IEEE 802.1 classifier bit rewrites are determined by forwarding class and packet priority, not by queue number and packet priority as on other routers.

The following scaling and performance parameters apply to MX Series routers:

- 48\* classifiers of each type, when subscriber management is enabled
- 32 rewrite tables of each type, when subscriber management is enabled
- Eight queues per port
- 64 WRED profiles
- 100-ms queue buffering for interfaces 1 Gbps and above; 500 ms for all others
- Line-rate CoS features





**NOTE:** \*Starting with Junos OS Release 16.1R5, Junos OS Release 17.1R3, Junos OS Release 17.2R2, and Junos OS Release 17.3R2, you can configure up to 48 classifiers per family at the [edit class-of-service classifiers] hierarchy level when subscriber management is enabled. In earlier releases, you could only configure up to 32 classifiers per family.

For more information about MX Series router CoS capabilities, including software configuration, see ["Configuring Hierarchical Schedulers for CoS" on page 448](#) and ["Enhanced Queuing DPC CoS Properties" on page 1071](#).

For Juniper Networks MX Series 5G Universal Routing Platforms, the following restrictions apply:

- You can only use multifield classifiers (but *not* BA classifiers) for IPv4 DSCP bits for virtual private LAN service (VPLS).
- You cannot use BA classifiers for IPv4 DSCP bits for Layer 2 VPNs.
- You cannot use BA classifiers for IPv6 DSCP bits for VPLS.
- You cannot use BA classifiers for IPv6 DSCP bits for Layer 2 VPNs.

On MX Series routers, you can apply classifiers or *rewrite rules* to an integrated bridging and routing (IRB) interface at the [edit class-of-service interfaces irb unit *logical-unit-number*] level of the hierarchy. All types of classifiers and rewrite rules are allowed. These classifiers and rewrite rules are independent of others configured on an MX Series router.

```
[edit class-of-service interfaces]
irb {
  unit logical-unit-number {
    classifiers {
      type (classifier-name | default) family (mpls | all);
    }
    rewrite-rules {
      dscp (rewrite-name | default);
      dscp-ipv6 (rewrite-name | default);
      exp (rewrite-name | default) protocol protocol-types;
      ieee-802.1 (rewrite-name | default) vlan-tag (outer | outer-and-inner);
      inet-precedence (rewrite-name | default);
    }
  }
}
```



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.

The IRB classifiers and rewrite rules are applied only to the “routed” packets. For logical interfaces that are part of a bridge domain, only IEEE classifiers and IEEE rewrite rules are allowed. Only the listed options are available for rewrite rules on an IRB.

For dual-tagged bridge domain logical interfaces, you can configure classification based on the inner or outer VLAN tag’s IEEE 802.1p bits using the `vlan-tag` statement with the `inner` or `outer` option:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
classifiers {
    ieee-802.1 (classifier-name | default) vlan-tag (inner | outer);
}
```

Also, for dual-tagged bridge domain logical interfaces, you can configure rewrite rules to rewrite the outer or both outer and inner VLAN tag’s IEEE 802.1p bits using the `vlan-tag` statement with the `outer` or `outer-and-inner` option:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
rewrite-rules {
    ieee-802.1 (rewrite-rule-name | default) vlan-tag (outer | outer-and-inner);
}
```

Change History Table

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

Release	Description
16.1R5	Starting with Junos OS Release 16.1R5, Junos OS Release 17.1R3, Junos OS Release 17.2R2, and Junos OS Release 17.3R2, you can configure up to 48 classifiers per family at the [edit class-of-service classifiers] hierarchy level when subscriber management is enabled.

Configuring and Applying IEEE 802.1ad Classifiers

If you apply an IEEE 802.1 classifier to a logical interface, this classifier takes precedence and is not compatible with any other classifier type. For Juniper Networks MX Series 5G Universal Routing Platform interfaces or IQ2 PICs with IEEE 802.1ad frame formats or EX Series switches, you can set the



forwarding class and loss priority for traffic on the basis of the three IEEE 802.1p bits (three bits in either the inner virtual LAN (VLAN) tag or the outer VLAN tag) and the drop eligible indicator (DEI) bit. You can apply the default map or customize one or more of the default values.

You then apply the classifier to the interface on which you configure IEEE 802.1ad frame formats.

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

**1. Define the custom IEEE 802.1ad map:**

- a. Create the classifier by specifying a name for it and defining it as an IEEE-802.1ad (DEI) classifier.

```
[edit]
user@host# edit class-of-service classifiers ieee-802.1ad dot1p_dei_class
```

- b. Assign the forwarding class and loss priority to the code-point alias.

```
[edit class-of-service classifiers ieee-802.1ad dot1p_dei_class]
user@host# set forwarding-class best-effort loss-priority low code-points [0000 1101]
```

**2. Apply the classifier to the logical interface:**

- a. Specify the interface to which you want to apply the classifier.

```
[edit]
user@host# edit class-of-service interfaces ge-2/0/0 unit 0
```

- b. Specify the name of the classifier you want to apply to the interface.

```
[edit class-of-service interfaces ge-2/0/0 unit 0]
user@host# set classifiers ieee-802.1ad dot1p_dei_class
```

**3. Verify the custom IEEE 802.1ad map configuration:**

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

```
class-of-service {
  classifiers {
```



```

        ieee-802.1ad dot1p_dei_class {
            forwarding-class best-effort {
                loss-priority low code-points [ 0000 1101 ];
            }
        }
    }
}

```

```

class-of-service {
    interfaces {
        ge-2/0/0 {
            unit 0 {
                classifiers {
                    ieee-802.1ad dot1p_dei_class;
                }
            }
        }
    }
}

```

## RELATED DOCUMENTATION

[Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 44](#)

[Applying Behavior Aggregate Classifiers to Logical Interfaces | 66](#)

## Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels

### IN THIS SECTION

- [Understanding Scheduling and Shaping of Traffic Routed to GRE Tunnels | 761](#)
- [Configuration Overview | 761](#)
- [Configuration Caveats | 761](#)



This topic covers the following information:

## Understanding Scheduling and Shaping of Traffic Routed to GRE Tunnels

On MX Series routers running Junos OS Release 12.3R4 or later revisions, 13.2R2 or later revision, or 13.3R1 or later, you can manage CoS scheduling and shaping of traffic routed to generic route encapsulation (GRE) tunnel interfaces configured on [MPC1 Q](#), [MPC2 Q](#), or [MPC2 EQ](#) modules.

A single egress logical interface can be converted to multiple GRE tunnel interfaces. A GRE tunnel physical interface can support many logical interfaces, but one or more of those logical interfaces might not have an output traffic control profiles attached. If a GRE tunnel logical interface is not attached to an output traffic control profile, the router does not assign the interface a dedicated scheduler. Instead, the interface uses a reserved scheduler intended for all *unshaped tunnel traffic* (traffic entering a GRE tunnel logical interface that does not have an explicit traffic control profile configuration).

## Configuration Overview

At GRE tunnel interfaces, the `output-traffic-control-profile` configuration statement can apply an output traffic scheduling and shaping profile at the physical or logical interface level, while the `output-traffic-control-profile-remaining` configuration statement can apply an output traffic scheduling and shaping profile for remaining traffic at the physical interface level only. Interface sets (sets of interfaces used to configure hierarchical CoS schedulers on supported Ethernet interfaces) are not supported on GRE tunnel interfaces.

By default—if you do not attach an output traffic control profile to the GRE tunnel physical interface—traffic entering the interface is scheduled and shaped using the default 95/5 scheduler with parameters as specified in the `tunnel-services` configuration.

If you use an output traffic control profile to configure the shaping rate at the GRE tunnel physical interface, the `shaping-rate` specified by the attached traffic control profile overrides the `bandwidth` specified as the tunnel services default value.

## Configuration Caveats

When configuring hierarchical CoS scheduling and shaping of traffic routed to GRE tunnels, keep the following guidelines in mind:

- You must first configure and commit a hierarchical scheduler on the GRE tunnel physical interface, specifying a maximum of two hierarchical scheduling levels for node scaling. After you commit the `hierarchical-scheduler` configuration, you can configure scheduling and queuing parameters at the GRE tunnel physical or logical interfaces.
- GRE tunnel interfaces support eight egress queues only. For interfaces on MPC1 Q, MPC2 Q, and MPC2 EQ modules, you can include the `max-queues-per-interface 4` statement at the `[edit fpc slot-`



`number pic pic-number]` hierarchy level to configure four-queue mode for the interface. However, any GRE tunnel interfaces configured on those ports have eight queues.

- Queuing and scheduling calculations include Layer 3 fields. For GRE interfaces, Layer 3 fields include the delivery header (the outer IP header), the 4-byte GRE header, and the payload protocol header and data.

## RELATED DOCUMENTATION

[Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | 760](#)

[Per-Unit Scheduling and Hierarchical Scheduling for MPC Interfaces | 1157](#)

[Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 1150](#)

## Example: Performing Output Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels

### IN THIS SECTION

- [Requirements | 762](#)
- [Overview | 763](#)
- [Configuration | 763](#)
- [Verification | 777](#)

This example shows how to configure a generic routing encapsulation (GRE) tunnel device to perform CoS output scheduling and shaping of IPv4 traffic routed to GRE tunnels. This feature is supported on MX Series routers running Junos OS Release 12.3R4 or later revisions, 13.2R2 or later revision, or 13.3R1 or later, with GRE tunnel interfaces configured on [MPC1 Q](#), [MPC2 Q](#), or [MPC2 EQ](#) modules.

### Requirements

This example uses the following Juniper Networks hardware and Junos OS software:

- Transport network—An IPv4 network running Junos OS Release 13.3.
- GRE tunnel device—One MX80 router installed as an ingress provider edge (PE) router.



- Input and output logical interfaces configurable on two ports of the built-in 10-Gigabit Ethernet Modular Interface Card (MIC):
  - Input logical interface ge-1/1/0.0 for receiving traffic that is to be transported across the network.
  - Output logical interfaces ge-1/1/1.0, ge-1/1/1.1, and ge-1/1/1.2 to convert to GRE tunnel source interfaces gr-1/1/10.1, gr-1/1/10.2, and gr-1/1/10.3.

For information about interfaces hosted on modules in MX80 routers, see the following topics:

- [MX5, MX10, MX40, and MX80 Modular Interface Card Description](#)
- [MX5, MX10, MX40, and MX80 Port and Interface Numbering](#)

## Overview

In this example, you configure the router with input and output logical interfaces for IPv4 traffic, and then you convert the output logical interface to four GRE tunnel source interfaces. You also install static routes in the routing table so that input traffic is routed to the four GRE tunnels.



**NOTE:** Before you apply a traffic control profile with a scheduler-map and shaping rate to a GRE tunnel interface, you must configure and commit a hierarchical scheduler on the GRE tunnel physical interface, specifying a maximum of two hierarchical scheduling levels for node scaling.

## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 764](#)
- [Configuring Interfaces, Hierarchical Scheduling on the GRE Tunnel Physical Interface, and Static Routes | 766](#)
- [Measuring GRE Tunnel Transmission Rates Without Shaping Applied | 770](#)
- [Configuring Output Scheduling and Shaping at GRE Tunnel Physical and Logical Interfaces | 771](#)

To configure scheduling and shaping in hierarchical CoS queues for traffic routed to GRE tunnel interfaces configured on MPC1Q, MPC2Q, or MPC2 EQ modules on an MX Series router, perform these tasks:



## CLI Quick Configuration

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

### Configuring Interfaces, Hierarchical Scheduling on the GRE Tunnel Physical Interface, and Static Routes

```
set chassis fpc 1 pic 1 tunnel-services bandwidth 1g
set interfaces ge-1/1/0 unit 0 family inet address 10.6.6.1/24
set interfaces ge-1/1/1 unit 0 family inet address 10.70.1.1/24 arp 10.70.1.3 mac
00:00:03:00:04:00
set interfaces ge-1/1/1 unit 0 family inet address 10.80.1.1/24 arp 10.80.1.3 mac
00:00:03:00:04:01
set interfaces ge-1/1/1 unit 0 family inet address 10.90.1.1/24 arp 10.90.1.3 mac
00:00:03:00:04:02
set interfaces ge-1/1/1 unit 0 family inet address 10.100.1.1/24 arp 10.100.1.3 mac
00:00:03:00:04:04
set interfaces gr-1/1/10 unit 1 family inet address 10.100.1.1/24
set interfaces gr-1/1/10 unit 1 tunnel source 10.70.1.1 destination 10.70.1.3
set interfaces gr-1/1/10 unit 2 family inet address 10.200.1.1/24
set interfaces gr-1/1/10 unit 2 tunnel source 10.80.1.1 destination 10.80.1.3
set interfaces gr-1/1/10 unit 3 family inet address 10.201.1.1/24
set interfaces gr-1/1/10 unit 3 tunnel source 10.90.1.1 destination 10.90.1.3
set interfaces gr-1/1/10 unit 4 family inet address 10.202.1.1/24
set interfaces gr-1/1/10 unit 4 tunnel source 10.100.1.1 destination 10.100.1.3
set interfaces gr-1/1/10 hierarchical-scheduler
set routing-options static route 10.2.2.0/24 next-hop gr-1/1/10.1
set routing-options static route 10.3.3.0/24 next-hop gr-1/1/10.2
set routing-options static route 10.4.4.0/24 next-hop gr-1/1/10.3
set routing-options static route 10.5.5.0/24 next-hop gr-1/1/10.4
```

### Configuring Output Scheduling and Shaping at GRE Tunnel Physical and Logical Interfaces

```
set class-of-service forwarding-classes queue 0 be
set class-of-service forwarding-classes queue 1 ef
set class-of-service forwarding-classes queue 2 af
set class-of-service forwarding-classes queue 3 nc
set class-of-service forwarding-classes queue 4 be1
set class-of-service forwarding-classes queue 5 ef1
set class-of-service forwarding-classes queue 6 af1
set class-of-service forwarding-classes queue 7 nc1
```



```

set class-of-service classifiers inet-precedence gr-inet forwarding-class be loss-priority low
code-points 000
set class-of-service classifiers inet-precedence gr-inet forwarding-class ef loss-priority low
code-points 001
set class-of-service classifiers inet-precedence gr-inet forwarding-class af loss-priority low
code-points 010
set class-of-service classifiers inet-precedence gr-inet forwarding-class nc loss-priority low
code-points 011
set class-of-service classifiers inet-precedence gr-inet forwarding-class be1 loss-priority low
code-points 100
set class-of-service classifiers inet-precedence gr-inet forwarding-class ef1 loss-priority low
code-points 101
set class-of-service classifiers inet-precedence gr-inet forwarding-class af1 loss-priority low
code-points 110
set class-of-service classifiers inet-precedence gr-inet forwarding-class nc1 loss-priority low
code-points 111
set class-of-service interfaces ge-1/1/0 unit 0 classifiers inet-precedence gr-inet
set class-of-service schedulers be_sch transmit-rate percent 30
set class-of-service schedulers ef_sch transmit-rate percent 40
set class-of-service schedulers af_sch transmit-rate percent 25
set class-of-service schedulers nc_sch transmit-rate percent 5
set class-of-service schedulers be1_sch transmit-rate percent 60
set class-of-service schedulers be1_sch priority low
set class-of-service schedulers ef1_sch transmit-rate percent 40
set class-of-service schedulers ef1_sch priority medium-low
set class-of-service schedulers af1_sch transmit-rate percent 10
set class-of-service schedulers af1_sch priority strict-high
set class-of-service schedulers nc1_sch shaping-rate percent 10
set class-of-service schedulers nc1_sch priority high
set class-of-service scheduler-maps sch_map_1 forwarding-class be scheduler be_sch
set class-of-service scheduler-maps sch_map_1 forwarding-class ef scheduler ef_sch
set class-of-service scheduler-maps sch_map_1 forwarding-class af scheduler af_sch
set class-of-service scheduler-maps sch_map_1 forwarding-class nc scheduler nc_sch
set class-of-service scheduler-maps sch_map_2 forwarding-class be scheduler be1_sch
set class-of-service scheduler-maps sch_map_2 forwarding-class ef scheduler ef1_sch
set class-of-service scheduler-maps sch_map_3 forwarding-class af scheduler af_sch
set class-of-service scheduler-maps sch_map_3 forwarding-class nc scheduler nc_sch
set class-of-service traffic-control-profiles gr-ifl-tcp3 guaranteed-rate 5m
set class-of-service traffic-control-profiles gr-ifd-tcp shaping-rate 10m
set class-of-service traffic-control-profiles gr-ifd-tcp-remain shaping-rate 7m
set class-of-service traffic-control-profiles gr-ifd-tcp-remain guaranteed-rate 4m
set class-of-service traffic-control-profiles gr-ifl-tcp1 scheduler-map sch_map_1
set class-of-service traffic-control-profiles gr-ifl-tcp1 shaping-rate 8m

```



```

set class-of-service traffic-control-profiles gr-ifl-tcp1 guaranteed-rate 3m
set class-of-service traffic-control-profiles gr-ifl-tcp2 scheduler-map sch_map_2
set class-of-service traffic-control-profiles gr-ifl-tcp2 guaranteed-rate 2m
set class-of-service traffic-control-profiles gr-ifl-tcp3 scheduler-map sch_map_3
set class-of-service interfaces gr-1/1/10 output-traffic-control-profile gr-ifd-tcp
set class-of-service interfaces gr-1/1/10 output-traffic-control-profile-remaining gr-ifd-remain
set class-of-service interfaces gr-1/1/10 unit 1 output-traffic-control-profile gr-ifl-tcp1
set class-of-service interfaces gr-1/1/10 unit 2 output-traffic-control-profile gr-ifl-tcp2
set class-of-service interfaces gr-1/1/10 unit 3 output-traffic-control-profile gr-ifl-tcp3

```

## Configuring Interfaces, Hierarchical Scheduling on the GRE Tunnel Physical Interface, and Static Routes

### Step-by-Step Procedure

To configure GRE tunnel interfaces (including enabling hierarchical scheduling) and static routes:

1. Configure the amount of bandwidth for tunnel services on the physical interface.

```

[edit]
user@host# set chassis fpc 1 pic 1 tunnel-services bandwidth 1g

```

2. Configure the GRE tunnel device output logical interface.

```

[edit]
user@host# set interfaces ge-1/1/0 unit 0 family inet address 10.6.6.1/24

```

3. Configure the GRE tunnel device output logical interface.

```

[edit]
user@host# set interfaces ge-1/1/1 unit 0 family inet address 10.70.1.1/24 arp 10.70.1.3 mac
00:00:03:00:04:00
user@host# set interfaces ge-1/1/1 unit 0 family inet address 10.80.1.1/24 arp 10.80.1.3 mac
00:00:03:00:04:01
user@host# set interfaces ge-1/1/1 unit 0 family inet address 10.90.1.1/24 arp 10.90.1.3 mac
00:00:03:00:04:02
user@host# set interfaces ge-1/1/1 unit 0 family inet address 10.100.1.1/24 arp 10.100.1.3
mac 00:00:03:00:04:04

```



4. Convert the output logical interface to four GRE tunnel interfaces.

```
[edit]
user@host# set interfaces gr-1/1/10 unit 1 family inet address 10.100.1.1/24
user@host# set interfaces gr-1/1/10 unit 1 tunnel source 10.70.1.1 destination 10.70.1.3
user@host# set interfaces gr-1/1/10 unit 2 family inet address 10.200.1.1/24
user@host# set interfaces gr-1/1/10 unit 2 tunnel source 10.80.1.1 destination 10.80.1.3
user@host# set interfaces gr-1/1/10 unit 3 family inet address 10.201.1.1/24
user@host# set interfaces gr-1/1/10 unit 3 tunnel source 10.90.1.1 destination 10.90.1.3
user@host# set interfaces gr-1/1/10 unit 4 family inet address 10.202.1.1/24
user@host# set interfaces gr-1/1/10 unit 4 tunnel source 10.100.1.1 destination 10.100.1.3
```

5. Enable the GRE tunnel interfaces to use hierarchical scheduling.

```
[edit]
user@host# set interfaces gr-1/1/10 hierarchical-scheduler
```

6. Install static routes in the routing table so that the device routes IPv4 traffic to the GRE tunnel source interfaces.

Traffic destined to the subnets 10.2.2.0/24, 10.3.3.0/24, 10.4.4.0/24, and 10.5.5.0/24 is routed to the tunnel interfaces at IP addresses 10.70.1.1, 10.80.1.1, 10.90.1.1, and 10.100.1.1, respectively.

```
[edit]
user@host# set routing-options static route 10.2.2.0/24 next-hop gr-1/1/10.1
user@host# set routing-options static route 10.3.3.0/24 next-hop gr-1/1/10.2
user@host# set routing-options static route 10.4.4.0/24 next-hop gr-1/1/10.3
user@host# set routing-options static route 10.5.5.0/24 next-hop gr-1/1/10.4
```

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

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

## Results

From configuration mode, confirm your configuration by entering the `show chassis fpc 1 pic 1`, `show interfaces ge-1/1/0`, `show interfaces ge-1/1/1`, `show interfaces gr-1/1/10`, and `show routing-options`



commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Confirm the configuration of interfaces, hierarchical scheduling on the GRE tunnel physical interface, and static routes.

```

user@host# show chassis fpc 1 pic 1
tunnel-services {
    bandwidth 1g;
}

user@host# show interfaces ge-1/1/0
unit 0 {
    family inet {
        address 10.6.6.1/24;
    }
}

user@host# show interfaces ge-1/1/1
unit 0 {
    family inet {
        address 10.70.1.1/24 {
            arp 10.70.1.3 mac 00:00:03:00:04:00;
        }
        address 10.80.1.1/24 {
            arp 10.80.1.3 mac 00:00:03:00:04:01;
        }
        address 10.90.1.1/24 {
            arp 10.90.1.3 mac 00:00:03:00:04:02;
        }
        address 10.100.1.1/24 {
            arp 10.100.1.3 mac 00:00:03:00:04:04;
        }
    }
}

user@host# show interfaces gr-1/1/10
hierarchical-scheduler;
unit 1 {
    tunnel {
        destination 10.70.1.3;
        source 10.70.1.1;
    }
}

```



```

    }
    family inet {
        address 10.100.1.1/24;
    }
}
unit 2 {
    tunnel {
        destination 10.80.1.3;
        source 10.80.1.1;
    }
    family inet {
        address 10.200.1.1/24;
    }
}
unit 3 {
    tunnel {
        destination 10.90.1.3;
        source 10.90.1.1;
    }
    family inet {
        address 10.201.1.1/24;
    }
}
unit 4 {
    tunnel {
        destination 10.100.1.3;
        source 10.100.1.1;
    }
    family inet {
        address 10.202.1.1/24;
    }
}

```

user@host# **show routing-options**

```

static {
    route 10.2.2.0/24 next-hop gr-1/1/10.1;
    route 10.3.3.0/24 next-hop gr-1/1/10.2;
    route 10.4.4.0/24 next-hop gr-1/1/10.3;
    route 10.5.5.0/24 next-hop gr-1/1/10.4;
}

```



## Measuring GRE Tunnel Transmission Rates Without Shaping Applied

### Step-by-Step Procedure

To establish a baseline measurement, note the transmission rates at each GRE tunnel source.

1. Pass traffic through the GRE tunnel at logical interfaces gr-1/1/10.1, gr-1/1/10.2, and gr-1/1/10.3.
2. To display the traffic rates at each GRE tunnel source, use the `show interfaces queue operational mode` command.

The following example command output shows detailed CoS queue statistics for logical interface gr-1/1/10.1 (the GRE tunnel from source IP address 10.70.1.1 to destination IP address 10.70.1.3).

```
user@host> show interfaces queue gr-1/1/10.1
Logical interface gr-1/1/10.1 (Index 331) (SNMP ifIndex 4045)
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Burst size: 0
Queue: 0, Forwarding classes: be
  Queued:
    Packets      :           31818312           102494 pps
    Bytes        :           6522753960        168091936 bps
  Transmitted:
    Packets      :           1515307           4879 pps
    Bytes        :           310637935          8001632 bps
    Tail-dropped packets :           21013826          68228 pps
    RED-dropped packets :           9289179          29387 pps
    Low          :           9289179          29387 pps
    Medium-low   :                0              0 pps
    Medium-high  :                0              0 pps
    High         :                0              0 pps
    RED-dropped bytes :          1904281695          48194816 bps
    Low          :          1904281695          48194816 bps
    Medium-low   :                0              0 bps
    Medium-high  :                0              0 bps
    High         :                0              0 bps
...

```



**NOTE:** This step shows command output for queue 0 (forwarding class be) only.



The command output shows that the GRE tunnel device transmits traffic from queue 0 at a rate of 4879 pps. Allowing for 182 bytes per Layer 3 packet, preceded by 24 bytes of GRE overhead (a 20-byte delivery header consisting of the IPv4 packet header followed by 4 bytes for GRE flags plus encapsulated protocol type), the traffic rate received at the tunnel destination device is 8,040,592 bps:

The command output shows that the GRE tunnel device transmits traffic from queue 0 at a rate of 4879 pps. Allowing for 182 bytes per Layer 3 packet, preceded by 24 bytes of GRE overhead (a 20-byte delivery header consisting of the IPv4 packet header followed by 4 bytes for GRE flags plus encapsulated protocol type), the traffic rate received at the tunnel destination device is 8,040,592 bps:

```
4879 packets/second X 206 bytes/packet X 8 bits/byte = 8,040,592 bits/second
```

## Configuring Output Scheduling and Shaping at GRE Tunnel Physical and Logical Interfaces

### Step-by-Step Procedure

To configure the GRE tunnel device with scheduling and shaping at GRE tunnel physical and logical interfaces:

1. Define eight transmission queues.

```
[edit]
user@host# set class-of-service forwarding-classes queue 0 be
user@host# set class-of-service forwarding-classes queue 1 ef
user@host# set class-of-service forwarding-classes queue 2 af
user@host# set class-of-service forwarding-classes queue 3 nc
user@host# set class-of-service forwarding-classes queue 4 be1
user@host# set class-of-service forwarding-classes queue 5 ef1
user@host# set class-of-service forwarding-classes queue 6 af1
user@host# set class-of-service forwarding-classes queue 7 nc1
```



**NOTE:** To configure up to eight forwarding classes with one-to-one mapping to output queues for interfaces on M120, M320, MX Series, and T Series routers and EX Series switches, use the `queue` statement at the `[edit class-of-service forwarding-classes]` hierarchy level.

If you need to configure up to 16 forwarding classes with multiple forwarding classes mapped to single queues for those interface types, use the `class` statement instead.



2. Configure BA classifier `gr-inet` that, based on IPv4 precedence bits set in an incoming packet, sets the forwarding class, loss-priority value, and DSCP bits of the packet.

```
[edit]
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class be loss-
priority low code-points 000
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class ef loss-
priority low code-points 001
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class af loss-
priority low code-points 010
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class nc loss-
priority low code-points 011
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class be1 loss-
priority low code-points 100
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class ef1 loss-
priority low code-points 101
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class af1 loss-
priority low code-points 110
user@host# set class-of-service classifiers inet-precedence gr-inet forwarding-class nc1 loss-
priority low code-points 111
```

3. Apply BA classifier `gr-inet` to the GRE tunnel device input at logical interface `ge-1/1/0.0`.

```
[edit]
user@host# set class-of-service interfaces ge-1/1/0 unit 0 classifiers inet-precedence gr-inet
```

4. Define a scheduler for each forwarding class.

```
[edit]
user@host# set class-of-service schedulers be_sch transmit-rate percent 30
user@host# set class-of-service schedulers ef_sch transmit-rate percent 40
user@host# set class-of-service schedulers af_sch transmit-rate percent 25
user@host# set class-of-service schedulers nc_sch transmit-rate percent 5
user@host# set class-of-service schedulers be1_sch transmit-rate percent 60
user@host# set class-of-service schedulers be1_sch priority low
user@host# set class-of-service schedulers ef1_sch transmit-rate percent 40
user@host# set class-of-service schedulers ef1_sch priority medium-low
user@host# set class-of-service schedulers af1_sch transmit-rate percent 10
user@host# set class-of-service schedulers af1_sch priority strict-high
```



```

user@host# set class-of-service schedulers nc1_sch shaping-rate percent 10
user@host# set class-of-service schedulers nc1_sch priority high

```

5. Define a scheduler map for each of three GRE tunnels.

```

[edit]
user@host# set class-of-service scheduler-maps sch_map_1 forwarding-class be scheduler be_sch
user@host# set class-of-service scheduler-maps sch_map_1 forwarding-class ef scheduler ef_sch
user@host# set class-of-service scheduler-maps sch_map_1 forwarding-class af scheduler af_sch
user@host# set class-of-service scheduler-maps sch_map_1 forwarding-class nc scheduler nc_sch
user@host# set class-of-service scheduler-maps sch_map_2 forwarding-class be scheduler be1_sch
user@host# set class-of-service scheduler-maps sch_map_2 forwarding-class ef scheduler ef1_sch
user@host# set class-of-service scheduler-maps sch_map_3 forwarding-class af scheduler af_sch
user@host# set class-of-service scheduler-maps sch_map_3 forwarding-class nc scheduler nc_sch

```

6. Define traffic control profiles for three GRE tunnel interfaces.

```

[edit]
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp1 scheduler-map sch_map_1
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp1 shaping-rate 8m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp1 guaranteed-rate 3m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp2 scheduler-map sch_map_2
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp2 guaranteed-rate 2m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp3 scheduler-map sch_map_3
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp3 guaranteed-rate 5m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp shaping-rate 10m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp-remain shaping-rate 7m
user@host# set class-of-service traffic-control-profiles gr-ifl-tcp-remain guaranteed-rate 4m

```

7. Apply CoS scheduling and shaping to the output traffic at the physical interface and logical interfaces.

```

[edit]
user@host# set class-of-service interfaces gr-1/1/10 output-traffic-control-profile gr-ifd-tcp
user@host# set class-of-service interfaces gr-1/1/10 output-traffic-control-profile-remaining
gr-ifd-remain
user@host# set class-of-service interfaces gr-1/1/10 unit 1 output-traffic-control-profile gr-
ifl-tcp1
user@host# set class-of-service interfaces gr-1/1/10 unit 2 output-traffic-control-profile gr-
ifl-tcp2

```



```
user@host# set class-of-service interfaces gr-1/1/10 unit 2 output-traffic-control-profile gr-
ifl-tcp3
```

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

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

## Results

From configuration mode, confirm your configuration by entering the `show class-of-service forwarding-classes`, `show class-of-service classifiers`, `show class-of-service interfaces ge-1/1/0`, `show class-of-service schedulers`, `show class-of-service scheduler-maps`, `show class-of-service traffic-control-profiles`, and `show class-of-service interfaces gr-1/1/10` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

Confirm the configuration of output scheduling and shaping at the GRE tunnel physical and logical interfaces.

```
user@host# show class-of-service forwarding-classes
queue 0 be;
queue 1 ef;
queue 2 af;
queue 3 nc;
queue 4 be1;
queue 5 ef1;
queue 6 af1;
queue 7 nc1;
```

```
user@host# show class-of-service classifiers
inet-precedence gr-inet {
    forwarding-class be {
        loss-priority low code-points 000;
    }
    forwarding-class ef {
        loss-priority low code-points 001;
    }
    forwarding-class af {
        loss-priority low code-points 010;
    }
    forwarding-class nc {
```



```

        loss-priority low code-points 011;
    }
    forwarding-class be1 {
        loss-priority low code-points 100;
    }
    forwarding-class ef1 {
        loss-priority low code-points 101;
    }
    forwarding-class af1 {
        loss-priority low code-points 110;
    }
    forwarding-class nc1 {
        loss-priority low code-points 111;
    }
}

```

user@host# **show class-of-service interfaces ge-1/1/0**

```

unit 0 {
    classifiers {
        inet-precedence gr-inet;
    }
}

```

user@host# **show class-of-service schedulers**

```

be_sch {
    transmit-rate percent 30;
}
ef_sch {
    transmit-rate percent 40;
}
af_sch {
    transmit-rate percent 25;
}
nc_sch {
    transmit-rate percent 5;
}
be1_sch {
    transmit-rate percent 60;
    priority low;
}
ef1_sch {
    transmit-rate percent 40;
    priority medium-low;
}

```



```

}
af1_sch {
    transmit-rate percent 10;
    priority strict-high;
}
nc1_sch {
    shaping-rate percent 10;
    priority high;
}

user@host# show class-of-service scheduler-maps
sch_map_1 {
    forwarding-class be scheduler be_sch;
    forwarding-class ef scheduler ef_sch;
    forwarding-class af scheduler af_sch;
    forwarding-class nc scheduler nc_sch;
}
sch_map_2 {
    forwarding-class be scheduler be1_sch;
    forwarding-class ef scheduler ef1_sch;
}
sch_map_3 {
    forwarding-class af scheduler af_sch;
    forwarding-class nc scheduler nc_sch;
}

user@host# show class-of-service traffic-control-profiles
gr-ifl-tcp1 {
    scheduler-map sch_map_1;
    shaping-rate 8m;
    guaranteed-rate 3m;
}
gr-ifl-tcp2 {
    scheduler-map sch_map_2;
    guaranteed-rate 2m;
}
gr-ifl-tcp3 {
    scheduler-map sch_map_3;
    guaranteed-rate 5m;
}
gr-ifd-remain {
    shaping-rate 7m;
    guaranteed-rate 4m;
}

```



```
}
gr-efd-tcp {
    shaping-rate 10m;
}

user@host# show class-of-service interfaces gr-1/1/10
gr-1/1/10 {
    output-traffic-control-profile gr-efd-tcp;
    output-traffic-control-profile-remaining gr-efd-remain;
    unit 1 {
        output-traffic-control-profile gr-efd-tcp1;
    }
    unit 2 {
        output-traffic-control-profile gr-efd-tcp2;
    }
    unit 3 {
        output-traffic-control-profile gr-efd-tcp3;
    }
}
```

## Verification

### IN THIS SECTION

- [Verifying That Scheduling and Shaping Are Attached to the GRE Tunnel Interfaces | 777](#)
- [Verifying That Scheduling and Shaping Are Functioning at the GRE Tunnel Interfaces | 779](#)

Confirm that the configurations are working properly.

### Verifying That Scheduling and Shaping Are Attached to the GRE Tunnel Interfaces

#### Purpose

Verify the association of traffic control profiles with GRE tunnel interfaces.



## Action

Verify the traffic control profile attached to the GRE tunnel physical interface by using the **show class-of-service interface gr-1/1/10 detail** operational mode command.

- ```

user@host> show class-of-service interface gr-1/1/10 detail
Physical interface: gr-1/1/10, Enabled, Physical link is Up
  Type: GRE, Link-level type: GRE, MTU: Unlimited, Speed: 1000mbps
  Device flags   : Present Running
  Interface flags: Point-To-Point SNMP-Traps

Physical interface: gr-1/1/10, Index: 220
Queues supported: 8, Queues in use: 8
  Output traffic control profile: gr-ifd-tcp, Index: 17721
  Output traffic control profile remaining: gr-ifd-remain, Index: 58414
  Congestion-notification: Disabled

Logical interface gr-1/1/10.1
  Flags: Point-To-Point SNMP-Traps 0x4000 IP-Header
10.70.1.3:10.70.1.1:47:df:64:0000000000000000 Encapsulation: GRE-NULL
  Gre keepalives configured: Off, Gre keepalives adjacency state: down
  inet 10.100.1.1/24
  Logical interface: gr-1/1/10.1, Index: 331


Object	Name	Type	Index
Traffic-control-profile	gr-ifl-tcp1	Output	17849
Classifier	ipprec-compatibility	ip	13



Logical interface gr-1/1/10.2
  Flags: Point-To-Point SNMP-Traps 0x4000 IP-Header
10.80.1.3:10.80.1.1:47:df:64:0000000000000000 Encapsulation: GRE-NULL
  Gre keepalives configured: Off, Gre keepalives adjacency state: down
  inet 10.200.1.1/24
  Logical interface: gr-1/1/10.2, Index: 332


Object	Name	Type	Index
Traffic-control-profile	gr-ifl-tcp2	Output	17856
Classifier	ipprec-compatibility	ip	13



Logical interface gr-1/1/10.3
  Flags: Point-To-Point SNMP-Traps 0x4000 IP-Header
10.90.1.3:10.90.1.1:47:df:64:0000000000000000 Encapsulation: GRE-NULL
  Gre keepalives configured: Off, Gre keepalives adjacency state: down
  inet 10.201.1.1/24

```



Logical interface: gr-1/1/10.3, Index: 333

| Object                  | Name                 | Type   | Index |
|-------------------------|----------------------|--------|-------|
| Traffic-control-profile | gr-ifl-tcp3          | Output | 17863 |
| Classifier              | ipprec-compatibility | ip     | 13    |

## Meaning

Ingress IPv4 traffic routed to GRE tunnels on the device is subject to CoS output scheduling and shaping.

## Verifying That Scheduling and Shaping Are Functioning at the GRE Tunnel Interfaces

### Purpose

Verify the traffic rate shaping at the GRE tunnel interfaces.

### Action

1. Pass traffic through the GRE tunnel at logical interfaces gr-1/1/10.1, gr-1/1/10.2, and gr-1/1/10.3.
2. To verify the rate shaping at each GRE tunnel source, use the `show interfaces queue operational mode` command.

The following example command output shows detailed CoS queue statistics for logical interface gr-1/1/10.1 (the GRE tunnel from source IP address 10.70.1.1 to destination IP address 10.70.1.3):

```
user@host> show interfaces queue gr-1/1/10.1
Logical interface gr-1/1/10.1 (Index 331) (SNMP ifIndex 4045)
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Burst size: 0
Queue: 0, Forwarding classes: be
  Queued:
    Packets      :          59613061          51294 pps
    Bytes        :       12220677505       84125792 bps
  Transmitted:
    Packets      :          2230632          3039 pps
    Bytes        :       457279560       4985440 bps
    Tail-dropped packets :          4471146          2202 pps
    RED-dropped packets :          52911283          46053 pps
    Low          :          49602496          46053 pps
```



```

Medium-low      :                0                0 pps
Medium-high     :                0                0 pps
High            :            3308787                0 pps
RED-dropped bytes :        10846813015        75528000 bps
Low             :        10168511680        75528000 bps
Medium-low      :                0                0 bps
Medium-high     :                0                0 bps
High            :        678301335                0 bps
Queue: 1, Forwarding classes: ef
Queued:
Packets         :        15344874        51295 pps
Bytes           :        3145699170      84125760 bps
Transmitted:
Packets         :        366115         1218 pps
Bytes           :        75053575      1997792 bps
Tail-dropped packets :        364489         1132 pps
RED-dropped packets :        14614270      48945 pps
Low             :        14614270      48945 pps
Medium-low      :                0                0 pps
Medium-high     :                0                0 pps
High            :                0                0 pps
RED-dropped bytes :        2995925350      80270528 bps
Low             :        2995925350      80270528 bps
Medium-low      :                0                0 bps
Medium-high     :                0                0 bps
High            :                0                0 bps
...

```



**NOTE:** This step shows command output for queue 0 (forwarding class be) and queue 1 (forwarding class ef) only.

## Meaning

Now that traffic shaping is attached to the GRE tunnel interfaces, the command output shows that traffic shaping specified for the tunnel at logical interface gr-1/1/10.1 (shaping-rate 8m and guaranteed-rate 3m) is honored.



- For queue 0, the GRE tunnel device transmits traffic at a rate of 3039 pps. The traffic rate received at the tunnel destination device is 5,008,272 bps:

$$3039 \text{ packets/second} \times 206 \text{ bytes/packet} \times 8 \text{ bits/byte} = 5,008,272 \text{ bits/second}$$

- For queue 0, the GRE tunnel device transmits traffic at a rate of 1218 pps. The traffic rate received at the tunnel destination device is 2,007,264 bps:

$$1218 \text{ packets/second} \times 206 \text{ bytes/packet} \times 8 \text{ bits/byte} = 2,007,264 \text{ bits/second}$$

Compare these statistics to the baseline measurements taken without traffic shaping, as described in ["Measuring GRE Tunnel Transmission Rates Without Shaping Applied" on page 770](#).

## RELATED DOCUMENTATION

[Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | 760](#)

[Configuring Traffic Control Profiles for Shared Scheduling and Shaping | 340](#)

## CoS-Based Interface Counters for IPv4 or IPv6 Aggregate on Layer 2

Beginning with Junos OS Release 14.1, Layer 2 CoS-based traffic metering is available for MX series routers with MPCs and MX80 routers. It can be used in greenfield deployments or as a replacement for term-matching counters configured via firewall filters. (Term-matching counters configured via firewall filters can have several drawbacks, including a one-filter-per-family limit, the inclusion of overhead bytes, and less operational efficiency than CoS-based counters). With CoS-based counters, a single aggregate counter per forwarding class can be used for inet and inet6 flows. Both bytes and packet total are counted. Note that flow rates are not measured, and forwarding-class accounting for host-bound traffic is not supported.

You can configure the counters with any or all of the following parameters:

- Logical | physical interfaces
- IPv4 | IPv6 traffic
- Unicast | multicast traffic
- Ingress | egress flows



CoS-based interface counters are highly accurate and can be configured to exclude overhead bytes (such as protocol encapsulations) so end-customer packets can be differentiated from other traffic. At ingress only packets forwarded to the fabric are counted, and at egress only packets forwarded to the WAN are counted. In other words, forwarding-class accounting applies to transit traffic only, not host-generated or host-bound traffic. Non-relevant network protocols such as ARP, BFD, and EOAM, as well as dropped packets, are not counted.

To support native interface counters, a new CLI option, `enhanced`, is introduced under `forwarding-class accounting` at both the physical and logical interface levels:

```

interfaces {
    interface-name{
        forwarding-class-accounting {
            enhanced {
                overhead-bytes overhead-value(;
                traffic-type (ucast | mcast);
                family (ipv4 | ipv6 | both );
                direction (ingress | egress | both);
            }
        }
    }
}

```

To view additional counter details, run the following `show` commands:

- `show interfaces forwarding-class-counters interface-name`
- `show class-of-service interface interface-name comprehensive`
- `show class-of-service interface interface-name detail`

The `comprehensive` option shows both forwarding-class accounting parameters and the forwarding-class counter, whereas `detail` shows only the forwarding-class accounting parameters. It does not display counters for each forwarding class.

**Change History Table**

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

| Release | Description                                                                                                                             |
|---------|-----------------------------------------------------------------------------------------------------------------------------------------|
| 14.1    | Beginning with Junos OS Release 14.1, Layer 2 CoS-based traffic metering is available for MX series routers with MPCs and MX80 routers. |



## RELATED DOCUMENTATION

*forwarding-class-accounting*

*enhanced*

*show class-of-service interface*

## Enabling a Timestamp for Ingress and Egress Queue Packets

Beginning with Junos OS Release 16.1, you can enable a packet timestamp feature to record the time at which the last packet is enqueued for CoS ingress and egress queues. Timestamps are enabled and reported per FPC. When the feature is enabled, the Packet Forwarding Engine begins collection the timestamp for all ingress and egress queue counters on the FPC. By default, packet timestamp information is not collected.

To activate packet timestamp collection for CoS ingress and egress queues:

- Enable the timestamp on the desired FPC.

```
[edit chassis fpc slot-number traffic-manager]
user@host# set packet-timestamp enable
```



**NOTE:** When you enable or disable the packet timestamp for an FPC that is already up, the FPC is automatically rebooted when you commit the changes. The action takes effect when the FPC is back up.



**NOTE:** For aggregated Ethernet interfaces, enable the packet timestamp on all FPCs that have an aggregated Ethernet leg. When you display the queue statistics for the interface, the timestamps for all the legs are shown.

The following commands display the collected timestamps in the Last-packet enqueued field:

- `show interfaces queue both-ingress-egress interface-name`
- `show interfaces queue interface-name`
- `show interfaces queue interface-name aggregate`

### Change History Table



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

| Release | Description                                                                                                                                                                |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 16.1    | Beginning with Junos OS Release 16.1, you can enable a packet timestamp feature to record the time at which the last packet is enqueued for CoS ingress and egress queues. |

**RELATED DOCUMENTATION**

| [CoS-Based Interface Counters for IPv4 or IPv6 Aggregate on Layer 2](#) | **781**



# Configuring Class of Service on PTX Series Packet Transport Routers

IN THIS CHAPTER

- CoS Features and Limitations on PTX Series Routers | 785
- CoS Feature Differences Between PTX Series Packet Transport Routers and T Series Routers | 790
- Understanding Scheduling on PTX Series Routers | 793
- Virtual Output Queues on PTX Series Packet Transport Routers | 797
- Example: Configuring Excess Rate for PTX Series Packet Transport Routers | 813
- Identifying the Source of RED Dropped Packets on PTX Series Routers | 821
- Configuring Queuing and Shaping on Logical Interfaces on PTX Series Routers | 829
- Example: Configuring Queuing and Shaping on Logical Interfaces in PTX Series Packet Transport Routers | 833
- Example: Configuring Strict-Priority Scheduling on a PTX Series Router | 842
- CoS Support on EVPN VXLANs | 853
- Understanding CoS CLI Configuration Statements on PTX Series Routers | 862
- Classification Based on Outer Header of Decapsulation Tunnel | 866

## CoS Features and Limitations on PTX Series Routers

Table 70 on page 786 summarizes CoS features and limitations on PTX Series Packet Transport Routers.

The following table lists the CoS features supported on the PTX Series router, as well as the limitations relevant to the PTX Series router. Note that this list is a subset of the overall CoS feature set.



Table 70: CoS Features and Limitations on PTX Series Routers

| CoS Feature                                                        | Cap<br>acit<br>y | Comments                                                                                                                                                                                                                                     |
|--------------------------------------------------------------------|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Classifiers</b>                                                 |                  |                                                                                                                                                                                                                                              |
| Maximum number per PFE                                             | 64               | <p>L2 classifiers (sum of IEEE-802.1p and IEEE-802.1ad cannot exceed 32)</p> <p>DSCP and IP precedence classifiers (sum of DSCP and IP precedence classifiers cannot exceed 32)</p> <p>DSCP IPv6 classifiers</p> <p>MPLS EXP classifiers</p> |
| dscp                                                               | Yes              | DSCP and IP precedence classifiers cannot be configured on the same logical interface.                                                                                                                                                       |
| dscp-ipv6                                                          | Yes              | Separate classifiers can be applied for IPv4 and IPv6 packets per logical interface.                                                                                                                                                         |
| ieee-802.1p and ieee-802.1ad                                       | Yes              | You can associate IEEE-802.1p and IEEE-802.1ad classifiers with any other type of classifier on the same logical interface. For L3 packets, an L3 classifier takes precedence over an IEEE classifier.                                       |
| inet-precedence                                                    | Yes              |                                                                                                                                                                                                                                              |
| mpls-exp                                                           | Yes              | <b>NOTE:</b> MPLS EXP classifiers are not supported on PTX10001-20C routers.                                                                                                                                                                 |
| Loss priorities based on the Frame Relay discard eligible (DE) bit | No               |                                                                                                                                                                                                                                              |
| <b>Drop Profiles</b>                                               |                  |                                                                                                                                                                                                                                              |
| Maximum number                                                     | 32               | You can configure up to 32 <i>drop-profiles</i> in the PTX chassis.                                                                                                                                                                          |



Table 70: CoS Features and Limitations on PTX Series Routers *(Continued)*

| CoS Feature                                        | Cap<br>acit<br>y | Comments                                                                                 |
|----------------------------------------------------|------------------|------------------------------------------------------------------------------------------|
| Per queue                                          | Yes              |                                                                                          |
| Per loss priority                                  | Yes              | <b>NOTE:</b> Packet loss priority <i>medium-low</i> is not supported on PTX1000 routers. |
| Per Transmission Co<br>ntrol Protocol (TCP)<br>bit | No               |                                                                                          |



**Table 70: CoS Features and Limitations on PTX Series Routers** *(Continued)*

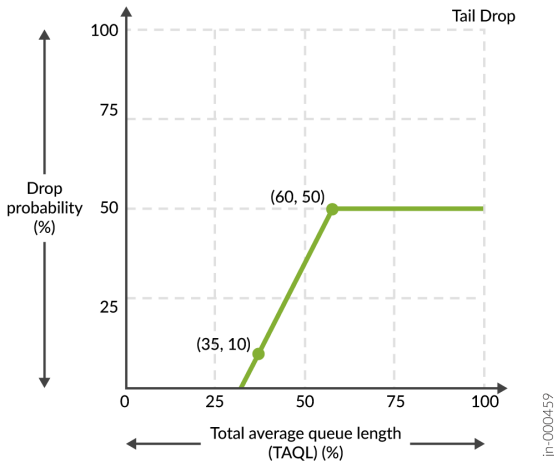
| CoS Feature | Cap<br>acit<br>y | Comments |
|-------------|------------------|----------|
|-------------|------------------|----------|

**NOTE:** PTX10K-LC1201 and PTX10K-LC1202 line cards on PTX10001-36MR, PTX10004, PTX10008, and PTX10016 routers support one or two drop profile points (fill level/drop probability pairs) per drop profile:

- Discrete and interpolated drop profiles are equivalent on these platforms; the hardware ignores the interpolated option. You cannot configure both discrete and interpolated pairs on a single drop profile.
- Point one drop probability must be less than or equal to 25%.
- You must configure fill level and drop probabilities in pairs and in ascending order.

Figure 51 on page 788 shows how two point RED curves generate a single line.

**Figure 51: Two Drop Profile Points**



Although the first point is (35,10), we see the drop profile points starting earlier where the line intersects the X axis.

At the second point (60,50), the drop probability is maximum and remains at point 50 from TAQL 60% through 100%, after which the tail drop happens.

---

**Policing**

---



Table 70: CoS Features and Limitations on PTX Series Routers *(Continued)*

| CoS Feature                     | Cap<br>acit<br>y | Comments                                                                                                                                                                                                                                                                                                                   |
|---------------------------------|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Traffic policing                | Yes              |                                                                                                                                                                                                                                                                                                                            |
| Two-rate tricolor marking (TCM) | Yes              | <b>NOTE:</b> Tri-color marking is not supported on PTX1000 routers.                                                                                                                                                                                                                                                        |
| <b>Queuing</b>                  |                  |                                                                                                                                                                                                                                                                                                                            |
| Priority                        | Yes<br>(4)       |                                                                                                                                                                                                                                                                                                                            |
| Per-queue output statistics     | Yes              | Red-dropped counters are not maintained per drop precedence. Also tail drop counters always show zero because packets are always dropped by the RED algorithm.                                                                                                                                                             |
| transmit-rate percent           | Yes              | Percentage transmit rate for a scheduler has the range 1 through 100 percent. The range is 0 through 100 percent for M, MX and T Series routers and EX Series switches; and 0 through 200 percent for the SONET/SDH OC48/STM16 IQE PIC.<br><br>On PTX Series routers, unconfigured interfaces are equivalent to percent 0. |
| <b>Rewrite Markers</b>          |                  |                                                                                                                                                                                                                                                                                                                            |
| Maximum number per PFE          | 64               | The sum of L2 and L3 rewrite rules cannot exceed 64.                                                                                                                                                                                                                                                                       |
| dscp                            | Yes              |                                                                                                                                                                                                                                                                                                                            |
| dscp-ipv6                       | Yes              |                                                                                                                                                                                                                                                                                                                            |
| ieee-802.1p and ieee-802.1ad    | Yes              | L2 and L3 rewrites can be applied to the same packet simultaneously.                                                                                                                                                                                                                                                       |



Table 70: CoS Features and Limitations on PTX Series Routers *(Continued)*

| CoS Feature     | Cap<br>acit<br>y | Comments |
|-----------------|------------------|----------|
| inet-precedence | No               |          |
| mpls-exp        | Yes              |          |



**NOTE:** Junos OS Evolved does not support applying classifier and rewrite rules to IRB interfaces on PTX Series routers.

## CoS Feature Differences Between PTX Series Packet Transport Routers and T Series Routers

This topic provides a list of class-of-service features available on PTX Series routers and compares them with class-of-service features on T Series routers.

### Classifiers

- T Series routers support VRF table labels for Layer 3 VPNs. On PTX Series routers, this feature is not supported, unless the router is running Junos OS Evolved. PTX Series routers running Junos OS Evolved support Layer 3 VPNs.
- On T Series routers, IEEE 802.1 classifiers cannot coexist with Layer 3 classifiers. On PTX Series routers, these classifiers can coexist.
- On T Series routers, IEEE classifiers are supported on Ethernet IQ, IQ2, and IQ2-E interfaces. These interfaces have the flexibility of classifying traffic based on inner or outer VLAN tags. On PTX Series routers, IEEE classification is always based on outer VLAN tags.

### Rewrite

- PTX Series routers do not support rewrite of both exp and inet-precedence fields using:
  - exp protocol mpls-any
  - exp protocol mpls-inet-both



- exp protocol mpls-inet-both-non-vpn
- On T Series routers, DSCP rewrite and DSCP IPv6 rewrite are not supported for the MPLS protocol. PTX Series routers support rewrite of both DSCP and DSCP IPv6 for protocol MPLS.
- PTX Series routers support Layer 2 rewrite of 802.1p and 802.1ad, to either the outer VLAN tag, or both outer and inner VLAN tags.

### Forwarding Class

- On T Series routers, you can override the default fabric priority queuing of egress traffic by including the priority statement at the following hierarchy level:

```
[class-of-service forwarding-classes queue queue-number class-name]
priority (high | low);
```

On PTX Series routers, fabric priority queuing is not supported; therefore, the priority statement for forwarding-classes is not supported.

### Tricolor Marking

- On T Series routers, the copy-plp-all statement must be configured to support tricolor marking. On PTX Series routers, tricolor marking is enabled by default.

### Schedulers

- T Series routers, which use egress queuing architecture, support chassis and fabric schedulers. Alternatively, PTX Series routers support a virtual output queuing (VOQ) architecture and the fabric schedulers utilize the CoS scheduling parameters to configure the fabric schedulers. There is not a separate configuration for the fabric schedulers on PTX Series routers. With the VOQ architecture, packets are queued and dropped on ingress during congestion.
- On T Series routers, high-priority queues have precedence to acquire excess bandwidth and might consume all excess bandwidth. On PTX Series routers, excess bandwidth is shared based on the ratio of the configured transfer rate. Therefore, all priority queues get a share of excess bandwidth.
- On T Series routers, strict-high-priority queues and high-priority queues are assigned the same hardware priority. On PTX Series routers, strict-high-priority queues and high-priority queues are assigned different hardware priorities. Strict-high-priority can starve other queues if a rate limiter is not applied on PTX Series routers.
- On T Series routers, if a strict-high-priority queue is oversubscribed, it can block all other queues except high-priority queues. On PTX Series routers, if a strict-high-priority queue is oversubscribed, it can block all other queues including high-priority queues.



To restrict the bandwidth of strict-high priority queues on PTX Series routers, use the `transmit-rate rate-limit` configuration statement.

- On both T Series routers and PTX Series routers, if a strict-high-priority queue is oversubscribed and results in oversubscription of the guaranteed bandwidth, the distribution of bandwidth that is not taken up by strict-high-priority queues is undetermined. T Series routers and PTX Series routers distribute this unused bandwidth differently.

### Buffer Size and Latency

- On T Series routers, memory allocation dynamic (MAD) is enabled by default and can be disabled. A comparable feature on PTX Series routers is `buffer-size shared mode of schedulers`.
- On T Series routers, the maximum delay bandwidth buffering configured per queue is 50 ms. On PTX Series routers, `buffer-size percent 100` requests the platform's full buffer latency at the port or shaping rate, that is, a nominal queue limit of  $(\text{productFullLatency} \times \text{portRate})$  bytes. For all PTX routers, the `productFullLatency` is 100 ms except for the PTX10001, and line cards LC1201 and LC1202, which have a 25 ms buffer.
- On T Series routers, the maximum latency associated with a packet is fairly consistent and independent of the number of sources sending the traffic to an interface. On PTX Series routers, over-provisioning is possible. When traffic is sent from multiple Packet Forwarding Engines (PFEs), the latency can be higher than when traffic is sent from one PFE due to the PTX dynamically adjusting buffers from multiple PFEs for a Virtual Output Queue (VOQ).
- On T Series routers, a high-priority queue has lower latency than a low priority queue with the same configured transfer rate and same offered load. On PTX Series routers, there is no latency difference.

### Drop Profile

- PTX Series PFEs not support drop-profile assignments for a queue based on the protocol. As a consequence, the `protocol` option for the `drop-profile-map` configuration statement is treated as `protocol any`.

### Interface Queue Statistics

- On T Series routers, transmitted byte counters are computed using Layer 3 packet length. On PTX Series routers, transmitted byte counters are computed using the full Layer 2 overhead (including all L2 encapsulation and CRC) plus 12 for the inter-packet gap plus 8 for the preamble.
- On Junos OS Evolved PTX Series routers, the tail-dropped counters and the RED-dropped counters are displayed separately in the output of the `show interfaces queue` command. On Junos OS PTX Series routers, tail-dropped counters are always zero. All the packet drops are shown as RED-dropped in the `show interfaces voq` output.



## RELATED DOCUMENTATION

[Understanding CoS CLI Configuration Statements on PTX Series Routers | 862](#)

## Understanding Scheduling on PTX Series Routers

### IN THIS SECTION

- [Output Queue Priorities Supported by the Junos OS CLI on PTX Series Routers | 793](#)
- [Scheduling Processes on PTX Series Routers | 793](#)
- [Strict-Priority and Scheduling Processes on PTX Series Routers | 795](#)

This topic covers class of service packet scheduling for interfaces on PTX Series routers:

### Output Queue Priorities Supported by the Junos OS CLI on PTX Series Routers

Output queues on the PTX Series interface hardware support these values for the queue priority—high, medium, low, and excess. The Junos OS supports five queue priority levels: strict-high, high, medium-high, medium-low, and low.



**NOTE:** If a strict-high-priority queue is constantly loaded to 100 percent of traffic capacity, other queues are starved. Queue starvation can cause the interface hardware to generate interrupts.

This starvation can be alleviated by using a rate-limiter on the strict-high queues.

### Scheduling Processes on PTX Series Routers

Physical interfaces on PTX Series routers support two mutually exclusive scheduling processes:

- *normal scheduling* (default mode) — A queue's transmit-rate is used to determine whether it is operating within the *guaranteed region* or in the *excess region*.

Within the *guaranteed region* (transmit-rates credits are positive), the scheduler uses the transmit rates to decide the bandwidth allocation. Queues that are at priority-level `low` or higher and have transmit-rate credits are serviced first by priority order and then within a priority level using packet round robin algorithm.



Within the *excess region* (for all queues in which transmit-rate credits are negative), CoS queues are selected based on the *weighted round-robin* (WRR) algorithm. If a queue does not have excess-rate configured, then its weight is set to 1.

If a queue is not configured with a `transmit-rate` statement (to specify the transmit rate or percentage of transmission capacity), then it will not be scheduled at priority level and only at excess level (unless priority is `strict-high`).

A queue must have a `transmit-rate` assigned to it in order to be scheduled at priority. `excess-rate` is used only to determine its weight when scheduled in the excess-region.

If multiple queues are in the excess region (queue priority `low`) and the `excess-rate` statement is used, then those queues are selected using the WRR algorithm.

The `remainder` keyword can be applied to `transmit-rate` as well as buffer size statements. This will cause the remaining portion of the specified resource to be assigned to the queue. The remaining resources for `transmit-rate` are the sum of all queues that specified a `transmit-rate` subtracted from the total `transmit-rate` available. If there are multiple queues assigned with the `remainder` keyword, the remainder of the resource is evenly divided amongst those queues.

There are two keywords that can be applied to the `transmit-rate` statement in order to limit the rate of a queue: `rate-limit` and `exact`, where `rate-limit` can be applied only to the `strict-high` queue and `exact` can be used for all other queues. The two keywords behave identically in that queues can transmit only up to the specified rate. All of their transmission will be scheduled at their configured priority level and they will never be scheduled at excess-priority level.



**BEST PRACTICE:** The `rate-limit` option of the `transmit-rate` configuration statement is allowed only on the `strict-high` queue. We recommend that you configure rate limit on `strict-high` queues because the other queues might not meet their guaranteed bandwidths.

Non `strict-high` queues can use the option `exact` to place a maximum `transmit-rate` limit on them, which is equivalent to `rate-limit`.

- *strict-priority scheduling* — Queues are processed in strict-priority order. There is no concept of guaranteed region and excess region. The packet scheduler is always operating in the guaranteed region, with the exception of priority `low`, which is always assigned to the excess priority level. The configured `transmit-rate` does not affect how the queue is serviced because packets are processed in order of queue priority. Among queues that are configured `low` priority, if `excess-rate` weights are configured, they are used by the hardware to perform WRR. Queues that are mapped to the same hardware priority or that have the same configured priority other than `low` are serviced in a packet round-robin fashion.

The queues are serviced in strict-priority order until they have reached their `transmit-rate` (that is, their guaranteed rate) and then are demoted to the excess level.





**NOTE:** The `rate-limit` and `exact` options of the `transmit-rate` configuration statement have no effect when strict-priority scheduling is configured.

To configure *strict-priority scheduling* for a physical interface on a PTX Series router, include the `strict-priority-scheduler` and `scheduler-map map-name` configuration statements in the traffic control profile you associate with an output interface.

### Strict-Priority and Scheduling Processes on PTX Series Routers

Table 71 on page 795 shows the different configurations available for Junos Priority Scheduler Modes, including those for Strict-Priority and Enhanced-Priority Modes. Table 71 on page 795 also shows how the output queue priority values in the Junos OS map to the output queue priorities supported by physical interfaces on PTX Series routers, and the scheduling action taken. Starting in Junos OS Release 17.4, the table shows differences for normal scheduling when `strict-high` is not configured, and for strict-priority scheduling.

**Table 71: Strict-Priority and Scheduling Processes on PTX Series Routers**

| Junos Priority | Scheduler Mode     | Normal                                    |        |                           | Strict Priority Scheduler |
|----------------|--------------------|-------------------------------------------|--------|---------------------------|---------------------------|
|                | Chassis Knob       | enhanced-priority-mode<br>(Junos OS only) |        | no-enhanced-priority-mode | *                         |
|                | Strict-High Config | No                                        | Yes    | *                         | *                         |
| strict-high    | —                  | —                                         | High   | High                      | High                      |
| high           | —                  | High                                      | Medium | High                      | High                      |
| medium-high    | —                  | Medium                                    | Low    | Medium                    | Medium                    |
| medium-low     | —                  | Medium                                    | Low    | Medium                    | Low                       |
| low            | —                  | Low                                       | Low    | Low                       | Excess                    |





**NOTE:** Packet scheduling is strict priority round-robin while the virtual output queues are in the guaranteed region.

After the virtual output queues consume their guaranteed credits, they are demoted to excess-priority scheduling, which is weighted round-robin.

The only exception is the strict-high priority, which is always scheduled as strict-high-priority.



PTX Series routers running Junos OS support enhanced-priority-mode and no-enhanced-priority-mode options under the [edit chassis fpc *fpc-slot* traffic-manager] configuration hierarchy.



PTX Series routers running Junos OS Evolved do not support enhanced-priority-mode mode.



**NOTE:** For PTX10K-LC1201 and PTX10K-LC1202 line cards on PTX10001-36MR, PTX10004, PTX10008, and PTX10016 routers running Junos OS Evolved:

- In strict priority scheduler mode, schedulers for queue 6 and queue 7 *must* have priority low (for Excess in hardware), and queues 0 through 5 should have higher priorities. These PTX models allow queues 6 and 7 to emulate two strict Excess priorities by setting large and small excess-rate values.
- When an FPC starts, if the system has any traffic-control-profiles *profile-name* strict-priority-scheduler configured, then all egress interfaces (et-\*) of that FPC treat queues 6 and 7 specially. For proper transmit scheduling, configure traffic-control-profiles *profile-name* strict-priority-scheduler for *all* port interfaces if *any* traffic control profile has strict-priority scheduler.

Change History Table

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

| Release | Description                                                                                                                                                  |
|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17.4    | Starting in Junos OS Release 17.4, the table shows differences for normal scheduling when strict-high is not configured, and for strict-priority scheduling. |



## RELATED DOCUMENTATION

[Understanding CoS CLI Configuration Statements on PTX Series Routers | 862](#)

[CoS Feature Differences Between PTX Series Packet Transport Routers and T Series Routers | 790](#)

[How Schedulers Define Output Queue Properties | 332](#)

[Understanding Scheduling on PTX Series Routers | 793](#)

[Example: Configuring Excess Rate for PTX Series Packet Transport Routers | 813](#)

*show interfaces voq*

*strict-priority-scheduler*

*traffic-control-profiles*

*transmit-rate*

## Virtual Output Queues on PTX Series Packet Transport Routers

### IN THIS SECTION

- [Introduction to Virtual Output Queues on PTX Series Packet Transport Routers | 797](#)
- [Understanding How VOQ Works on PTX Series Routers | 802](#)
- [Fabric Scheduling and Virtual Output Queues on PTX Series Routers | 805](#)
- [Understanding the Packet Forwarding Engine Fairness and Virtual Output Queue Process | 807](#)
- [VOQ Queue-depth Monitoring | 809](#)

This section describes the virtual output queue (VOQ) architecture on PTX Series Packet Transport Routers and includes the following topics:

### Introduction to Virtual Output Queues on PTX Series Packet Transport Routers

#### IN THIS SECTION

- [VOQ Architecture | 799](#)
- [Round-Trip Time Buffering | 799](#)
- [VOQ Advantages | 799](#)



## ● Does VOQ Change How I Configure CoS? | 800

This topic introduces the virtual output queue (VOQ) architecture on PTX Series Packet Transport routers and how it operates with the configurable class-of-service (CoS) components on PTX Series routers.

Junos and PTX Series hardware CoS features use *virtual output queues* on the *ingress* to buffer and queue traffic for each egress output queue. The PTX Series router supports up to eight egress output queues per output port (physical interface).

The traditional method of forwarding traffic through a router is based on buffering ingress traffic in input queues on ingress interfaces, forwarding the traffic across the fabric to output queues on egress interfaces, and then buffering traffic again on the output queues before transmitting the traffic to the next hop. The traditional method of queueing packets on an ingress port is storing traffic destined for different egress ports in the same input queue (buffer).

During periods of congestion, the router might drop packets at the egress port, so the router might spend resources transporting traffic across the switch fabric to an egress port, only to drop that traffic instead of forwarding it. And because input queues store traffic destined for different egress ports, congestion on one egress port could affect traffic on a different egress port, a condition called head-of-line blocking (HOLB).

Virtual output queue (VOQ) architecture takes a different approach:

- Instead of separate physical buffers for input and output queues, the PTX Series router uses the physical buffers on the ingress pipeline of each Packet Forwarding Engine to store traffic for every egress port. Every output queue on an egress port has buffer storage space on every ingress pipeline on all of the Packet Forwarding Engines on the router. The mapping of ingress pipeline storage space to output queues is 1-to-1, so each output queue receives buffer space on each ingress pipeline.
- Instead of one input queue containing traffic destined for multiple different output queues (a one-to-many mapping), each output queue has a dedicated VOQ comprised of the input buffers on each Packet Forwarding Engine that are dedicated to that output queue (a 1-to-1 mapping). This architecture prevents communication between any two ports from affecting another port.
- Instead of storing traffic on a physical output queue until it can be forwarded, a VOQ does not transmit traffic from the ingress port across the fabric to the egress port until the egress port has the resources to forward the traffic. A VOQ is a collection of input queues (buffers) that receive and store traffic destined for one output queue on one egress port. Each output queue on each egress port has its own dedicated VOQ, which consists of all of the input queues that are sending traffic to that output queue.



A VOQ is a collection of input queues (buffers) that receive and store traffic destined for one output queue on one egress port. Each output queue on each egress port has its own dedicated VOQ, which consists of all of the input queues that are sending traffic to that output queue.

### **VOQ Architecture**

A VOQ represents the ingress buffering for a particular output queue. Each of the Packet Forwarding Engines in the PTX Series router uses a specific output queue. The traffic stored on the Packet Forwarding Engines comprises the traffic destined for one particular output queue on one port, and is the VOQ for that output queue.

A VOQ is distributed across all of the Packet Forwarding Engines in the router that are actively sending traffic to that output queue. Each output queue is the sum of the total buffers assigned to that output queue across all of the Packet Forwarding Engines in the router. So the output queue itself is virtual, not physical, although the output queue is comprised of physical input queues.

### **Round-Trip Time Buffering**

Although there is no output queue buffering during periods of congestion (no long-term storage), there is a small physical output queue buffer on egress line cards to accommodate the round-trip time for traffic to traverse the fabric from ingress to egress. The round-trip time consists of the time it takes the ingress port to request egress port resources, receive a grant from the egress port for resources, and transmit the data across the fabric.

That means if a packet is not dropped at the router ingress, and the router forwards the packet across the fabric to the egress port, the packet will not be dropped and will be forwarded to the next hop. All packet drops take place in the ingress pipeline.

### **VOQ Advantages**

VOQ architecture provides two major advantages:

#### **Eliminate Head-of-Line Blocking**

VOQ architecture eliminates head-of-line blocking (HOLB) issues. On non-VOQ switches, HOLB occurs when congestion at an egress port affects a different egress port that is not congested. HOLB occurs when the congested port and the non-congested port share the same input queue on an ingress interface.

VOQ architecture avoids HOLB by creating a different dedicated virtual queue for each output queue on each interface.

Because different egress queues do not share the same input queue, a congested egress queue on one port cannot affect an egress queue on a different port. For the same reason, a congested egress queue



on one port cannot affect another egress queue on the same port—each output queue has its own dedicated virtual output queue composed of ingress interface input queues.

Performing queue buffering at the ingress interface ensures that the router only sends traffic across the fabric to an egress queue if that egress queue is ready to receive that traffic. If the egress queue is not ready to receive traffic, the traffic remains buffered at the ingress interface.

## **Increase Fabric Efficiency and Utilization**

Traditional output queue architecture has some inherent inefficiencies that VOQ architecture addresses.

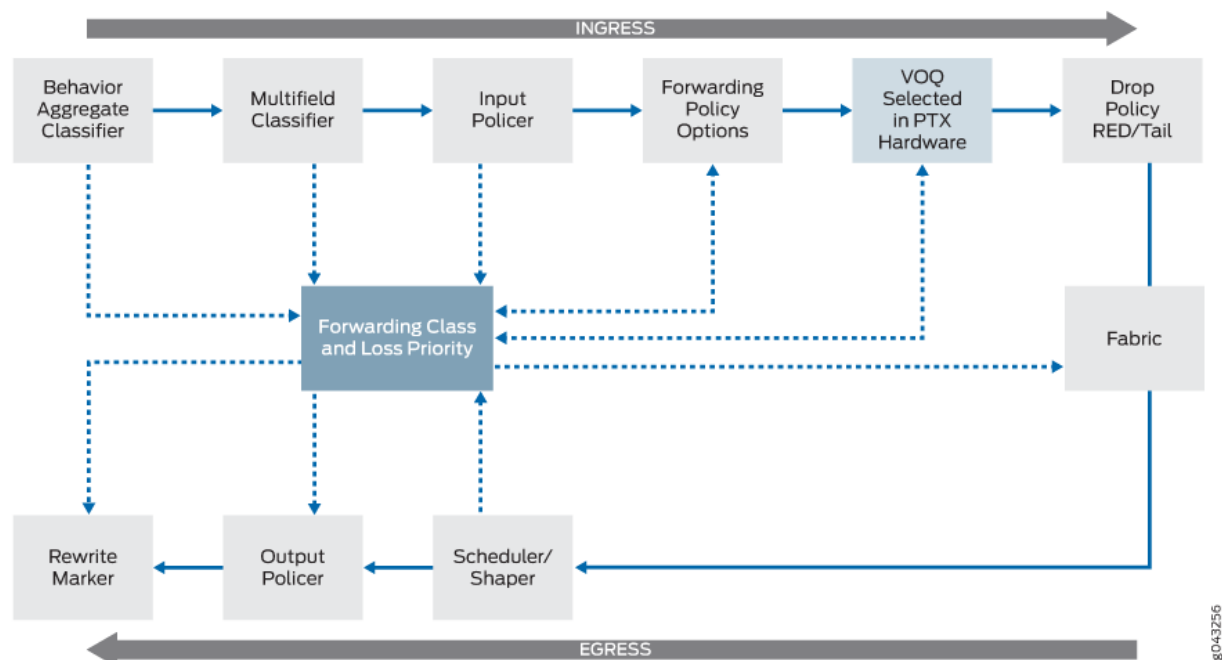
- **Packet buffering**—Traditional queueing architecture buffers each packet twice in long-term DRAM storage, once at the ingress interface and once at the egress interface. VOQ architecture buffers each packet only once in long-term DRAM storage, at the ingress interface. The fabric is fast enough to be transparent to egress CoS policies, so instead of buffering packets a second time at the egress interface, the router can forward traffic at a rate that does not require deep egress buffers, without affecting the configured egress CoS policies (scheduling).
- **Consumption of resources**—Traditional queueing architecture sends packets from the ingress interface input queue (buffer), across the fabric, to the egress interface output queue (buffer). At the egress interface, packets might be dropped, even though the router has expended resources transporting the packets across the fabric and storing them in the egress queue. VOQ architecture does not send packets across the fabric to the egress interface until the egress interface is ready to transmit the traffic. This increases system utilization because no resources are wasted transporting and storing packets that are dropped later.

## **Does VOQ Change How I Configure CoS?**

There are no changes to the way you configure the CoS features. [Figure 52 on page 801](#) shows the Junos OS and PTX Series hardware CoS components and VOQ selection, illustrating the sequence in which they interact.



Figure 52: Packet Flow Through CoS Components on PTX Series Routers



The VOQ selection process is performed by ASICs that use either the behavior aggregate (BA) classifier or the multifield classifier, depending on your configuration, to select one of the eight possible virtual output queues for an egress port. The virtual output queues on the ingress buffer data for the egress port based on your CoS configuration.

Although the CoS features do not change, there are some operational differences with VOQ:

- Random early detection (RED) occurs on the ingress Packet Forwarding Engines. With routers that support only egress output queuing, RED and associated congestion drops occur on the egress. Performing RED on the ingress saves valuable resources and increases router performance.

Although RED occurs on the ingress with VOQ, there is no change to how you configure the drop profiles.

- Fabric scheduling is controlled through request and grant control messages. Packets are buffered in ingress virtual output queues until the egress Packet Forwarding Engine sends a grant message to the ingress Packet Forwarding Engine indicating it is ready to receive them. For details on fabric scheduling, see ["Fabric Scheduling and Virtual Output Queues on PTX Series Routers" on page 805](#).



# Understanding How VOQ Works on PTX Series Routers

## IN THIS SECTION

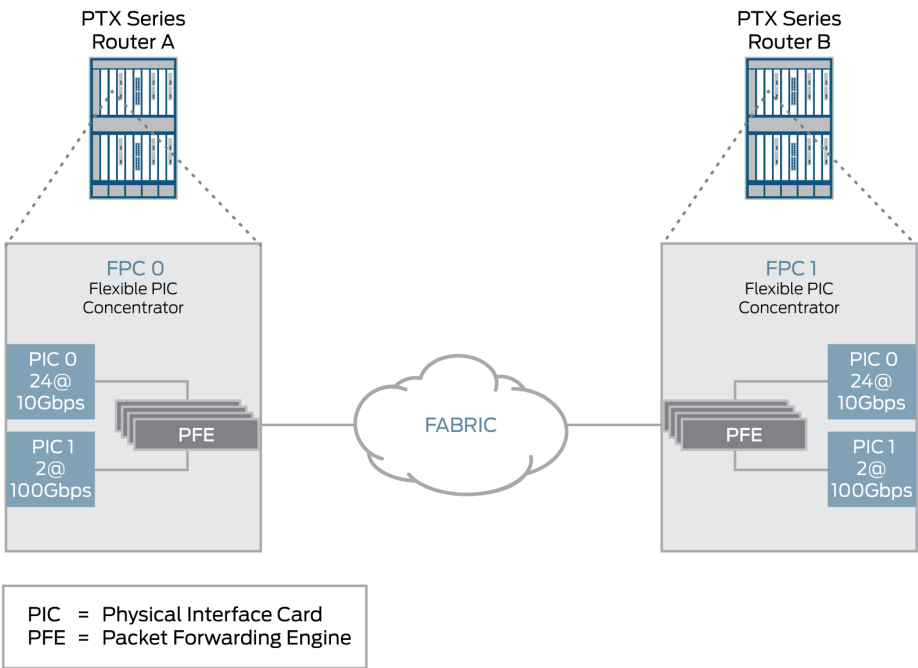
- [Understanding the Components of the VOQ Process | 802](#)
- [Understanding the VOQ Process | 803](#)

This topic describes how the VOQ process works on PTX Series routers.

## Understanding the Components of the VOQ Process

[Figure 53 on page 802](#) shows the hardware components of the PTX Series routers involved in the VOQ process.

**Figure 53: VOQ Components on PTX Series Routers**



These components perform the following functions:

- **Physical Interface Card (PIC)**—Provides the physical connection to various network media types, receiving incoming packets from the network and transmitting outgoing packets to the network.



- **Flexible PIC Concentrator (FPC)**—Connects the PICs installed in it to the other packet transport router components. You can have up to eight FPCs per chassis.
- **Packet Forwarding Engine**—Provides Layer 2 and Layer 3 packet switching and encapsulation and de-encapsulation, forwarding and route lookup functions, and manages packet buffering and the queuing of notifications. The Packet Forwarding Engine receives incoming packets from the PICs installed on the FPC and forwards them through the switch planes to the appropriate destination port.
- **Output queues**—(Not shown) PTX Series routers support up to eight output queues per output port (physical interface). These output queues are controlled by the CoS scheduler configuration, which establishes how to handle the traffic within the output queues for transmission onto the switch fabric. In addition, these egress output queues control when packets are sent from the virtual output queues on the ingress to the egress output queues.

### Understanding the VOQ Process

PTX Series routers support up to eight output queues per output port (physical interface). These output queues are controlled by the CoS scheduler configuration, which establishes how to handle the traffic within the output queues for transmission onto the fabric. In addition, these egress output queues control when packets are sent from the virtual output queues on the ingress to the egress output queues.

For every egress output queue, the VOQ architecture provides *virtual* queues on each and every ingress Packet Forwarding Engine. These queues are referred to as virtual because the queues physically exist on the ingress Packet Forwarding Engine *only* when the line card actually has packets enqueued to it.

[Figure 54 on page 804](#) shows three ingress Packet Forwarding Engines—PFE0, PFE1, and PFE2. Each ingress Packet Forwarding Engine provides up to eight virtual output queues (PFE*n*.e0.q0 through PFE*n*.e0.q7) for the single egress port 0. The egress Packet Forwarding Engine PFE*n* distributes the bandwidth to each ingress VOQ in a round-robin fashion.

For example, egress PFE N's VOQ e0.q0 has 10 Gbps of bandwidth available to it. PFE 0 has an offered load 10 Gbps to e0.q0, PFE1 and PFE2 have an offered load of 1Gbps to e0.q0. The result is that PFE1 And PFE2 will get 100 percent of their traffic through, while PFE0 will only get 80 percent of its traffic through.



Figure 54: Virtual Output Queues on PTX Series Routers

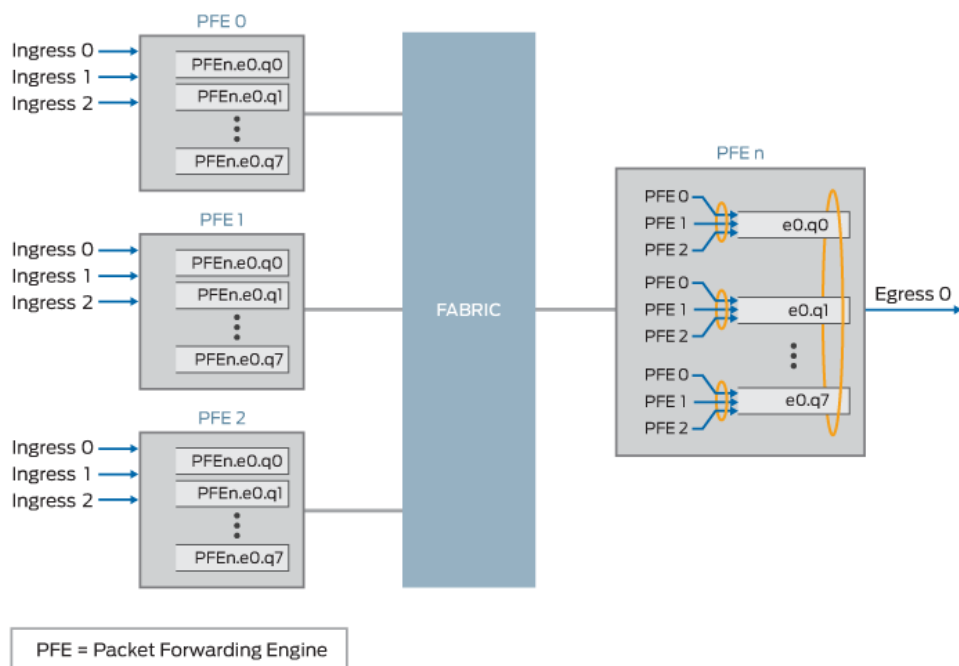
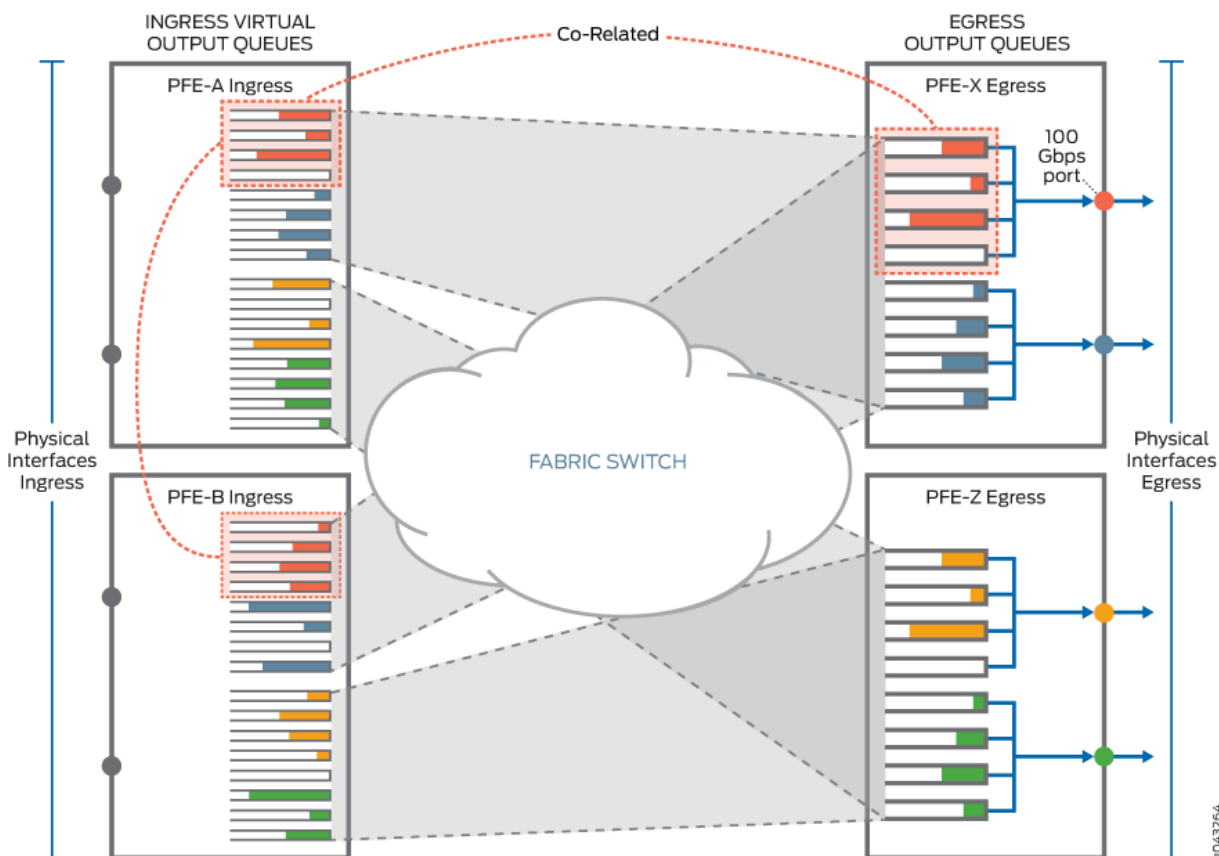


Figure 55 on page 805 illustrates an example of the correlation between the egress output queues and the ingress virtual output queues. On the egress side, PFE-X has a 100 Gbps port, which is configured with four different forwarding classes. As a result, the 100 Gbps egress output port on PFE-X uses four out of eight available egress output queues (as denoted by the four queues highlighted with dashed-orange lines on PFE-X), and the VOQ architecture provides four corresponding *virtual* output queues on *each* ingress Packet Forwarding Engine (as denoted by the four virtual queues on PFE-A and PFE-B highlighted with dashed-orange lines). The virtual queues on PFE-A and PFE-B exist only when there is traffic to be sent.



Figure 55: Example of VOQ



### Fabric Scheduling and Virtual Output Queues on PTX Series Routers

This topic describes the fabric scheduling process on PTX Series routers that use VOQ.

VOQ uses request and grant messages to control fabric scheduling on PTX Series routers. The egress Packet Forwarding Engines control data delivery from the ingress virtual output queues by using request and grant messages. The virtual queues buffer packets on the ingress until the egress Packet Forwarding Engine confirms that it is ready to receive them by sending a grant message to the ingress Packet Forwarding Engine.



Figure 56: Fabric Scheduling and Virtual Output Queues Process

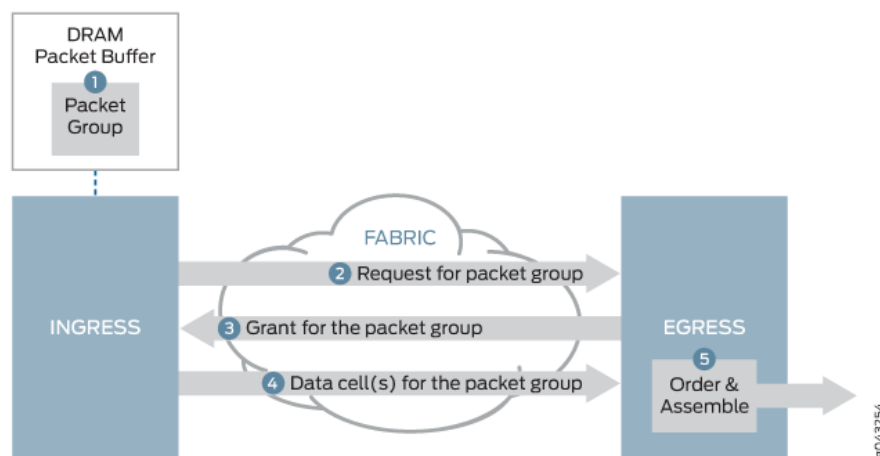


Figure 56 on page 806 illustrates the fabric scheduling process used by PTX Series routers with VOQ. When packets arrive at an ingress port, the ingress pipeline stores the packet in the ingress queue associated with the destination output queue. The router makes the buffering decision after performing the packet lookup. If the packet belongs to a forwarding class for which the maximum traffic threshold has been exceeded, the packet may not be buffered and might be dropped. The scheduling process works as follows:

1. An ingress Packet Forwarding Engine receives a packet and buffers it in virtual queues, then groups the packet with other packets destined for the same egress interface and data output queue.
2. The ingress line card Packet Forwarding Engine sends a request, which contains a reference to the packet group, over the fabric to the egress Packet Forwarding Engine.
3. When there is available egress bandwidth, the egress line card grant scheduler responds by sending a bandwidth grant to the ingress line card Packet Forwarding Engine. .
4. When the ingress line card Packet Forwarding Engine receives the grant from the egress line card Packet Forwarding Engine, the ingress Packet Forwarding Engine segments the packet group and sends all of the pieces over the fabric to the egress Packet Forwarding Engine.
5. The egress Packet Forwarding Engine receives the pieces, reassembles them into the packet group, and enqueues individual packets to a data output queue corresponding to the virtual output queue.

Ingress packets remain in the VOQ on the ingress port input queues until the output queue is ready to accept and forward more traffic.

Under most conditions, the fabric is fast enough to be transparent to egress class-of-service (CoS) policies, so the process of forwarding traffic from the ingress pipeline, across the fabric, to egress ports, does not affect the configured CoS policies for the traffic. The fabric only affects CoS policy if there is a fabric failure or if there is an issue of port fairness.



When a packet ingresses and egresses the same Packet Forwarding Engine (local switching), the packet does not traverse the fabric. However, the router uses the same request and grant mechanism to receive egress bandwidth as packets that cross the fabric, so locally routed packets and packets that arrive at a Packet Forwarding Engine after crossing the fabric are treated fairly when the traffic is vying for the same output queue.

## Understanding the Packet Forwarding Engine Fairness and Virtual Output Queue Process

### IN THIS SECTION

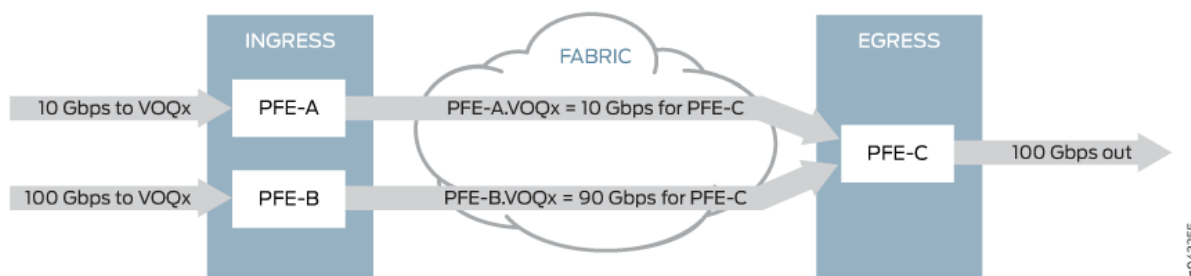
- [Handling Congestion | 808](#)

This topic describes the Packet Forwarding Engine fairness scheme used with VOQ on PTX Series routers.

Packet Forwarding Engine fairness means that all Packet Forwarding Engines are treated equally from an egress perspective. If multiple egress Packet Forwarding Engines need to transmit data from the same virtual output queue, they are serviced in round-robin fashion. Servicing of virtual output queues is *not* dependent upon the load that is present at each of the source Packet Forwarding Engines.

[Figure 57 on page 807](#) illustrates the Packet Forwarding Engine fairness scheme used with VOQ in a simple example with three Packet Forwarding Engines. Ingress PFE-A has a single stream of 10 Gbps data destined for VOQx on PFE-C. PFE-B has a single stream of 100 Gbps data also destined for VOQx on PFE-C. On PFE-C, VOQx is serviced by a 100 Gbps interface and that is the only active virtual output queue on that interface.

**Figure 57: Packet Forwarding Engine Fairness with Virtual Output Queue Process**





In [Figure 57 on page 807](#), we have a total of 110 Gbps of source data destined for a 100 Gbps output interface. As a result, we need to drop 10 Gbps of data. Where does the drop occur and how does this drop affect traffic from PFE-A versus PFE-B?

Because PFE-A and PFE-B are serviced in round-robin fashion by egress PFE-C, all 10 Gbps of traffic from PFE-A makes it through to the egress output port. However, 10 Gbps of data is dropped on PFE-B, allowing only 90 Gbps of data from PFE-B to be sent to PFE-C. So, the 10 Gbps stream has a 0% drop and the 100 Gbps stream has only a 10% drop.

However, if PFE-A and PFE-B were each sourcing 100 Gbps of data, then they would each drop 50 Gbps of data. This is because the egress PFE-C actually controls the servicing and drain rate on the ingress virtual queues using the round-robin algorithm. With the round-robin algorithm, higher bandwidth sources are always penalized when multiple sources are present. The algorithm attempts to make the two sources equal in bandwidth; however, because it cannot raise the bandwidth of the slower source, it drops the bandwidth of the higher source. The round robin algorithm continues this sequence until the sources have equal egress bandwidth.

Each ingress Packet Forwarding Engine provides up to eight virtual output queues for a single egress port. The egress Packet Forwarding Engine distributes the bandwidth to each ingress virtual output queue; therefore they will receive equal treatment regardless of their presented load. The drain-rate of a queue is the rate at which a queue is draining. The egress Packet Forwarding Engine divides its bandwidth for each output queue equally across the ingress Packet Forwarding Engines. So, the drain-rate of each ingress Packet Forwarding Engine = Drain-rate of output queue / Number of ingress Packet Forwarding Engines.

## Handling Congestion

There are two main types of congestion that can occur:

- **Ingress congestion** — Occurs when the ingress Packet Forwarding Engine has more offered load than the egress can handle. The ingress congestion case, is very similarly to a traditional router in that the queues build-up and once they cross their configured threshold, packets are dropped.
- **Egress congestion** — Occurs when the sum of all the ingress Packet Forwarding Engines exceeds the capability of the egress router. All drops are performed on the ingress Packet Forwarding Engines. However, the size of the ingress queue is attenuated by the queue's drain-rate (how fast the egress Packet Forwarding Engine is requesting packets). This rate is essentially determined by the rate that requests are being converted in to grants by the egress Packet Forwarding Engine. The egress Packet Forwarding Engine services the request-to-grant conversion in round-robin fashion; it is not dependent on the ingress Packet Forwarding Engines offered load. For instance, if the ingress Packet Forwarding Engine's drain-rate is half of what it expects it to be (as is the case when 2 ingress Packet Forwarding Engines are presenting an oversubscribed load for the target output queue), then the ingress Packet Forwarding Engine's reduce the size of this queue to be half of its original size (when it was getting its full drain rate).



## VOQ Queue-depth Monitoring

### IN THIS SECTION

- [Configure VOQ Queue-depth Monitoring | 810](#)

VOQ queue-depth monitoring, or latency monitoring, measures peak queue occupancy of a VOQ. This feature enables the reporting of peak queue length for a given physical interface for each individual Packet Forwarding Engine (PFE).



**NOTE:** In addition to the peak queue-length data, each queue also maintains drop statistics and time-averaged queue length on the ingress data path. Also, each queue maintains queue transmission statistics on the egress data path.

In a typical deployment scenario that uses strict-priority scheduling, a HIGH priority queue can starve LOW priority queues. Thus the packets in such LOW priority queues can remain longer than desired. You can use this VOQ queue-depth monitoring feature, along with queue transmission statistics, to detect such stalled conditions.



**NOTE:** You can only enable VOQ queue-depth monitoring on transit WAN interfaces.

To enable VOQ queue-depth monitoring on an interface, you first create a monitoring profile, and then attach that profile to the interface. If you attach a monitoring profile to an aggregated Ethernet (ae-) interface, each member interface has its own dedicated hardware VOQ monitor, unless you also apply the shared option to the monitoring profile attached to the ae- interface.

By default, a monitoring profile that you assign to an ae- interface replicates across all members of the ae- interface. The monitoring profile also reports VOQ depth individually on each interface. On large systems, this process can quickly consume the maximum supported hardware monitoring profile IDs. To conserve monitoring profile IDs, include the shared option at the `[set class-of-service interfaces ae-interface monitoring-profile profile-name]` hierarchy level. The configured shared option creates only one monitoring profile ID to share across all member interfaces. The option also reports the largest peak on a member interface as the common peak for the ae- interface.



**NOTE:** You cannot enable the shared option on mixed mode ae- interfaces.

Each monitoring profile consists of one or more export filters. An export filter defines a peak queue-length percentage threshold for one or more queues on the physical interface. Once the defined peak



queue-length percentage threshold is met for any queue in the export filter, Junos exports the VOQ telemetry data for all queues in the export filter.



**NOTE:** The queue-depth monitoring data goes out *only* through a telemetry channel. In addition to configuring a monitoring profile (as shown below) you *must* initiate a regular sensor subscription in order for the data to go out. There is no CLI display option.

## Configure VOQ Queue-depth Monitoring

Configure VOQ queue-depth monitoring to export queue utilization data. You can use this data to monitor micro-bursts and also assist in identifying stalled transit output queues. To configure VOQ queue-depth monitoring:

1. Configure the monitoring profile.
2. Attach the monitoring profile to an interface.

To configure the monitoring profile:

1. Name the monitoring profile. For example:

```
[edit class-of-service]
set monitoring-profile mp1
```

2. Name an export filter for the monitoring profile. For example:

```
[edit class-of-service monitoring-profile mp1]
set export-filters ef1
```

3. Define which queues (0 through 7) belong to the export filter. For example:

```
[edit class-of-service monitoring-profile mp1 export-filters ef1]
set queue [0 1]
```

4. (Optional) Define the threshold peak queue length percentage to export VOQ telemetry data. The default percentage is 0. For example:

```
[edit class-of-service monitoring-profile mp1 export-filters ef1]
set peak-queue-length percent 50
```



5. (Optional) Define one or more other export filters for the monitoring profile. For example:

```
[edit class-of-service monitoring-profile mp1]
set export-filters ef2 queue [2 3]
```

6. Commit your changes.

To attach the monitoring profile to an interface:

1. Attach the monitoring profile to an interface. For example:

```
[edit class-of-service]
set interfaces et-0/0/1 monitoring-profile mp1
set interfaces ae0 monitoring-profile mp1 shared
```

2. Commit your changes.

Check your configuration. For example:

```
[edit class-of-service]
user@host# show
monitoring-profile mp1 {
  export-filters ef1 {
    peak-queue-length {
      percent 50;
    }
    queue [ 0 1 ];
  }
  export-filters ef2 {
    queue [ 2 3 ];
  }
}
interfaces {
  ae0 {
    monitoring-profile mp1 shared;
  }
  et-0/0/1 {
    monitoring-profile mp1;
```



```
}
}
```

Run these show commands to verify your configuration:

```
user@host> show class-of-service interface et-0/0/1
Physical interface: et-0/0/1, Index: 1098
Maximum usable queues: 8, Queues in use: 4
Exclude aggregate overhead bytes: disabled
Logical interface aggregate statistics: disabled
  Scheduler map: default, Index: 0
  Congestion-notification: Disabled
Monitoring Profile Name: mp1
```

```
Logical interface: et-0/0/1.16386, Index: 1057
```

```
user@host> show class-of-service interface ae0
Physical interface: ae0, Index: 7860
Maximum usable queues: 8, Queues in use: 4
Exclude aggregate overhead bytes: disabled
Logical interface aggregate statistics: disabled
  Scheduler map: default, Index: 0
  Congestion-notification: Disabled
Monitoring Profile Name: mp1 [shared]
```

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

**Monitoring profile: mp1**

| Export filter | Queue Number | Peak Queue Length |
|---------------|--------------|-------------------|
| ef1           | 0            | 50%               |
| ef1           | 1            | 50%               |
| ef2           | 2            | 0%                |
| ef2           | 3            | 0%                |



**NOTE:** As you can see from this example, not setting a peak-queue-length percent for an export filter defaults the percentage to 0 percent, as export filter ef2 shows. This example shows different queues on the physical interface having different peak queue length thresholds for exporting VOQ telemetry data.



## RELATED DOCUMENTATION

*show interfaces voq*

[The Junos OS CoS Components Used to Manage Congestion and Control Service Levels | 6](#)

[Packet Flow Through the Junos OS CoS Process Overview | 18](#)

## Example: Configuring Excess Rate for PTX Series Packet Transport Routers

### IN THIS SECTION

- [Requirements | 813](#)
- [Overview | 813](#)
- [Configuration | 814](#)
- [Verification | 820](#)

You can configure excess rate to customize the distribution of available excess bandwidth among the queues for PTX Series Packet Transport Routers. When excess rate is not configured, the excess bandwidth available is distributed in proportion to the transmit rates allocated to the queues.

### Requirements

This example uses the following hardware and software components:

- One PTX Series Packet Transport Router
- Junos OS Release 12.1X48R2 or later

### Overview

This set of examples illustrates how you configure schedulers for the PTX Series Packet Transport Router to distribute the remaining bandwidth (excess rate) among the configured queues.

When you configure excess rate, use the following guidelines:

- The transmit-rate statements of the configured schedulers can add up to at most 100 percent.
- All queues on the PTX Series Packet Transport Router have the same excess priority. Excess priority configuration is not supported.



- If a strict-high-priority queue is configured and is rate-limited, this queue gets the rate-limited bandwidth first. Then the configured `transmit-rate` value of other queues is met (regardless of queue priority), and finally the excess bandwidth is distributed in proportion to the configured `excess-rate` values.



**BEST PRACTICE:** We recommend that you configure rate limit on strict-high queues because the other queues might not meet their guaranteed bandwidths. See *transmit-rate*.

## Configuration

### IN THIS SECTION

- [Configuring Schedulers Without Specifying Excess Rate | 814](#)
- [Configuring Schedulers by Specifying Excess Rate | 816](#)
- [Configuring Schedulers to Control Excess Rate for Non-High-Priority Queues | 818](#)

To configure excess rate, perform one or more of these tasks:

### Configuring Schedulers Without Specifying Excess Rate

#### CLI Quick Configuration

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

```
set class-of-service schedulers sched_queue_0 transmit-rate percent 20
set class-of-service schedulers sched_queue_1 transmit-rate percent 40
set class-of-service schedulers sched_queue_2 transmit-rate percent 10
set class-of-service schedulers sched_queue_3 transmit-rate percent 10
```

### Step-by-Step Procedure

In this example, four queues are configured and each associated scheduler is assigned the indicated transmit rate. Across the four queues, the transmit rate totals to 80 percent. No excess rate is



configured. Assuming that each queue has loads greater than or equal to the configured transmit rate, the remaining 20 percent of the bandwidth is distributed in proportion to the configured transmit rates (20:40:10:10):

- sched\_queue\_0—5% (20% of the guaranteed rate plus 5% of the remaining bandwidth is 25%)
- sched\_queue\_1—10% (40% of the guaranteed rate plus 10% of the remaining bandwidth is 50%)
- sched\_queue\_2—2.5% (10% of the guaranteed rate plus 2.5% of the remaining bandwidth is 12.5%)
- sched\_queue\_3—2.5% (10% of the guaranteed rate plus 2.5% of the remaining bandwidth is 12.5%)

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

To configure the schedulers:

1. Create the scheduler for queue 0:

```
[edit class-of-service]
user@host# set schedulers sched_queue_0 transmit-rate percent 20
```

2. Create the scheduler for queue 1:

```
[edit class-of-service]
user@host# set schedulers sched_queue_1 transmit-rate percent 40
```

3. Create the scheduler for queue 2:

```
[edit class-of-service]
user@host# set schedulers sched_queue_2 transmit-rate percent 10
```

4. Create the scheduler for queue 3:

```
[edit class-of-service]
user@host# set schedulers sched_queue_3 transmit-rate percent 10
```



## Results

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

```

sched_queue_0 {
    transmit-rate percent 20;
}
sched_queue_1 {
    transmit-rate percent 40;
}
sched_queue_2 {
    transmit-rate percent 10;
}
sched_queue_3 {
    transmit-rate percent 10;
}

```

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

## Configuring Schedulers by Specifying Excess Rate

### CLI Quick Configuration

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

```

set class-of-service schedulers sched_queue_0 transmit-rate percent 25
set class-of-service schedulers sched_queue_0 excess-rate percent 25
set class-of-service schedulers sched_queue_1 transmit-rate percent 25
set class-of-service schedulers sched_queue_1 excess-rate percent 50
set class-of-service schedulers sched_queue_2 transmit-rate percent 25
set class-of-service schedulers sched_queue_3 transmit-rate percent 25

```

## Step-by-Step Procedure

In this example, four schedulers are configured and each is assigned a transmit rate of 25 percent. Queue 0 is configured with 25 percent and queue 1 with 50 percent of the excess rate. If the offered



load through queue 2 is only 10 percent, the remaining bandwidth is distributed as: queue excess rate / total excess rate \* remaining bandwidth percentage. If a queue has transmit rate configured but not excess rate, the excess rate for that queue is 1. In this example, the excess rate ratio is 25:50:1:1, which yields the following distribution of the 15 percent remaining bandwidth from queue 2:

- sched\_queue\_0—4.93% ( $25 / 76 * 15\%$ )
- sched\_queue\_1—9.87% ( $50 / 76 * 15\%$ )
- sched\_queue\_3—0.197% ( $1 / 76 * 15\%$ )

When the offered load on queue 2 increases to 25 percent or greater, the other queues get only their configured transmit rates.

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

To configure the schedulers:

1. Create the scheduler for queue 0:

```
[edit class-of-service]
user@host# set schedulers sched_queue_0 transmit-rate percent 25
user@host# set schedulers sched_queue_0 excess-rate percent 25
```

2. Create the scheduler for queue 1:

```
[edit class-of-service]
user@host# set schedulers sched_queue_1 transmit-rate percent 25
user@host# set schedulers sched_queue_1 excess-rate percent 50
```

3. Create the scheduler for queue 2:

```
[edit class-of-service]
user@host# set schedulers sched_queue_2 transmit-rate percent 25
```

4. Create the scheduler for queue 3:

```
[edit class-of-service]
user@host# set schedulers sched_queue_3 transmit-rate percent 25
```



## Results

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

```

sched_queue_0 {
    transmit-rate percent 25;
    excess-rate percent 25;
}
sched_queue_1 {
    transmit-rate percent 25;
    excess-rate percent 50;
}
sched_queue_2 {
    transmit-rate percent 25;
}
sched_queue_3 {
    transmit-rate percent 25;
}
```

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

## Configuring Schedulers to Control Excess Rate for Non-High-Priority Queues

### CLI Quick Configuration

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

```

set class-of-service schedulers sched_queue_0 transmit-rate percent 90
set class-of-service schedulers sched_queue_0 priority high
set class-of-service schedulers sched_queue_1 transmit-rate percent 10
set class-of-service schedulers sched_queue_1 priority low
set class-of-service schedulers sched_queue_2 excess-rate percent 10
set class-of-service schedulers sched_queue_3 excess-rate percent 30
```



## Step-by-Step Procedure

In this example, the scheduler for queue 0 is configured to transmit up to 90 percent of traffic if there is enough offered load. When the traffic to queue 0 is less than 90 percent, excess rate is configured to distribute the remaining bandwidth in the ratio 1:1:10:30 (when the offered load on queue 1 is greater than 10 percent), which yields the following distribution of the remaining bandwidth from queue 0:

- $\text{sched\_queue\_1} = 0.0244 * x\% (1 / 41 * \text{remaining bandwidth } (x)\%)$
- $\text{sched\_queue\_2} = 0.244 * x\% (10 / 41 * \text{remaining bandwidth } (x)\%)$
- $\text{sched\_queue\_3} = 0.732 * x\% (30 / 41 * \text{remaining bandwidth } (x)\%)$



**NOTE:** Although the transmit-rate values on queues can add up to at most 100 percent, the excess-rate value does not have this restriction because it is a ratio.

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

To configure the schedulers:

1. Create the scheduler for queue 0:

```
[edit class-of-service]
user@host# set schedulers sched_queue_0 transmit-rate percent 90
user@host# set schedulers sched_queue_0 priority high
```

2. Create the scheduler for queue 1:

```
[edit class-of-service]
user@host# set schedulers sched_queue_1 transmit-rate percent 10
user@host# set schedulers sched_queue_1 priority low
```

3. Create the scheduler for queue 2:

```
[edit class-of-service]
user@host# set schedulers sched_queue_2 excess-rate percent 10
```



#### 4. Create the scheduler for queue 3:

```
[edit class-of-service]
user@host# set schedulers sched_queue_3 excess-rate percent 30
```

## Results

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

```
sched_queue_0 {
  transmit-rate percent 90;
  priority high;
}
sched_queue_1 {
  transmit-rate percent 10;
  priority low;
}
sched_queue_2 {
  excess-rate percent 10;
}
sched_queue_3 {
  excess-rate percent 30;
}
```

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

## Verification

### IN THIS SECTION

- [Verifying the Excess Rate Configuration | 821](#)



## Verifying the Excess Rate Configuration

### Purpose

Verify that the excess rate configuration is producing the results you expect.

### Action

From operational mode, enter the `show interfaces queue interface` command for the physical interface to verify.

### Meaning

The show command output lists the traffic by queue and forwarding class names. Verify that the Bytes field for active queues on the specified physical interface match the proportions you expect from the excess rate configuration.

## RELATED DOCUMENTATION

---

[How Schedulers Define Output Queue Properties | 332](#)

---

[Configuring a Scheduler | 670](#)

---

*excess-rate*

---

[CoS Features and Limitations on PTX Series Routers | 785](#)

## Identifying the Source of RED Dropped Packets on PTX Series Routers

This topic describes how to identify the source of random early detection (RED) dropped packets.

Junos OS and PTX Series hardware CoS features use virtual output queues (VOQs) on the ingress to buffer and queue traffic for each egress output queue.

VOQ is a queuing strategy that eliminates congestion drops on the egress and alleviates head-of-line blocking. Head-of-line blocking is a condition in which a queue of packets is blocked from making progress because the packet at the head of the queue is waiting for resources to become available, while other packets behind this packet could be serviced. For example, if the ingress has a single queue for an egress Packet Forwarding Engine, then packets destined for a slow, congested interface can block packets destined for a fast, uncongested interface attached to the same egress Packet Forwarding Engine.

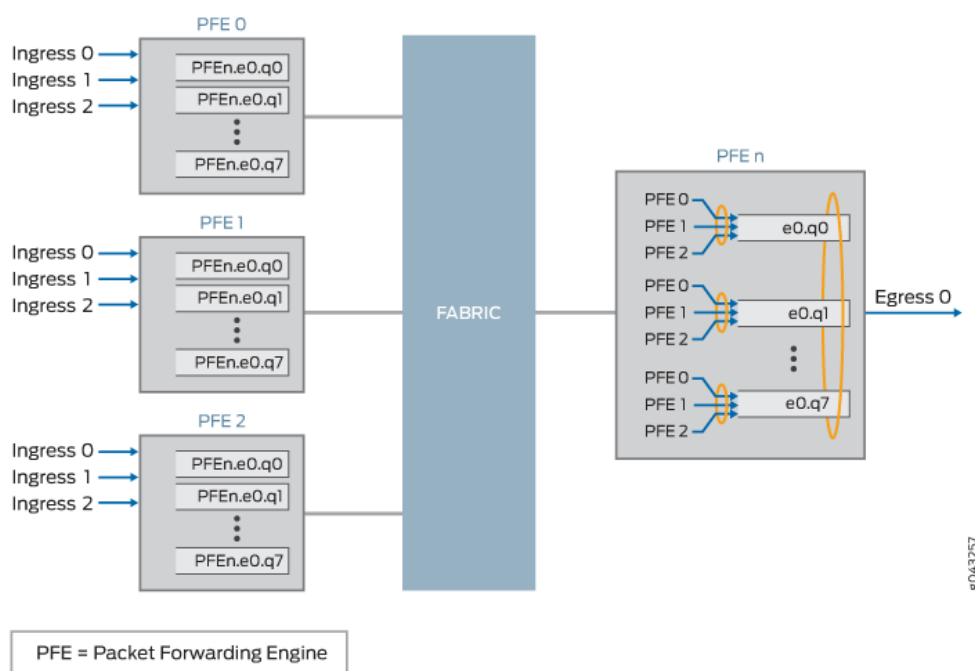


With VOQ, *virtual* queues are maintained on the ingress Packet Forwarding Engines, instead of on the egress Packet Forwarding Engine. However, the scheduling of the ingress virtual output queues is controlled by the egress Packet Forwarding Engine. For every egress output queue (shallow buffer), the VOQ architecture provides *virtual* queues on each and every ingress Packet Forwarding Engine. These queues are referred to as virtual because the queues physically exist on the ingress Packet Forwarding Engine *only* when the line card actually has packets enqueued to it.

Figure 58 on page 822 shows three ingress Packet Forwarding Engines—PFE0, PFE1, and PFE2. Each ingress Packet Forwarding Engine provides up to eight virtual output queues (PFE*n*.e0.q0 through PFE*n*.e0.q7) for the single egress port 0. The egress Packet Forwarding Engine PFE*n* distributes the bandwidth to each ingress VOQ in a round-robin fashion; therefore they will receive equal treatment regardless of their presented load.

For example, egress PFE*n*'s VOQ e0.q0 has 10 Gbps of bandwidth available to it. PFE0 has an offered load of 10 Gbps to e0.q0, whereas PFE1 and PFE2 have an offered load of 1 Gbps to e0.q0. The result is that PFE1 and PFE2 get 100 percent of their traffic through, whereas PFE0 gets only 80 percent of its traffic through.

Figure 58: Virtual Output Queuing on PTX Series Routers



When congestion occurs because of the load on the egress output queue, the ingress VOQs corresponding to the egress output queue contain RED dropped packets.





**NOTE:** For more information about VOQ, see ["Understanding Virtual Output Queues on PTX Series Packet Transport Routers"](#) on page 797.

Using the following procedure, you can identify the ingress Packet Forward Engine (in terms of ingress traffic) that is contributing to the egress congestion.

To determine which ingress Packet Forwarding Engine is contributing to the RED dropped packets:

1. Determine whether there are RED dropped packets on the egress link.
  - a. Run the `show interfaces queue interface-name` command on the egress interface.

```
user@host> show interfaces queue et-7/0/0
```

- b. In the `show` output, determine whether the interface is experiencing RED dropped packets, by locating the RED-dropped packets field and checking whether its value is greater than zero.

The following example shows RED-dropped statistics for the egress Ethernet interface configured on port 0 of PIC 0, located on the FPC in slot 7.

```
user@host> show interfaces queue et-7/0/0
Physical interface: et-7/0/0, Enabled, Physical link is Up
  Interface index: 206, SNMP ifIndex: 790
Forwarding classes: 16 supported, 8 in use
Egress queues: 8 supported, 8 in use
Queue: 0, Forwarding classes: fc0
  Queued:
    Packets          :          539433200          14896082 pps
    Bytes            :          38302319880        8461137824 bps
  Transmitted:
    Packets          :          67108815          1859497 pps
    Bytes            :          4294964160        952062464 bps
    Tail-dropped packets :          0          0 pps
    RL-dropped packets  :          0          0 pps
    RL-dropped bytes    :          0          0 bps
    RED-dropped packets :          472324385        13036585 pps
    RED-dropped bytes   :          34007355720      7509075360 bps
Queue: 1, Forwarding classes: fc1
  Queued:
    Packets          :          539433555          14877096 pps
    Bytes            :          38302345472        8450201072 bps
```



## Transmitted:

|                      |   |             |                |
|----------------------|---|-------------|----------------|
| Packets              | : | 67108811    | 1859498 pps    |
| Bytes                | : | 4294963904  | 952062976 bps  |
| Tail-dropped packets | : | 0           | 0 pps          |
| RL-dropped packets   | : | 0           | 0 pps          |
| RL-dropped bytes     | : | 0           | 0 bps          |
| RED-dropped packets  | : | 472324744   | 13017598 pps   |
| RED-dropped bytes    | : | 34007381568 | 7498138096 bps |

Queue: 2, Forwarding classes: fc2

## Queued:

|         |   |             |                |
|---------|---|-------------|----------------|
| Packets | : | 539433811   | 14892745 pps   |
| Bytes   | : | 38302363728 | 8459214984 bps |

## Transmitted:

|                      |   |             |                |
|----------------------|---|-------------|----------------|
| Packets              | : | 67108833    | 1859501 pps    |
| Bytes                | : | 4294965312  | 952064512 bps  |
| Tail-dropped packets | : | 0           | 0 pps          |
| RL-dropped packets   | : | 0           | 0 pps          |
| RL-dropped bytes     | : | 0           | 0 bps          |
| RED-dropped packets  | : | 472324978   | 13033244 pps   |
| RED-dropped bytes    | : | 34007398416 | 7507150472 bps |

Queue: 3, Forwarding classes: fc3

## Queued:

|         |   |             |                |
|---------|---|-------------|----------------|
| Packets | : | 539433461   | 14879323 pps   |
| Bytes   | : | 38302338584 | 8451484208 bps |

## Transmitted:

|                      |   |             |                |
|----------------------|---|-------------|----------------|
| Packets              | : | 67108826    | 1859498 pps    |
| Bytes                | : | 4294964864  | 952062976 bps  |
| Tail-dropped packets | : | 0           | 0 pps          |
| RL-dropped packets   | : | 0           | 0 pps          |
| RL-dropped bytes     | : | 0           | 0 bps          |
| RED-dropped packets  | : | 472324635   | 13019825 pps   |
| RED-dropped bytes    | : | 34007373720 | 7499421232 bps |

Queue: 4, Forwarding classes: fc4

## Queued:

|         |   |             |                |
|---------|---|-------------|----------------|
| Packets | : | 539433755   | 14884190 pps   |
| Bytes   | : | 38302359616 | 8454286816 bps |

## Transmitted:

|                      |   |            |               |
|----------------------|---|------------|---------------|
| Packets              | : | 67108843   | 1859508 pps   |
| Bytes                | : | 4294965952 | 952068096 bps |
| Tail-dropped packets | : | 0          | 0 pps         |
| RL-dropped packets   | : | 0          | 0 pps         |
| RL-dropped bytes     | : | 0          | 0 bps         |
| RED-dropped packets  | : | 472324912  | 13024682 pps  |



```

    RED-dropped bytes      :          34007393664          7502218720 bps
Queue: 5, Forwarding classes: fc5
  Queued:
    Packets                :          539433849          14892950 pps
    Bytes                  :          38302366384          8459333176 bps
  Transmitted:
    Packets                :          67108843          1859497 pps
    Bytes                  :          4294965952          952062464 bps
    Tail-dropped packets :              0              0 pps
    RL-dropped packets   :              0              0 pps
    RL-dropped bytes     :              0              0 bps
    RED-dropped packets  :          472325006          13033453 pps
    RED-dropped bytes    :          34007400432          7507270712 bps
Queue: 6, Forwarding classes: fc6
  Queued:
    Packets                :          539434160          14879808 pps
    Bytes                  :          38302388632          8451762856 bps
  Transmitted:
    Packets                :          67108861          1859514 pps
    Bytes                  :          4294967104          952071168 bps
    Tail-dropped packets :              0              0 pps
    RL-dropped packets   :              0              0 pps
    RL-dropped bytes     :              0              0 bps
    RED-dropped packets  :          472325299          13020294 pps
    RED-dropped bytes    :          34007421528          7499691688 bps
Queue: 7, Forwarding classes: fc7
  Queued:
    Packets                :          539434364          14900946 pps
    Bytes                  :          38302403328          8463940000 bps
  Transmitted:
    Packets                :          67108860          1859496 pps
    Bytes                  :          4294967040          952061952 bps
    Tail-dropped packets :              0              0 pps
    RL-dropped packets   :              0              0 pps
    RL-dropped bytes     :              0              0 bps
    RED-dropped packets  :          472325504          13041450 pps
    RED-dropped bytes    :          34007436288          7511878048 bps

```



2. If the interface is experiencing RED dropped packets, run the `show interface voq interface-name` command on the egress interface that is experiencing the RED dropped packets.

```
user@host> show interfaces voq et-7/0/0 non-zero
```



**TIP:** When using the `show interfaces voq` command, you can use command filters to help locate the exact queue. For command usage, see *show interfaces voq*.

3. In the `show` output, determine whether the interface is experiencing RED dropped packets. The following example shows the count of the ingress RED-dropped packets for the egress Ethernet interface configured on port 0 of PIC 0, located on the FPC in slot 7.

The sample output shows that the cause of the congestion is the ingress Packet Forwarding Engine PFE 0, on FPC number 4, and the ingress Packet Forwarding Engine PFE 0 on FPC number 6, as denoted by the count of RED-dropped packets.

```
user@host> show interfaces voq et-7/0/0 non-zero
Physical interface: et-7/0/0, Enabled, Physical link is Up
Interface index: 156, SNMP ifIndex: 699

Queue: 0, Forwarding classes: q00

FPC number: 4
PFE: 0
  RED-dropped packets :          6834995          96929 pps
  RED-dropped bytes   :          5249276160        595537368 bps

FPC number: 6
PFE: 0
  RED-dropped packets :          6835203          96964 pps
  RED-dropped bytes   :          5249435904        595749256 bps

Queue: 1, Forwarding classes: q01

FPC number: 4
PFE: 0
  RED-dropped packets :          6834998          96967 pps
  RED-dropped bytes   :          5249278464        595766280 bps

FPC number: 6
```



PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6835201    | 96627 pps     |
| RED-dropped bytes :   | 5249434368 | 593677664 bps |

Queue: 2, Forwarding classes: q02

FPC number: 4

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6834997    | 96921 pps     |
| RED-dropped bytes :   | 5249277696 | 595482712 bps |

FPC number: 6

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6835205    | 96827 pps     |
| RED-dropped bytes :   | 5249437440 | 594907344 bps |

Queue: 3, Forwarding classes: q03

FPC number: 4

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6834997    | 96961 pps     |
| RED-dropped bytes :   | 5249277696 | 595731736 bps |

FPC number: 6

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6835202    | 96522 pps     |
| RED-dropped bytes :   | 5249435136 | 593031808 bps |

Queue: 4, Forwarding classes: q04

FPC number: 4

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6834995    | 97021 pps     |
| RED-dropped bytes :   | 5249276160 | 596099296 bps |

FPC number: 6

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6835199    | 96935 pps     |
| RED-dropped bytes :   | 5249432832 | 595572304 bps |

Queue: 5, Forwarding classes: q05

FPC number: 4



PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6834996    | 96949 pps     |
| RED-dropped bytes :   | 5249276928 | 595656872 bps |

FPC number: 6

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6835204    | 96899 pps     |
| RED-dropped bytes :   | 5249436672 | 595348960 bps |

Queue: 6, Forwarding classes: q06

FPC number: 4

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6835000    | 97019 pps     |
| RED-dropped bytes :   | 5249280000 | 596088832 bps |

FPC number: 6

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6835201    | 96916 pps     |
| RED-dropped bytes :   | 5249434368 | 595455624 bps |

Queue: 7, Forwarding classes: q07

FPC number: 4

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6834999    | 96929 pps     |
| RED-dropped bytes :   | 5249279232 | 595536704 bps |

FPC number: 6

PFE: 0

|                       |            |               |
|-----------------------|------------|---------------|
| RED-dropped packets : | 6835202    | 96941 pps     |
| RED-dropped bytes :   | 5249435136 | 595609968 bps |



**NOTE:** For an aggregate interface, follow the same steps, but you must run the `show interface queue` command on each child link of the aggregate interface to determine which child egress link is experiencing the congestion. Then run the `show interface voq` command on that child link to determine which ingress Packet Forward Engine is contributing to the congestion.



## RELATED DOCUMENTATION

[Virtual Output Queues on PTX Series Packet Transport Routers | 797](#)

*show interfaces voq*

*show interfaces queue*

## Configuring Queuing and Shaping on Logical Interfaces on PTX Series Routers

You can enable per-logical interface queuing (sometimes called per-unit scheduling) on on PTX Series Packet Transport Routers and specify a traffic-shaping rate for each logical interface. In conjunction, you can also configure other class-of-service (CoS) features, including classifiers, schedulers, and rewrite rules.

- On PTX5000 Series routers:
  - Only 100-Gigabit Ethernet interfaces are supported.
  - You can configure a maximum of 10 logical interfaces on each physical interface.
  - The maximum shaping rate cannot exceed 100 Gbps for all logical interfaces configured on an interface.
  - You cannot configure per-logical interface queuing on Aggregated Ethernet interfaces.
- On PTX10000 Series routers:
  - Only 400-Gigabit Ethernet interfaces are supported.
  - You can configure a maximum of 6 logical interfaces with per-unit scheduling enabled on each interface (maximum of 2 logical interfaces per interface on PTX10016 routers).
  - The maximum shaping rate cannot exceed 100 Gbps combined for all logical interfaces configured on an interface.
- You cannot configure per-logical interface queuing if you have configured strict priority scheduling on any interface on the router.
- The interface wildcard character (\*) is not supported—for example, the following configuration is not supported and should not be used:

```
set class-of-service interface et-* unit * shaping-rate 1g
set class-of-service interface et-* unit * scheduler-map sch0
```



To configure per-logical interface queuing and traffic shaping on PTX Series routers:

1. Enable per-logical interface queuing on the interface:

```
[edit interfaces et-fpc/pic/port]
user@host# set per-unit-scheduler
```

2. Enable the reception and transmission of 8021.q VLAN-tagged frames on the interface:

```
[edit interfaces et-fpc/pic/port]
user@host# set vlan-tagging
```

3. Configure logical interface properties.

- a. Specify a VLAN identifier for each logical interface:

```
[edit interfaces et-fpc/pic/port unit logical-unit-number]
user@host# set vlan-id number
```

- b. Specify a protocol family and IP address for each logical interface:

```
[edit interfaces et-fpc/pic/port unit logical-unit-number]
user@host# set family (inet | inet6 | mpls) address ip-address
```

4. Configure per-logical interface traffic shaping by specifying the amount of bandwidth to be allocated to each logical interface:

```
[edit class-of-service interfaces et-fpc/pic/port unit logical-unit-number]
user@host# set shaping-rate rate
```



**NOTE:** If you do not configure a shaping rate, then the line rate of the physical interface is evenly distributed across all logical interfaces, with each logical interface's portion considered its shaping rate.

5. (Optional) Configure one or more classifiers and apply them to the logical interface.



- a. Define one or more behavior aggregate (BA) classifiers:

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

- b. Define one or more forwarding classes for each classifier:

```
[edit class-of-service classifiers classifier-type classifier-name]
user@host# set forwarding-class forwarding-class-name loss-priority level code-points
[ aliases ] [ bit-patterns ]
```

- c. Apply one or more classifiers to the logical interface:

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

## 6. (Optional) Configure one or more rewrite rules to set CoS bits on outgoing packets.

- a. Define one or more rewrite rules:

```
[edit class-of-service rewrite-rules]
user@host# set traffic-type rewrite-rule-name
```

- b. Define one or more forwarding classes for each rewrite rule:

```
[edit class-of-service rewrite-rules traffic-type rewrite-rule-name]
user@host# set forwarding-class forwarding-class-name loss-priority level code-points
[ aliases ] [ bit-patterns ]
```

- c. Apply one or more rewrite rules to the logical interface for outgoing traffic:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
user@host# set rewrite-rules traffic-type rewrite-rule-name
```

## 7. (Optional) Configure one or more scheduler maps and apply them to the logical interface. Use a scheduler map to associate the properties of the output queues you define in schedulers with forwarding classes.



- a. Specify the name of a scheduler map:

```
[edit class-of-service]
user@host# set scheduler-maps scheduler-map-name
```

- b. Specify the name of a forwarding class to associate with the scheduler map:

```
[edit class-of-service scheduler-maps scheduler-map-name]
user@host set forwarding-class forwarding-class-name
```

- c. Specify the name of a scheduler configured at the [edit class-of-service schedulers *scheduler-name*] hierarchy level to associate with the scheduler map:

```
[edit class-of-service scheduler-maps scheduler-map-name]
user@host# set schedulers scheduler-name
```

- d. Apply the scheduler map to the logical interface:

```
[edit class-of-service]
user@host# set interfaces et-fpc/pic/port unit logical-unit-number scheduler-map scheduler-map-name
```

8. (Optional) Configure one or more traffical control profiles and apply them to the logical interface. Use a traffic control profile to configure traffic shaping and scheduling profiles for the logical interface.

- a. Specify the name of a traffic control profile:

```
[edit class-of-service]
user@host# set traffic-control-profiles profile-name
```

- b. Specify a shaping rate for the traffic control profile:

```
[edit class-of-service traffic-control-profiles profile-name]
user@host set shaping-rate (percent percentage | rate)
```



- c. Specify a scheduler map for the traffic control profile:

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set scheduler-map map-name
```

- d. Apply the traffic control profile to the logical interface:

```
[edit class-of-service]
user@host# set interfaces et-fpc/pic/port unit logical-unit-number output-traffic-control-
profile profile-name
```

## RELATED DOCUMENTATION

[The Junos OS CoS Components Used to Manage Congestion and Control Service Levels | 6](#)

[Example: Configuring Queuing and Shaping on Logical Interfaces in PTX Series Packet Transport Routers | 833](#)

*per-unit-scheduler*

*shaping-rate*

## Example: Configuring Queuing and Shaping on Logical Interfaces in PTX Series Packet Transport Routers

### IN THIS SECTION

- [Requirements | 834](#)
- [Overview | 834](#)
- [Configuration | 834](#)

You can enable per-logical interface queuing on interfaces on PTX Series Packet Transport Routers and specify a scheduler map and traffic-shaping rate for each logical interface.



## Requirements

This example uses the following hardware and software components:

- Junos OS Evolved Release 24.2R1 or later.
- One PTX10004 router with LC1201 or LC1202 line card.

## Overview

This example shows how to configure six VLANs on six logical interfaces, enable per-logical interface queuing, and specify a scheduler map and traffic-shaping rate for each logical interface. The total traffic-shaping rate combined for all logical interfaces cannot exceed 100 Gbps.

## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 834](#)
- [Procedure | 836](#)
- [Results | 839](#)

### CLI Quick Configuration

To configure logical interface queuing and traffic shaping on the PTX10004 router, copy the following commands and paste them into the terminal window of the router:

```
[edit]
set interfaces et-2/0/0 vlan-tagging
set interfaces et-2/0/0 per-unit-scheduler
set interfaces et-2/0/0 unit 0 vlan-id 0
set interfaces et-2/0/0 unit 1 vlan-id 1
set interfaces et-2/0/0 unit 2 vlan-id 2
set interfaces et-2/0/0 unit 3 vlan-id 3
set interfaces et-2/0/0 unit 4 vlan-id 4
set interfaces et-2/0/0 unit 5 vlan-id 5

set class-of-service classifiers dscp dscp_v4 forwarding-class GOLD loss-priority low code-
points 000000
set class-of-service classifiers dscp dscp_v4 forwarding-class SILVER loss-priority low code-
```



```

points 000001
set class-of-service classifiers dscp dscp_v4 forwarding-class BRONZE loss-priority low code-
points 000010
set class-of-service classifiers dscp dscp_v4 forwarding-class BE loss-priority low code-points
000011
set class-of-service classifiers dscp dscp_v4 forwarding-class BE1 loss-priority low code-points
000100
set class-of-service classifiers dscp dscp_v4 forwarding-class BE2 loss-priority low code-points
000101
set class-of-service classifiers dscp dscp_v4 forwarding-class BE3 loss-priority low code-points
000110
set class-of-service classifiers dscp dscp_v4 forwarding-class BE4 loss-priority low code-points
000111

set class-of-service forwarding-classes class GOLD queue-num 0
set class-of-service forwarding-classes class SILVER queue-num 1
set class-of-service forwarding-classes class BRONZE queue-num 2
set class-of-service forwarding-classes class BE queue-num 3
set class-of-service forwarding-classes class BE1 queue-num 4
set class-of-service forwarding-classes class BE2 queue-num 5
set class-of-service forwarding-classes class BE3 queue-num 6
set class-of-service forwarding-classes class BE4 queue-num 7

set class-of-service schedulers GOLD transmit-rate percent 20
set class-of-service schedulers SILVER transmit-rate percent 20
set class-of-service schedulers BRONZE transmit-rate percent 10
set class-of-service schedulers BE transmit-rate percent 10

set class-of-service schedulers GOLD priority high
set class-of-service schedulers SILVER priority medium-high
set class-of-service schedulers BRONZE priority medium-low
set class-of-service schedulers BE priority low

set class-of-service scheduler-maps OUT forwarding-class GOLD scheduler GOLD
set class-of-service scheduler-maps OUT forwarding-class SILVER scheduler SILVER
set class-of-service scheduler-maps OUT forwarding-class BRONZE scheduler BRONZE
set class-of-service scheduler-maps OUT forwarding-class BE scheduler BE
set class-of-service scheduler-maps OUT forwarding-class BE1 scheduler BE
set class-of-service scheduler-maps OUT forwarding-class BE2 scheduler BE
set class-of-service scheduler-maps OUT forwarding-class BE3 scheduler BE
set class-of-service scheduler-maps OUT forwarding-class BE4 scheduler BE

set class-of-service interfaces et-2/0/0 unit 0 scheduler-map OUT

```



```

set class-of-service interfaces et-2/0/0 unit 1 scheduler-map OUT
set class-of-service interfaces et-2/0/0 unit 2 scheduler-map OUT
set class-of-service interfaces et-2/0/0 unit 3 scheduler-map OUT
set class-of-service interfaces et-2/0/0 unit 4 scheduler-map OUT
set class-of-service interfaces et-2/0/0 unit 5 scheduler-map OUT

set class-of-service interfaces et-2/0/0 unit 0 shaping-rate 20g
set class-of-service interfaces et-2/0/0 unit 1 shaping-rate 16g
set class-of-service interfaces et-2/0/0 unit 2 shaping-rate 16g
set class-of-service interfaces et-2/0/0 unit 3 shaping-rate 16g
set class-of-service interfaces et-2/0/0 unit 4 shaping-rate 16g
set class-of-service interfaces et-2/0/0 unit 5 shaping-rate 16g

```

## Procedure

### Step-by-Step Procedure

To configure the PTX10004 router:

1. Enable the reception and transmission of 8021.q VLAN-tagged frames on the interface:

```

[edit interfaces]
user@host# set et-2/0/0 vlan-tagging

```

2. Enable per-logical interface scheduling on the interface:

```

[edit interfaces]
user@host# set et-2/0/0 per-unit-scheduling

```

3. Specify a VLAN identifier for each logical interface:

```

[edit interfaces]
user@host# set et-2/0/0 unit 0 vlan-id 0
user@host# set et-2/0/0 unit 1 vlan-id 1
user@host# set et-2/0/0 unit 2 vlan-id 2
user@host# set et-2/0/0 unit 3 vlan-id 3
user@host# set et-2/0/0 unit 4 vlan-id 4
user@host# set et-2/0/0 unit 5 vlan-id 5

```



4. Define classifiers to assign incoming packets to forwarding classes based on each packet's code point value and loss priority:

```
[edit class-of-service classifiers]
user@host# set dscp dscp_v4 forwarding-class GOLD loss-priority low code-points 000000
user@host# set dscp dscp_v4 forwarding-class SILVER loss-priority low code-points 000001
user@host# set dscp dscp_v4 forwarding-class BRONZE loss-priority low code-points 000010
user@host# set dscp dscp_v4 forwarding-class BE loss-priority low code-points 000011
user@host# set dscp dscp_v4 forwarding-class BE1 loss-priority low code-points 000100
user@host# set dscp dscp_v4 forwarding-class BE2 loss-priority low code-points 000101
user@host# set dscp dscp_v4 forwarding-class BE3 loss-priority low code-points 000110
user@host# set dscp dscp_v4 forwarding-class BE4 loss-priority low code-points 000111
```

5. Assign each forwarding class to a queue:

```
[edit class-of-service forwarding-classes]
user@host# set class GOLD queue-num 0
user@host# set class SILVER queue-num 1
user@host# set class BRONZE queue-num 2
user@host# set class BE queue-num 3
user@host# set class BE1 queue-num 4
user@host# set class BE2 queue-num 5
user@host# set class BE3 queue-num 6
user@host# set class BE4 queue-num 7
```

6. Define schedulers, each with their own transmit rate and priority:

```
[edit class-of-service schedulers]
user@host# set GOLD transmit-rate percent 20
user@host# set SILVER transmit-rate percent 20
user@host# set BRONZE transmit-rate percent 10
user@host# set BE transmit-rate percent 10

user@host# set GOLD priority high
user@host# set SILVER priority medium-high
user@host# set BRONZE priority medium-low
user@host# set BE priority low
```



7. Define a scheduler map that maps a scheduler to each forwarding class (and therefore queue):

```
[edit class-of-service scheduler-maps]
user@host# set OUT forwarding-class GOLD scheduler GOLD
user@host# set OUT forwarding-class SILVER scheduler SILVER
user@host# set OUT forwarding-class BRONZE scheduler BRONZE
user@host# set OUT forwarding-class BE scheduler BE
user@host# set OUT forwarding-class BE1 scheduler BE
user@host# set OUT forwarding-class BE2 scheduler BE
user@host# set OUT forwarding-class BE3 scheduler BE
user@host# set OUT forwarding-class BE4 scheduler BE
```

8. Assign the scheduler map to each logical interface:

```
[edit class-of-service interfaces]
user@host# set et-2/0/0 unit 0 scheduler-map OUT
user@host# set et-2/0/0 unit 1 scheduler-map OUT
user@host# set et-2/0/0 unit 2 scheduler-map OUT
user@host# set et-2/0/0 unit 3 scheduler-map OUT
user@host# set et-2/0/0 unit 4 scheduler-map OUT
user@host# set et-2/0/0 unit 5 scheduler-map OUT
```



**NOTE:** Keep in mind, with per-logical interface queueing, each logical interface has eight queues and can have its own scheduler map for its own service profile.

9. Specify the amount of bandwidth to allocate to each logical interface:

```
[edit class-of-service interfaces]
user@host# set et-2/0/0 unit 0 shaping-rate 20g
user@host# set et-2/0/0 unit 1 shaping-rate 16g
user@host# set et-2/0/0 unit 2 shaping-rate 16g
user@host# set et-2/0/0 unit 3 shaping-rate 16g
user@host# set et-2/0/0 unit 4 shaping-rate 16g
user@host# set et-2/0/0 unit 5 shaping-rate 16g
```



## Results

Confirm your results by entering the `show interfaces` and `show class-of-service` commands:

```
user@host# show interfaces
```

```
et-2/0/0 {  
    per-unit-scheduling  
    vlan-tagging;  
    unit 0 {  
        vlan-id 0;  
    }  
    unit 1 {  
        vlan-id 1;  
    }  
    unit 2 {  
        vlan-id 2;  
    }  
    unit 3 {  
        vlan-id 3;  
    }  
    unit 4 {  
        vlan-id 4;  
    }  
    unit 5 {  
        vlan-id 5;  
    }  
}
```

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

```
classifiers {  
    dscp dscp_v4 {  
        forwarding-class BE {  
            loss-priority low code-points 000011;  
        }  
        forwarding-class BE1 {  
            loss-priority low code-points 000100;  
        }  
        forwarding-class BE2 {  
            loss-priority low code-points 000101;  
        }  
        forwarding-class BE3 {
```



```

        loss-priority low code-points 000110;
    }
    forwarding-class BE4 {
        loss-priority low code-points 000111;
    }
    forwarding-class BRONZE {
        loss-priority low code-points 000010;
    }
    forwarding-class GOLD {
        loss-priority low code-points 000000;
    }
    forwarding-class SILVER {
        loss-priority low code-points 000001;
    }
}
}
forwarding-classes {
    class BE queue-num 3;
    class BE1 queue-num 4;
    class BE2 queue-num 5;
    class BE3 queue-num 6;
    class BE4 queue-num 7;
    class BRONZE queue-num 2;
    class GOLD queue-num 0;
    class SILVER queue-num 1;
}interfaces {
    et-2/0/0 {
        unit 0 {
            scheduler-map OUT;
            shaping-rate 20g;
        }
        unit 1 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
        unit 2 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
        unit 3 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
    }
}

```



```

        unit 4 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
        unit 5 {
            scheduler-map OUT;
            shaping-rate 16g;
        }
    }
}

scheduler-maps {
    OUT {
        forwarding-class BE scheduler BE;
        forwarding-class BE1 scheduler BE;
        forwarding-class BE2 scheduler BE;
        forwarding-class BE3 scheduler BE;
        forwarding-class BE4 scheduler BE;
        forwarding-class BRONZE scheduler BRONZE;
        forwarding-class GOLD scheduler GOLD;
        forwarding-class SILVER scheduler SILVER;
    }
}

schedulers {
    BE {
        transmit-rate percent 10;
        priority low;
    }
    BRONZE {
        transmit-rate percent 10;
        priority medium-low;
    }
    GOLD {
        transmit-rate percent 20;
        priority high;
    }
    SILVER {
        transmit-rate percent 20;
        priority medium-high;
    }
}

```



## RELATED DOCUMENTATION

[Configuring Queuing and Shaping on Logical Interfaces on PTX Series Routers | 829](#)

*per-unit-scheduler*

*shaping-rate*

## Example: Configuring Strict-Priority Scheduling on a PTX Series Router

### IN THIS SECTION

- [Requirements | 842](#)
- [Overview | 842](#)
- [Configuration | 844](#)
- [Verification | 850](#)

This example shows how to configure *strict-priority scheduling* for a physical interface on a PTX Series router.

### Requirements

This example uses the following hardware and software components:

- One PTX Series Packet Transport Router
- One or more routers that provide input packets and receive output packets
- Junos OS Release 13.3 or later

### Overview

#### IN THIS SECTION

- [Topology | 843](#)



This example illustrates how you configure strict-priority scheduling for a physical interface on a PTX Series router to perform processing of queues in strict-priority order. Queues in the guaranteed region with the same priority are processed in round-robin fashion. Queues in the excess region are processed based on the WRR algorithm.

When you configure strict-priority scheduling, use the following guidelines:

- The configured `transmit-rate` does not affect the queue drain rate because packets are processed in order of queue priority.
- You can configure only one queue with `strict-high` priority at the `[edit class-of-service schedulers scheduler-name priority]` hierarchy level.
- You cannot configure both `transmit-rate exact` and `strict-high` priority at the `[edit class-of-service schedulers scheduler-name]` hierarchy level.
- You cannot configure `scheduler-map` or `shaping-rate` on an interface where you configure an output traffic control profile.
- You cannot configure `transmit-rate` on a queue with `low` priority or the commit will fail.



**NOTE:** If a strict-high priority queue is constantly loaded to 100 percent of traffic capacity, other queues are starved. Queue starvation can cause the interface hardware to generate critical interrupts.

## Topology

In [Figure 59 on page 843](#), the PTX Series router has inputs from Router A, `et-1/1/15` and `et-1/1/12`, and an output to Router B, `et-7/1/12`. This example configures classification on the two ingress interfaces and configures strict-priority scheduling on the egress interface.

**Figure 59: Topology for Configuring Strict-Priority Scheduling on a PTX Series Router**





## Configuration

### IN THIS SECTION

- [Configuring Strict-Priority Scheduling | 844](#)

## Configuring Strict-Priority Scheduling

### CLI Quick Configuration

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

```
set class-of-service classifiers dscp cls forwarding-class be loss-priority high code-points
000000
set class-of-service classifiers dscp cls forwarding-class ef loss-priority medium-high code-
points 000001
set class-of-service classifiers dscp cls forwarding-class af loss-priority medium-low code-
points 000010
set class-of-service classifiers dscp cls forwarding-class nc loss-priority low code-points
000011
set class-of-service classifiers dscp cls forwarding-class af11 loss-priority low code-points
000100
set class-of-service classifiers dscp cls forwarding-class af12 loss-priority low code-points
000101
set class-of-service classifiers dscp cls forwarding-class af13 loss-priority low code-points
000110
set class-of-service classifiers dscp cls forwarding-class nc2 loss-priority low code-points
000111
set class-of-service forwarding-classes queue 0 be
set class-of-service forwarding-classes queue 1 ef
set class-of-service forwarding-classes queue 2 af
set class-of-service forwarding-classes queue 3 nc
set class-of-service forwarding-classes queue 4 af11
set class-of-service forwarding-classes queue 5 af12
set class-of-service forwarding-classes queue 6 af13
set class-of-service forwarding-classes queue 7 nc2
set class-of-service traffic-control-profiles tcp1 scheduler-map sch0
```



```

set class-of-service traffic-control-profiles tcp1 strict-priority-scheduler
set class-of-service interfaces et-1/1/12 unit 0 classifiers dscp cls
set class-of-service interfaces et-1/1/15 unit 0 classifiers dscp cls
set class-of-service interfaces et-7/1/12 output-traffic-control-profile tcp1
set class-of-service scheduler-maps sch0 forwarding-class be scheduler be_sch
set class-of-service scheduler-maps sch0 forwarding-class ef scheduler ef_sch
set class-of-service scheduler-maps sch0 forwarding-class af scheduler af_sch
set class-of-service scheduler-maps sch0 forwarding-class nc scheduler nc_sch
set class-of-service scheduler-maps sch0 forwarding-class af11 scheduler af11_sch
set class-of-service scheduler-maps sch0 forwarding-class af12 scheduler af12_sch
set class-of-service scheduler-maps sch0 forwarding-class af13 scheduler af13_sch
set class-of-service scheduler-maps sch0 forwarding-class nc2 scheduler nc2_sch
set class-of-service schedulers be_sch transmit-rate percent 60
set class-of-service schedulers be_sch priority high
set class-of-service schedulers ef_sch transmit-rate percent 5
set class-of-service schedulers ef_sch priority medium-high
set class-of-service schedulers af_sch transmit-rate percent 5
set class-of-service schedulers af_sch priority high
set class-of-service schedulers nc_sch transmit-rate percent 5
set class-of-service schedulers nc_sch priority strict-high
set class-of-service schedulers af11_sch transmit-rate percent 5
set class-of-service schedulers af11_sch priority high
set class-of-service schedulers af12_sch transmit-rate percent 5
set class-of-service schedulers af12_sch priority medium-high
set class-of-service schedulers af13_sch transmit-rate percent 5
set class-of-service schedulers af13_sch priority medium-low
set class-of-service schedulers nc2_sch priority low

```

## Step-by-Step Procedure

In this example, eight schedulers are configured based on eight DSCP classifier configurations. Each associated scheduler is assigned a priority and transmit rate, although the transmit rate is ignored by the strict-priority scheduler. The scheduler map sch0 is configured with the mapping of forwarding classes to schedulers. Within the traffic control profile tcp1, the scheduler map and the strict-priority scheduler feature are configured. Two input interfaces on the PTX Series router, et-1/1/12 and et-1/1/15, are configured with the DSCP classifiers. The output traffic control profile on et-7/1/12 is configured with the traffic control profile tcp1.

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

To configure strict-priority scheduling:



### 1. Configure the DSCP forwarding classes.

```
[edit class-of-service dscp cls]
user@host# set forwarding-class be loss-priority high code-points 000000
user@host# set forwarding-class ef loss-priority medium-high code-points 000001
user@host# set forwarding-class af loss-priority medium-low code-points 000010
user@host# set forwarding-class nc loss-priority low code-points 000011
user@host# set forwarding-class af11 loss-priority low code-points 000100
user@host# set forwarding-class af12 loss-priority low code-points 000101
user@host# set forwarding-class af13 loss-priority low code-points 000110
user@host# set forwarding-class nc2 loss-priority low code-points 000111
```

### 2. Configure the mapping of queues to forwarding classes.

```
[edit class-of-service forwarding-classes]
user@host# set queue 0 be
user@host# set queue 1 ef
user@host# set queue 2 af
user@host# set queue 3 nc
user@host# set queue 4 af11
user@host# set queue 5 af12
user@host# set queue 6 af13
user@host# set queue 7 nc2
```

### 3. Configure the transmit rate and priority for each scheduler.

Although you can configure a transmit rate, the value that you configure is overridden by the strict-priority scheduler.

```
[edit class-of-service]
user@host# set schedulers be_sch transmit-rate percent 60
user@host# set schedulers be_sch priority high
user@host# set schedulers ef_sch transmit-rate percent 5
user@host# set schedulers ef_sch priority medium-high
user@host# set schedulers af_sch transmit-rate percent 5
user@host# set schedulers af_sch priority high
user@host# set schedulers nc_sch transmit-rate percent 5
user@host# set schedulers nc_sch priority strict-high
user@host# set schedulers af11_sch transmit-rate percent 5
user@host# set schedulers af11_sch priority high
```



```

user@host# set schedulers af12_sch transmit-rate percent 5
user@host# set schedulers af12_sch priority medium-high
user@host# set schedulers af13_sch transmit-rate percent 5
user@host# set schedulers af13_sch priority medium-low
user@host# set schedulers nc2_sch priority low

```

4. Configure the scheduler map with the mapping of forwarding classes to schedulers.

```

[edit class-of-service scheduler-maps sch0]
user@host# set forwarding-class be scheduler be_sch
user@host# set forwarding-class ef scheduler ef_sch
user@host# set forwarding-class af scheduler af_sch
user@host# set forwarding-class nc scheduler nc_sch
user@host# set forwarding-class af11 scheduler af11_sch
user@host# set forwarding-class af12 scheduler af12_sch
user@host# set forwarding-class af13 scheduler af13_sch
user@host# set forwarding-class nc2 scheduler nc2_sch

```

5. Configure the traffic control profile to do strict-priority scheduling and define the scheduler map to use.

```

[edit class-of-service traffic-control-profiles tcp1]
user@host# set scheduler-map sch0
user@host# set strict-priority-scheduler

```

6. Apply the classifiers to the input interfaces, and the traffic control profile to the output interface.

```

[edit class-of-service interfaces]
user@host# set et-1/1/12 unit 0 classifiers dscp cls
user@host# set et-1/1/15 unit 0 classifiers dscp cls
user@host# set et-7/1/12 output-traffic-control-profile tcp1

```



## Results

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

```
[edit]
user@host# show class-of-service
forwarding-classes {
    queue 0 be;
    queue 1 ef;
    queue 2 af;
    queue 3 nc;
    queue 4 af11;
    queue 5 af12;
    queue 6 af13;
    queue 7 nc2;
}
interfaces {
    et-1/1/12 {
        unit 0 {
            classifiers {
                dscp cls;
            }
        }
    }
    et-1/1/15 {
        unit 0 {
            classifiers {
                dscp cls;
            }
        }
    }
    et-7/1/12 {
        output-traffic-control-profile tcp1;
    }
}
scheduler-maps {
    sch0 {
        forwarding-class be scheduler be_sch;
        forwarding-class ef scheduler ef_sch;
        forwarding-class af scheduler af_sch;
```



```

        forwarding-class nc scheduler nc_sch;
        forwarding-class af11 scheduler af11_sch;
        forwarding-class af12 scheduler af12_sch;
        forwarding-class af13 scheduler af13_sch;
        forwarding-class nc2 scheduler nc2_sch;
    }
}
schedulers {
    be_sch {
        transmit-rate percent 60;
        priority high;
    }
    ef_sch {
        transmit-rate percent 5;
        priority medium-high;
    }
    af_sch {
        transmit-rate percent 5;
        priority high;
    }
    nc_sch {
        transmit-rate percent 5;
        priority strict-high;
    }
    af11_sch {
        transmit-rate percent 5;
        priority high;
    }
    af12_sch {
        transmit-rate percent 5;
        priority medium-high;
    }
    af13_sch {
        transmit-rate percent 5;
        priority medium-low;
    }
    nc2_sch {
        priority low;
    }
}
traffic-control-profiles {
    tcp1 {
        scheduler-map sch0;
    }
}

```



```

        strict-priority-scheduler;
    }
}

```

## Verification

### IN THIS SECTION

- [Verifying Strict-Priority Scheduling | 850](#)

## Verifying Strict-Priority Scheduling

### Purpose

Verify that the strict-priority scheduling configuration is producing the results you expect.

### Action

From operational mode, enter the `show interfaces queue interface-name interface-name` command and select the output physical interface to verify.

```

user@host> show interfaces queue interface-name et-7/1/12

```

Physical interface: et-7/1/12, Enabled, Physical link is Up

Interface index: 231, SNMP ifIndex: 612

Forwarding classes: 16 supported, 8 in use

Egress queues: 8 supported, 8 in use

Queue: 0, Forwarding classes: be

Queued:

|         |   |           |                |
|---------|---|-----------|----------------|
| Packets | : | 394488    | 131507 pps     |
| Bytes   | : | 591732000 | 1578084848 bps |

Transmitted:

|                      |   |           |                |
|----------------------|---|-----------|----------------|
| Packets              | : | 394488    | 131507 pps     |
| Bytes                | : | 591732000 | 1578084848 bps |
| Tail-dropped packets | : | 0         | 0 pps          |
| RL-dropped packets   | : | 0         | 0 pps          |
| RL-dropped bytes     | : | 0         | 0 bps          |
| RED-dropped packets  | : | 0         | 0 pps          |



```

    RED-dropped bytes      :                0                0 bps
Queue: 1, Forwarding classes: ef
  Queued:
    Packets                :            234498            82115 pps
    Bytes                  :        352963584        988886784 bps
  Transmitted:
    Packets                :            82425            27551 pps
    Bytes                  :        123637500        330618176 bps
    Tail-dropped packets :                0                0 pps
    RL-dropped packets    :                0                0 pps
    RL-dropped bytes      :                0                0 bps
    RED-dropped packets   :            152073            54564 pps
    RED-dropped bytes     :        229326084        658268608 bps
Queue: 2, Forwarding classes: af
  Queued:
    Packets                :            345175            115068 pps
    Bytes                  :        517762500        1380824240 bps
  Transmitted:
    Packets                :            345175            115068 pps
    Bytes                  :        517762500        1380824240 bps
    Tail-dropped packets :                0                0 pps
    RL-dropped packets    :                0                0 pps
    RL-dropped bytes      :                0                0 bps
    RED-dropped packets   :                0                0 pps
    RED-dropped bytes     :                0                0 bps
Queue: 3, Forwarding classes: nc
  Queued:
    Packets                :            986224            328769 pps
    Bytes                  :        1479336000        3945236360 bps
  Transmitted:
    Packets                :            986224            328769 pps
    Bytes                  :        1479336000        3945236360 bps
    Tail-dropped packets :                0                0 pps
    RL-dropped packets    :                0                0 pps
    RL-dropped bytes      :                0                0 bps
    RED-dropped packets   :                0                0 pps
    RED-dropped bytes     :                0                0 bps
Queue: 4, Forwarding classes: af11
  Queued:
    Packets                :            493110            164383 pps
    Bytes                  :        739665000        1972606056 bps
  Transmitted:
    Packets                :            493110            164383 pps

```



|                      |   |           |                |
|----------------------|---|-----------|----------------|
| Bytes                | : | 739665000 | 1972606056 bps |
| Tail-dropped packets | : | 0         | 0 pps          |
| RL-dropped packets   | : | 0         | 0 pps          |
| RL-dropped bytes     | : | 0         | 0 bps          |
| RED-dropped packets  | : | 0         | 0 pps          |
| RED-dropped bytes    | : | 0         | 0 bps          |

Queue: 5, Forwarding classes: af12

Queued:

|         |   |           |                |
|---------|---|-----------|----------------|
| Packets | : | 461830    | 164375 pps     |
| Bytes   | : | 695777416 | 1981272728 bps |

Transmitted:

|                      |   |           |                |
|----------------------|---|-----------|----------------|
| Packets              | : | 82778     | 27543 pps      |
| Bytes                | : | 124167000 | 330521208 bps  |
| Tail-dropped packets | : | 0         | 0 pps          |
| RL-dropped packets   | : | 0         | 0 pps          |
| RL-dropped bytes     | : | 0         | 0 bps          |
| RED-dropped packets  | : | 379052    | 136832 pps     |
| RED-dropped bytes    | : | 571610416 | 1650751520 bps |

Queue: 6, Forwarding classes: af13

Queued:

|         |   |           |                |
|---------|---|-----------|----------------|
| Packets | : | 462258    | 164556 pps     |
| Bytes   | : | 696421280 | 1983445256 bps |

Transmitted:

|                      |   |           |                |
|----------------------|---|-----------|----------------|
| Packets              | : | 82973     | 27637 pps      |
| Bytes                | : | 124459500 | 331648480 bps  |
| Tail-dropped packets | : | 0         | 0 pps          |
| RL-dropped packets   | : | 0         | 0 pps          |
| RL-dropped bytes     | : | 0         | 0 bps          |
| RED-dropped packets  | : | 379285    | 136919 pps     |
| RED-dropped bytes    | : | 571961780 | 1651796776 bps |

Queue: 7, Forwarding classes: nc2

Queued:

|         |   |           |               |
|---------|---|-----------|---------------|
| Packets | : | 227750    | 82215 pps     |
| Bytes   | : | 343447000 | 991843712 bps |

Transmitted:

|                      |   |           |               |
|----------------------|---|-----------|---------------|
| Packets              | : | 0         | 0 pps         |
| Bytes                | : | 0         | 0 bps         |
| Tail-dropped packets | : | 0         | 0 pps         |
| RL-dropped packets   | : | 0         | 0 pps         |
| RL-dropped bytes     | : | 0         | 0 bps         |
| RED-dropped packets  | : | 227750    | 82215 pps     |
| RED-dropped bytes    | : | 343447000 | 991843712 bps |



## Meaning

The `show` command output lists the traffic by queue and forwarding class names. The Bytes field under the Transmitted field for each queue shows the actual bytes transmitted.

From the sample output, you can see that the strict-high queue gets the highest priority and transmits without drops. The high-priority queues are then transmitted. The medium-high and medium-low priority queues are processed in a round-robin fashion. The low-priority queue is starved.

Keep in mind the following conditions that apply to strict-priority scheduling:

- If the traffic on the output interface is undersubscribed, no queue should show dropped traffic.
- The strict-high queue is processed first, followed by the high-priority queues (in a round-robin fashion), and finally all remaining queues in the guaranteed region (in a round-robin fashion).
- If the ingress traffic exceeds the capacity of the output interface, the queues are processed in strict-priority order.
- Queues in the excess region are processed based on the WRR algorithm.

## RELATED DOCUMENTATION

[Understanding Scheduling on PTX Series Routers | 793](#)

[Example: Configuring Excess Rate for PTX Series Packet Transport Routers | 813](#)

[Understanding CoS CLI Configuration Statements on PTX Series Routers | 862](#)

[How Schedulers Define Output Queue Properties | 332](#)

[Configuring a Scheduler | 670](#)

*excess-rate*

*strict-priority-scheduler*

*traffic-control-profiles*

## CoS Support on EVPN VXLANs

### IN THIS SECTION

- [Understanding CoS on VXLAN Interfaces | 854](#)



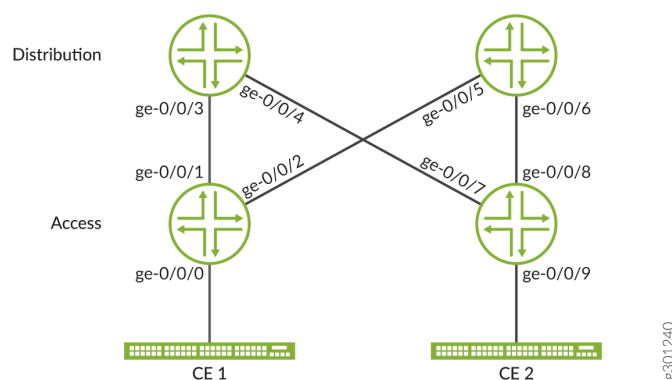
- [Configuring CoS on VXLAN Interfaces | 855](#)
- [Implementing CoS on VXLAN Interfaces \(Junos OS Evolved\) | 858](#)
- [Platform-Specific CoS on EVPN-VxLANs Behavior | 859](#)

You can configure class of service (CoS) features on VXLAN interfaces. VXLAN traffic from different tenants traverses network boundaries over the same physical underlay network. To ensure fairness in the treatment of traffic for all tenants in the VXLAN, and to prioritize higher priority traffic, apply CoS features to the VXLAN interfaces.

## Understanding CoS on VXLAN Interfaces

This section describes how classification and rewrite rules are applied to packets in a VXLAN instance. [Figure 60 on page 854](#) shows a simple VXLAN with two leaf nodes and one spine node.

**Figure 60: Classifiers and Rewrite Rules on VXLANs**



Refer to [Figure 60 on page 854](#) to understand the packet flow with DSCP/ToS fields in a VXLAN:

1. CE 1 sends a packet with Layer3 DSCP/ToS bit programmed to the Leaf 1 node.
2. Leaf 1 receives the original packet and appends the VXLAN header on top of the original packet. The outer VXLAN Layer3 header uses the original packet DSCP/Tos bit. You can create classifiers based on the original packet DSCP/802.1p bit. The ingress interface on the ingress leaf supports DSCP and 802.1p classifiers.



3. If rewrite is configured on Leaf 1, the inner header will have the DSCP/802.1p bit set by CE 1 and the outer header will have the rewrite bit. Only DSCP rewrite rules are supported, except on QFX10000 switches where 802.1p rewrite is also supported if the underlay is tagged.
4. The Spine node receives the VXLAN packet and can use ingress classification using these DSCP bits and forward the packet to the egress interface with the appropriate forwarding class.
5. The Spine egress interface can rewrite these bits using rewrite rules. These Spine rewrite rules only affects the outer Layer3 DSCP field. The inner/original packet still holds the DSCP/802.1p bit that was set by CE 1.
6. Leaf 2 receives the packet, processes the tunnel termination, and remove the outer VXLAN header.
7. Leaf 2 classification and rewrite functionality works on the inner header.
8. The original packet arrives on CE 2.



**NOTE:** On the leaf nodes, if the packet is multicast, you can use multi-destination classification to create appropriate multicast classification and rewrite rules.

## Configuring CoS on VXLAN Interfaces

This section shows sample configurations of classifiers and rewrite rules for the leaf and spine nodes in VXLAN using [Figure 60 on page 854](#) as a reference. You can create schedulers as normal for the classifiers on each node.

Sample configuration of classifiers and rewrite rules on Leaf 1.

1. Create a classifier based on the *original*/DSCP/ToS bits:

```
[edit class-of-service classifiers]
user@leaf1#set dscp dscp_cf forwarding-class best-effort loss-priority low code-points 100000
user@leaf1#set dscp dscp_cf forwarding-class network-control loss-priority high code-points 110000
user@leaf1#set dscp dscp_cf forwarding-class expedited-forwarding loss-priority low code-points 011010
user@leaf1#set dscp dscp_cf forwarding-class assured-forwarding loss-priority high code-points 001010
```

2. Apply the classier to the ingress interface:

```
[edit class-of-service interfaces]
user@leaf1#set ge-0/0/0 unit 0 classifiers dscp dscp_cf
```



3. Create a rewrite rule for the *outer* VXLAN DSCP/ToS bits:

```
[edit class-of-service rewrite-rules]
user@leaf1#set dscp dscp_rw forwarding-class best-effort loss-priority low code-points af22
user@leaf1#set dscp dscp_rw forwarding-class network-control loss-priority high code-points af31
user@leaf1#set dscp dscp_rw forwarding-class expedited-forwarding loss-priority low code-points af13
user@leaf1#set dscp dscp_rw forwarding-class assured-forwarding loss-priority high code-points cs3
```

4. Apply the rewrite rule to the egress Leaf 1 interfaces:

```
[edit class-of-service interfaces]
user@leaf1#set ge-0/0/1 unit 0 rewrite-rules dscp dscp_rw
user@leaf1#set ge-0/0/2 unit 0 rewrite-rules dscp dscp_rw
```

Sample configuration of classifiers and rewrite rules on the Spine.

1. Create a classifier based on the outer VXLAN DSCP/ToS bits:

```
[edit class-of-service classifiers]
user@spine#set dscp dscp_cf forwarding-class best-effort loss-priority low code-points af22
user@spine#set dscp dscp_cf forwarding-class network-control loss-priority high code-points af31
user@spine#set dscp dscp_cf forwarding-class expedited-forwarding loss-priority low code-points af13
user@spine#set dscp dscp_cf forwarding-class assured-forwarding loss-priority high code-points cs3
```

2. Apply the classifier to the ingress Spine interfaces:

```
[edit class-of-service interfaces]
user@spine#set ge-0/0/3 unit 0 classifiers dscp dscp_cf
user@spine#set ge-0/0/5 unit 0 classifiers dscp dscp_cf
```



### 3. Create a rewrite rule for the outer VXLAN DSCP/ToS bits:

```
[edit class-of-service rewrite-rules]
user@spine#set dscp dscp_rw forwarding-class best-effort loss-priority low code-points af22
user@spine#set dscp dscp_rw forwarding-class network-control loss-priority high code-points af31
user@spine#set dscp dscp_rw forwarding-class expedited-forwarding loss-priority low code-points af13
user@spine#set dscp dscp_rw forwarding-class assured-forwarding loss-priority high code-points cs3
```

### 4. Apply the rewrite rule to the egress Spine interfaces:

```
[edit class-of-service interfaces]
user@spine#set ge-0/0/4 unit 0 rewrite-rules dscp dscp_rw
user@spine#set ge-0/0/6 unit 0 rewrite-rules dscp dscp_rw
```

Sample configuration of classifiers and rewrite rules on Leaf 2.

### 1. Create a classifier based on the *original* DSCP/ToS bits, as the VXLAN header is removed at tunnel termination *before* forwarding classes are applied:

```
[edit class-of-service classifiers]
user@leaf2#set dscp dscp_cf forwarding-class best-effort loss-priority low code-points 100000
user@leaf2#set dscp dscp_cf forwarding-class network-control loss-priority high code-points 110000
user@leaf2#set dscp dscp_cf forwarding-class expedited-forwarding loss-priority low code-points 011010
user@leaf2#set dscp dscp_cf forwarding-class assured-forwarding loss-priority high code-points 001010
```

### 2. Apply the classifier to the ingress Leaf 2 interfaces:

```
[edit class-of-service interfaces]
user@leaf2#set ge-0/0/7 unit 0 classifiers dscp dscp_cf
user@leaf2#set ge-0/0/8 unit 0 classifiers dscp dscp_cf
```



### 3. Create a rewrite rule for the *original*/DSCP/ToS bits:

```
[edit class-of-service rewrite-rules]
user@leaf2#set dscp dscp_rw forwarding-class best-effort loss-priority low code-points 100000
user@leaf2#set dscp dscp_rw forwarding-class network-control loss-priority high code-points 110000
user@leaf2#set dscp dscp_rw forwarding-class expedited-forwarding loss-priority low code-points 011010
user@leaf2#set dscp dscp_rw forwarding-class assured-forwarding loss-priority high code-points 001010
```

### 4. Apply the rewrite rule to the egress Leaf 2 interface:

```
[edit class-of-service interfaces]
user@leaf2#set ge-0/0/9 unit 0 rewrite-rules dscp dscp_rw
```

To check the CoS configuration on one of the interfaces:

```
user@node#show class-of-service interface interface-name
```

To check the queue statistics on one of the interfaces:

```
user@node#show interfaces queue interface-name
```

## Implementing CoS on VXLAN Interfaces (Junos OS Evolved)

CoS for EVPN VXLAN traffic is supported using a combination of classifiers, schedulers, and rewrite rules. This section describes how these components are implemented across different nodes on devices running Junos OS Evolved to apply CoS on the EVPN VXLAN traffic.

- **Classification at User Network Interface (UNI)/Ingress PE** — Traffic classification based on IEEE 802.1p and Differentiated Services code point (DSCP) are supported on the ingress PE where the EVPN VXLAN tunnel is initiated. BA and MF classifiers can be applied to Enterprise style (EP) or Service Provider (SP) style access interfaces.
- **Classification at Network Node Interface (NNI)/Egress PE** — Traffic classification based on IEEE 802.1p and Differentiated Services code point (DSCP) are supported on the egress PE where the EVPN VXLAN tunnel is terminated. BA classifiers can be applied to the underlying logical interface or unit. MF classifiers are not supported in tunnel terminations.



- **Rewrite at NNI** — After the encapsulation of the VXLAN tunnel, the rewrites on the outer/tunnel header are configured using the rewrite rules on the underlying logical interface or unit. Based on the configured rewrite rules, the VXLAN traffic is classified in the Spine/Network.

DSCP rewrite on the outer/tunnel header of VXLAN packets is supported on the NNI interface.

Rewrite rules are supported in the following EVPN VXLAN scenarios:

- Intra-VNI L2 gateway — Rewrite rules are applied to both unicast and broadcast, unknown unicast, and multicast (BUM) traffic.
  - Inter-VNI L3 gateway — Centrally-routed bridging (CRB) and edge-routed bridging (ERB).
  - EVPN Type 5 routes.
- **Rewrite at UNI** — After the termination of the VXLAN tunnel, the rewrites on the inner headers are configured using rewrite rules on the Enterprise style (EP) or Service Provider (SP) style access interfaces. Based on the configured rewrite rules, the decapsulated packets are classified in the CE side network. The following rewrite rules are supported on the UNI interface for the decapsulated packets:
    - DSCP rewrites on the inner IPv4/IPv6 header
    - IEEE 802.1p rewrites on the inner Ethernet header (if tagged)

Rewrite rules are supported in the following EVPN VXLAN scenarios:

- Intra-VNI L2 gateway — Rewrite rules are applied to both unicast and broadcast, unknown unicast, and multicast (BUM) traffic.
- Inter-VNI L3 gateway — Centrally-routed bridging (CRB) and edge-routed bridging (ERB).
- EVPN Type 5 routes.
- **Scheduling** — Traffic prioritization and bandwidth reservation are achieved by using schedulers. The schedulers are associated with a forwarding class set via classifiers.

## Platform-Specific CoS on EVPN-VxLANs Behavior

Use [Feature Explorer](#) to confirm platform and release support for specific features.

Use the following table to review platform-specific behaviors for CoS on EVPN-VxLANs:



| Platform               | Difference                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PTX Series routers     | <p>The following limitations apply to PTX routers:</p> <ul style="list-style-type: none"> <li>• DSCP rewrite rules are not supported on Integrated Routing and Bridging (IRB) (L3 gateway scenarios).</li> <li>• IEEE 802.1p rewrite rules are not supported on the NNI interface.</li> <li>• Explicit congestion notification (ECN) rewrites are not supported on either UNI or NNI interfaces.</li> <li>• Priority-based flow control (PFC) is not supported.</li> <li>• No support for CoS classification and rewrite mechanism for IRB underlay.</li> </ul> <p><b>NOTE:</b> In EVPN-VXLAN networks with an IPv6 underlay, some PTX Series routers support CoS classification and explicit congestion notification (ECN) copy operations, but not PFC, DSCP rewrite, or IEEE 802.1p rewrite.</p> <p>See <a href="#">Feature Explorer</a> for platform and release support.</p> |
| PTX10002-36QDD routers | <p>PTX10002-36QDD routers that originate or terminate EVPN-VXLAN tunnels and have ECN enabled automatically copy the ECN bits from the inner header to the outer header. The router copies the ECN bits from the outer header to the inner header if the inner header has the ECT bit set. If the router experiences congestion, it sets the CE bits if the ECT bit is enabled.</p> <p>These routers support ECN copy operations in EVPN-VXLAN networks with an IPv4 underlay or an IPv6 underlay.</p>                                                                                                                                                                                                                                                                                                                                                                            |



*(Continued)*

| Platform                     | Difference                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| QFX5130 and QFX5700 switches | <p>The following limitations apply to the QFX5130 and QFX5700 platforms:</p> <ul style="list-style-type: none"> <li>• HQoS is not supported due to hardware limitations.</li> <li>• Classifier, rewrite and scheduler on IRB interface is not supported.</li> <li>• DOT1P rewrite and classifier on the NNI port is not supported.</li> <li>• DOT1P and DSCP rewrite on the UNI port is not supported.</li> <li>• DSCP rewrite on the NNI port is supported with the following conditions: <ul style="list-style-type: none"> <li>• DSCP rewrite on the NNI port is supported by default and does not work if vxlan-tos-copy-filter at [edit forwarding-options] is enabled.</li> <li>• Inner ECN bits are copied to the outer VXLAN header regardless of whether vxlan-tos-copy-filter at [edit forwarding-options] is enabled.</li> <li>• Do not enable vxlan-tos-copy-filter and vxlan-disable-copy-tos-encap at [edit forwarding-options] at the same time as this causes indeterministic behavior.</li> </ul> </li> <li>• PFC configuration will cause momentary traffic drops of up to 10ms.</li> <li>• DSCP IPV6 classifiers and rewrites are not supported. Use DSCP classifier and rewrite instead.</li> <li>• TOS copy feature does not work for Type-5 EVPN VXLANs.</li> </ul> |



*(Continued)*

| Platform                 | Difference                                                                                                                                                                                                                                                                                                                         |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| QFX10000 Series switches | <p>The following limitation applies to QFX10000 platforms:</p> <ul style="list-style-type: none"> <li>Because IRB interfaces do not support dscp rewrite rules, you can apply rewrite rules on underlying L2 interfaces. 802.1p/dscp values in a VXLAN tunneled packet are written using underlying L2 interface rules.</li> </ul> |

## Understanding CoS CLI Configuration Statements on PTX Series Routers

PTX Series Packet Transport Routers have no new Junos OS CLI configuration statements. However, some statements or statement options supported on other platforms are not supported or may not have effect on PTX Series devices. These exceptions are summarized here.

### [edit chassis] Hierarchy Level

The following statement is not applicable to PTX Series Packet Transport Routers. There are always eight queues available. However, if there is a requirement to use only four of eight queues, you can do this by configuring the forwarding class to queue mapping, as appropriate.

```
[edit chassis fpc slot-number pic pic-number],
  max-queues-per-interface (4 | 8);
```

The following CLI is not applicable to PICs supported on PTX Series Packet Transport Routers:

```
[edit chassis fpc slot-number pic pic-number],
  q-pic-large-buffer {
    [large-scale | small-scale]
  }
```



On PTX Series Packet Transport Routers, buffer occupancy is computed as weighted average. However, configuration of weight at the PIC level is not supported. The default weights are applied.

```
[edit chassis fpc slot-number pic pic-number],
red-buffer-occupancy {
    weighted-averaged [ instant-usage-weight-exponent ] weight-value;
}
```

The following CLI is not applicable to PICs supported on PTX Series Packet Transport Routers:

```
[edit chassis fpc slot-number pic pic-number],
traffic-manager {
    egress-shaping-overhead number;
    ingress-shaping-overhead number;
    mode session-shaping;
}
```

#### [edit class-of-service] Hierarchy Level

The following CLI is not applicable to PTX Series Packet Transport Routers because there are no separate fabric queues and egress queues:

```
fabric {
    scheduler-map {
        priority (high | low) scheduler scheduler-name;
    }
}
```

The following CLI does not support the priority and policing-priority options.

```
forwarding-classes {
    class queue-num queue-number priority (high | low);
    queue queue-number class-name priority (high | low) [ policing-priority (premium | normal) ];
}
```

The following statements are not supported on PTX Series Packet Transport Routers:

- `inet-precedence rewrite`
- Rewrite of both `exp` and `inet-precedence` fields for VPN and non-VPN traffic that use the `mpls-inet-both` and `mpls-inet-both-non-vpn` protocol types.



- exp-push-push-push and exp-swap-push-push rules
- input-scheduler-map and input-shaping-rate
- The physical interface scheduler is applied on the Packet Forwarding Engine, hence the scheduler-map-chassis statement is not applicable.

```

interfaces {
    interface-name {
        input-scheduler-map map-name;
        input-shaping-rate rate;
        scheduler-map-chassis map-name;
        unit logical-unit-number {
            rewrite-rules{
                inet-precedence (rewrite-name | default) protocol
                    protocol-types;
                exp (write-name | default) protocol protocol-types;
                exp-push-push-push default;
                exp-swap-push-push default;
            }
        }
    }
}

```

In the following CLI, only the `inet-precedence` statement is not supported.

```

rewrite-rules {
    (dscp | dscp-ipv6 | exp | ieee-802.1 | ieee-802.1ad | inet-precedence) rewrite-name {
        import (rewrite-name | default);
        forwarding-class class-name {
            loss-priority level code-point (alias | bits);
        }
    }
}

```

Classifiers on routing instances are not supported on PTX Series Packet Transport Routers because L3VPN is not supported. Hence, the following CLI is not applicable.

```

[edit class-of-service]
routing-instances routing-instance-name {
    classifiers {

```



```

    exp (classifier-name | default);
    dscp (classifier-name | default);
    dscp-ipv6 (classifier-name | default);
  }
}

```

The following limitations apply to statements under `schedulers` on PTX Series Packet Transport Routers:

- **protocol** (`non-tcp` | `tcp`) is not supported for **drop-profile-map**. The **any** option is supported.
- **excess-priority** is not supported.
- **rate-limit** is supported for **transmit-rate**. It is applied only when schedulers are configured as **strict-high**.

```

schedulers (CoS) {
  scheduler-name {
    buffer-size (percent percentage | remainder | temporal microseconds);
    drop-profile-map loss-priority (any | low | medium-low | medium-high
high) protocol (any ) drop-profile profile-name;
    priority priority-level;
    transmit-rate (rate | percent percentage | remainder) <exact | rate-limit>;
  }
}

```



**NOTE:** Configurations that are supported only on Gigabit Ethernet IQ PICs, channelized IQ PICs, and so forth are not applicable to PTX Series Packet Transport Routers. These PICs are not supported on this platform. Those CLIs are not listed here.

### [edit firewall] Hierarchy Level

In the following CLI, the **dscp** clause is not supported.

```

family family-name {
  filter filter-name {
    term term-name {
      from {
        match-conditions;
      }
      then {
        dscp 0;
      }
    }
  }
}

```



```

        forwarding-class class-name;
        loss-priority (high | low);
        three-color-policer {
            (single-rate | two-rate) policer-name;
        }
    }
}

```

## Classification Based on Outer Header of Decapsulation Tunnel

### SUMMARY

By default, when a packet exits a tunnel, the terminating router removes the tunnel (outer) header and forwards the packet based on the classification in the inner header. On supported routers that act as a terminating router for a tunnel, you can forward the packet based on the classification you assign to the *outer* header at the underlay (ingress) interface.

### IN THIS SECTION

- [Overview of Classification Based on Outer Header of Decapsulation Tunnel | 866](#)
- [Configuring Classification Based on Outer Header of Decapsulation Tunnel | 867](#)

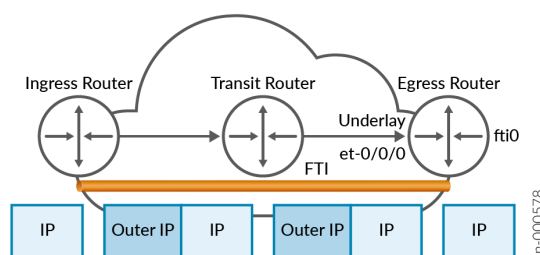
### Overview of Classification Based on Outer Header of Decapsulation Tunnel

A flexible tunnel interface (FTI) is a type of logical tunnel interface that uses static routing and BGP protocols to exchange routes over a tunnel that connects endpoints to routers. You can configure GRE, IPIP, or UDP tunnels on the logical interfaces of an FTI. For information on configuring FTIs, see *Configuring Flexible Tunnel Interfaces*.

For an IP packet entering the tunnel, the ingress router (as shown in [Figure 61 on page 867](#)) encapsulates the packet with the tunnel header (the outer IP header). This tunnel header directs the packet through the tunnel to the egress router.



Figure 61: FTI Tunnel



By default, when the packet exits the tunnel, the egress router removes the tunnel (outer) header and forwards the packet based on the classification in the inner header. You *can* forward the packet based on the classification you assign to the *outer* header at the underlay (ingress) interface of the egress router. You do this by first setting the `no-default` option at the [edit class-of-service interfaces `fti` unit 0 classifiers] hierarchy level. You then assign the desired classifier to the underlay interface of the egress router.



**NOTE:** Applying the `no-default` option disables the application of any default classifier to the routing instance. In this case, applying the `no-default` option specifically prevents classification based on the inner header, enabling classification based on the outer header.

## Configuring Classification Based on Outer Header of Decapsulation Tunnel

This section describes how to configure the classification of a packet exiting a tunnel on a terminating router to be based on the outer (underlay) header of the packet.

1. Ensure the flexible tunnel interface that terminates the tunnel is indeed configured for tunnel termination.

For example:

```
[edit interfaces]
user@host# show fti0
unit 0 {
  tunnel {
    encapsulation gre/ipip/udp {
      tunnel-termination;
      ...
    }
  }
}
```



```

    }
}

```

2. Apply the no-default option for classifiers on the flexible tunnel interface.

For example:

```

[edit class-of-service]
user@host# set interfaces fti0 unit 0 classifiers no-default
user@host# show interfaces
fti0 {
    unit 0 {
        classifiers {
            no-default;
        }
    }
}

```

3. Apply the desired classifier for packets exiting the tunnel to the underlay (ingress) interface.

For example:

```

[edit class-of-service]
user@host# set interfaces et-0/0/0 unit 0 classifiers dscp dscp1
user@host# show interfaces
et-0/0/0 {
    unit 0 {
        classifiers {
            dscp dscp1;
        }
    }
}
fti0 {
    unit 0 {
        classifiers {
            no-default;
        }
    }
}

```

4. Commit your changes.



## RELATED DOCUMENTATION

*classifiers (Interfaces)*

---

*Configuring Flexible Tunnel Interfaces*



# 4

PART

## Configuring Line Card-Specific and Interface-Specific Functionality

---

[Feature Support of Line Cards and Interfaces | 871](#)

[Configuring Class of Service for Tunnels | 876](#)

[Configuring Class of Service on Services PICs | 892](#)

[Configuring Class of Service on IQ and Enhanced IQ \(IQE\) PICs | 922](#)

[Configuring Class of Service on Ethernet IQ2 and Enhanced IQ2 PICs | 1004](#)

[Configuring Class of Service on 10-Gigabit Ethernet LAN/WAN PICs with SFP+ | 1060](#)

[Configuring Class of Service on Enhanced Queuing DPCs | 1071](#)

[Configuring Class of Service on MICs, MPCs, and MLCs | 1094](#)

[Configuring Class of Service on Aggregated, Channelized, and Gigabit Ethernet Interfaces | 1201](#)

---



# Feature Support of Line Cards and Interfaces

## IN THIS CHAPTER

- CoS Features of the Router Hardware, PIC, MIC, and MPC Interface Families | 871
- Scheduling on the Router Hardware, PIC, MIC, and MPC Interface Families | 872
- Schedulers on the Router Hardware, PIC, MIC, and MPC Families | 873
- Queuing Parameters for the Router Hardware, PIC, MIC, and MPC Interface Families | 874

## CoS Features of the Router Hardware, PIC, MIC, and MPC Interface Families

Table 72 on page 871 compares the PIC families with regard to major CoS features. Note that this table reflects the ability to perform the CoS function *at the PIC , MIC, or MPC interface level* and not on the system as a whole.

Table 72: CoS Features of the Router Hardware and Interface Families Compared

| Feature:                 | M320 and T Series | MIC and MPC Interfaces    | IQ PICs                 | IQ2 PICs                | IQ2E PICs               | Enhanced IQ PICs |
|--------------------------|-------------------|---------------------------|-------------------------|-------------------------|-------------------------|------------------|
| BA classification        | Yes               | Yes                       | –                       | –                       | –                       | Yes              |
| ToS bit rewrites         | Yes               | Yes                       | Yes, for IEEE bits only | Yes, for IEEE bits only | Yes, for IEEE bits only | –                |
| Ingress ToS bit rewrites | –                 | Yes, with firewall filter | –                       | –                       | –                       | Yes              |



**Table 72: CoS Features of the Router Hardware and Interface Families Compared (Continued)**

| Feature:              | M320 and T Series | MIC and MPC Interfaces | IQ PICs | IQ2 PICs | IQ2E PICs | Enhanced IQ PICs |
|-----------------------|-------------------|------------------------|---------|----------|-----------|------------------|
| Hierarchical policers | –                 | Yes                    | –       | –        | –         | Yes              |

## Scheduling on the Router Hardware, PIC, MIC, and MPC Interface Families

[Table 73 on page 872](#) compares the PIC, MIC, and MPC interface families with regard to scheduling abilities or features. Note that this table reflects the ability to perform the function *at the PIC, MIC, or MPC interface level* and not necessarily on the system as a whole.

In this table, the OSE PICs refer to the 10-port 10-Gigabit OSE PICs (described in some guides as the 10-Gigabit Ethernet LAN/WAN PICs with SFP+).

**Table 73: Scheduling on the Router Hardware and Interface Families Compared**

| Scheduling Feature:                    | M320 and T Series | MIC and MPC Interfaces | IQ PICs | IQ2 PICs | IQ2E PICs | OSE PICs on T Series | Enhanced IQ PICs |
|----------------------------------------|-------------------|------------------------|---------|----------|-----------|----------------------|------------------|
| Per-unit scheduling                    | –                 | Yes, for EQ MPC        | Yes     | Yes      | Yes       | –                    | Yes              |
| Physical port and logical unit shaping | –                 | Yes                    | –       | Yes      | Yes       | –                    | Yes              |



**Table 73: Scheduling on the Router Hardware and Interface Families Compared (Continued)**

| Scheduling Feature:                  | M320 and T Series | MIC and MPC Interfaces | IQ PICs | IQ2 PICs                                                 | IQ2E PICs | OSE PICs on T Series    | Enhanced IQ PICs         |
|--------------------------------------|-------------------|------------------------|---------|----------------------------------------------------------|-----------|-------------------------|--------------------------|
| Guaranteed rate or peak rate support | –                 | Yes                    | –       | Yes, supports both CIR and PIR on the same logical unit. | Yes       | Yes, at the queue level | Yes, at the logical unit |
| Excess rate support                  | –                 | Yes                    | –       | –                                                        | –         | Yes                     | Yes, at the logical unit |
| Shared scheduler support             | –                 | –                      | –       | Yes                                                      | Yes       | –                       | –                        |

## Schedulers on the Router Hardware, PIC, MIC, and MPC Families

Table 74 on page 873 compares the PIC, MIC, and MPC interface families with regard to scheduler statements or features. Note that this table reflects the ability to perform the scheduler function *at the PIC, MIC, or MPC interface level* and not necessarily on the system as a whole.

In this table, the OSE PICs refer to the 10-port 10-Gigabit OSE PICs (described in some guides as the 10-Gigabit Ethernet LAN/WAN PICs with SFP+).

**Table 74: Schedulers on the Router Hardware and Interface Families Compared**

| Scheduler Statement or Feature: | M320 and T Series | MIC and MPC Interfaces | IQ PICs | IQ2 PICs | IQ2E PICs | OSE PICs on T Series | Enhanced IQ PICs |
|---------------------------------|-------------------|------------------------|---------|----------|-----------|----------------------|------------------|
| Exact transmit rate             | Yes               | Yes                    | Yes     | –        | –         | Yes                  | Yes              |



**Table 74: Schedulers on the Router Hardware and Interface Families Compared (Continued)**

| Scheduler Statement or Feature:   | M320 and T Series | MIC and MPC Interfaces | IQ PICs | IQ2 PICs | IQ2E PICs | OSE PICs on T Series | Enhanced IQ PICs |
|-----------------------------------|-------------------|------------------------|---------|----------|-----------|----------------------|------------------|
| Rate-limit transmit rate          | –                 | Yes                    | –       | Yes      | Yes       | Yes                  | Yes              |
| More than one high-priority queue | Yes               | Yes                    | Yes     | –        | Yes       | –                    | Yes              |
| Excess priority or sharing        | –                 | Yes                    | –       | –        | –         | –                    | Yes              |
| Hierarchical Scheduling           | –                 | Yes, for EQ MPC        | –       | –        | Yes       | –                    | –                |

## Queuing Parameters for the Router Hardware, PIC, MIC, and MPC Interface Families

Table 75 on page 875 compares the PIC, MIC, and MPC interface families with regard to queuing parameters and features. In this table, the OSE PICs refer to the 10-port 10-Gigabit OSE PICs (described in some guides as the 10-Gigabit Ethernet LAN/WAN PICs with SFP+).



**Table 75: Queue Parameters on the Router Hardware and Interface Families Compared**

| Queuing Statement or Feature:   | M320 and T Series                          | MIC and MPC Interfaces                      | IQ PICs                                                  | IQ2 PICs | IQ2E PICs | OSE PICs on T Series | Enhanced IQ PICs |
|---------------------------------|--------------------------------------------|---------------------------------------------|----------------------------------------------------------|----------|-----------|----------------------|------------------|
| Maximum number of queues        | 8                                          | 8                                           | 8 on M320 or T Series routers, 4 on M7, M10, M20 routers | 8        | 8         | 4 ingress, 8 egress  | 8                |
| Maximum delay buffer bandwidth  | 80 ms: Type 1 and 2 FPC, 50 ms: Type 3 FPC | 100 ms for 1 Gbps and up; 500 ms for others | 100 ms                                                   | 200 ms   | 200 ms    | –                    | up to 4000 ms    |
| Packet transmit priority level  | 3 and 3                                    | 3 and 2                                     | 2 and 2                                                  | 2        | 3         | 2                    | 3 and 2          |
| Maximum number of drop profiles | 32 (32 samples)                            | 64                                          | 32 (32 samples)                                          | 32       | 32        | –                    | 64               |
| Packet loss priority level      | 4                                          | 4                                           | 4                                                        | 4        | 4         | 4                    | 4                |



# Configuring Class of Service for Tunnels

## IN THIS CHAPTER

- CoS for Tunnels Overview | 876
- Tunneling and BA Classifiers | 878
- Configuring CoS for GRE and IP-IP Tunnels | 878
- Example: Configuring CoS for GRE or IP-IP Tunnels | 879
- Configuring a GRE Tunnel to Copy ToS Bits to the Outer IP Header | 891

## CoS for Tunnels Overview

For Multiservices and Services PIC, Link Services, and Tunnel PICs installed on Juniper Networks M Series Multiservice Edge Routers and T Series Core Routers with enhanced Flexible PIC Concentrators (FPCs), class-of-service (CoS) information is preserved inside generic routing encapsulation (GRE) and IP-IP tunnels.

For the ES PIC installed on M Series and T Series routers with enhanced FPCs, class-of-service information is preserved inside IP Security (IPsec) tunnels. For IPsec tunnels, you do not need to configure CoS, because the ES PIC copies the type-of-service (ToS) byte from the inner IP header to the GRE or IP-IP header.

For IPsec tunnels, the IP header type-of-service (ToS) bits are copied to the outer IPsec header at encryption side of the tunnel. You can rewrite the outer ToS bits in the IPsec header using a rewrite rule. On the decryption side of the IPsec tunnel, the ToS bits in the IPsec header are not written back to the original IP header field. You can still apply a *firewall filter* to the ToS bits to apply a packet action on egress. For more information about ToS bits and the Multiservices PICs, see ["Multiservices PIC ToS Translation" on page 904](#). For more information about IPsec and Multiservices PICs, see the [Junos OS Services Interfaces Library for Routing Devices](#).



To configure CoS for tunnels, include the following statements at the [edit class-of-service] and [edit interfaces] hierarchy level:

```
[edit class-of-service]
interfaces {
  interface-name {
    unit logical-unit-number {
      rewrite-rules {
        dscp (rewrite-name | default);
        dscp-ipv6 (rewrite-name | default);
        exp (rewrite-name | default) protocol protocol-types;
        exp-push-push-push default;
        exp-swap-push-push default;
        ieee-802.1 (rewrite-name | default);
        inet-precedence (rewrite-name | default);
      }
    }
  }
}
rewrite-rules {
  (dscp | dscp-ipv6 | exp | ieee-802.1 | inet-precedence) rewrite-name {
    import (rewrite-name | default);
    forwarding-class class-name {
      loss-priority level code-point (alias | bits);
    }
  }
}
[edit interfaces]
gre-interface-name {
  unit logical-unit-number {
    copy-tos-to-outer-ip-header;
    copy-tos-to-outer-ip-header-transit;
    force-control-packets-on-transit-path
    tunnel {
      traffic-class traffic-class;
    }
  }
}
```



## RELATED DOCUMENTATION

[Configuring CoS for GRE and IP-IP Tunnels | 878](#)

[Example: Configuring CoS for GRE or IP-IP Tunnels | 879](#)

## Tunneling and BA Classifiers

BA classifiers can be used with GRE and IP-IP tunnels on the following routers and switches:

- EX Series switches
- M7i and M10i routers
- M Series routers with E-FPC or EP-FPC
- M120 routers
- M320 routers
- MX routers
- T Series routers



**NOTE:** MPCs do not support BA classifiers on gr- interfaces. Use multifield classifiers instead.

When a GRE or IP-IP tunnel is configured on an incoming (core-facing) interface, the queue number and PLP information are carried through the tunnel. At the egress (customer-facing) interface, the packet is queued and the CoS bits rewritten based on the information carried through the tunnel.

If no BA classifier is configured in the incoming interface, the default classifier is applied. If no rewrite rule is configured, the default rewrite rule is applied.



**NOTE:** For GRE and IP-IP tunnels, IP precedence and DSCP rewrite marking of the inner header do not work with more than eight forwarding classes.

## Configuring CoS for GRE and IP-IP Tunnels

To configure CoS for GRE and IP-IP tunnels, perform the following configuration tasks:



1. To configure CoS for an IP-IP tunnel, include the `tunnel` statement at the `[edit interfaces ip-fpc/pic/port unit logical-unit-number]` hierarchy level. To configure CoS for a GRE tunnel, include the `tunnel` statement at the `[edit interfaces gr-fpc/pic/port unit logical-unit-number]` hierarchy level.
2. To rewrite traffic on the outbound interface, include the `rewrite-rules` statement at the `[edit class-of-service]` and `[edit class-of-service interfaces interface-name unit logical-unit-number]` hierarchy levels. For GRE and IP-IP tunnels, you can configure IP precedence and DSCP rewrite rules.
3. To classify traffic on the inbound interface, you can configure a behavior aggregate (BA) classifier or a firewall filter. Include the `loss-priority` and `forwarding-class` statements at the `[edit firewall filter filter-name term term-name then]` hierarchy level, or the `classifiers` statement at the `[edit class-of-service]` hierarchy level.



**NOTE:** You cannot configure BA classifiers on `gr-` interfaces. You must classify traffic on `gr-` interfaces using firewall filters (multifield classifiers).

4. For a GRE tunnel, the default is to set the ToS bits in the outer IP header to all 0s. To copy the ToS bits from the inner IP header to the outer, include the `copy-tos-to-outer-ip-header-transit` statement at the `[edit interfaces gr-fpc/pic/port unit logical-unit-number]` hierarchy level. (This inner-to-outer ToS bits copying is already the default behavior for IP-IP tunnels.)

To verify that this option is enabled at the interface level, use the `show interfaces interface-name detail` command.

To set a static ToS/Traffic Class value in the outer IP header, include the `traffic-class traffic-class-value` statement at the `[edit interfaces gr-fpc/pic/port unit logical-unit-number tunnel]` hierarchy level.

Setting this value overrides the `copy-tos-to-outer-ip-header-transit` statement. If rewrite rules are configured on the egress WAN interface, those rewrite rules will overwrite this setting. Therefore the `traffic-class` setting only makes sense when no rewrite rules are configured.

## RELATED DOCUMENTATION

[CoS for Tunnels Overview](#) | 876

## Example: Configuring CoS for GRE or IP-IP Tunnels

### IN THIS SECTION

[Requirements](#) | 880



- Overview | 880
- Configuration | 881
- Configuring Router A | 883
- Configuring Router B | 888
- Configuring Router C | 889
- Verification | 890

This topic provides an example of how to configure class of service (CoS) for GRE or IP-IP tunnels.

## Requirements

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

## Overview

### IN THIS SECTION

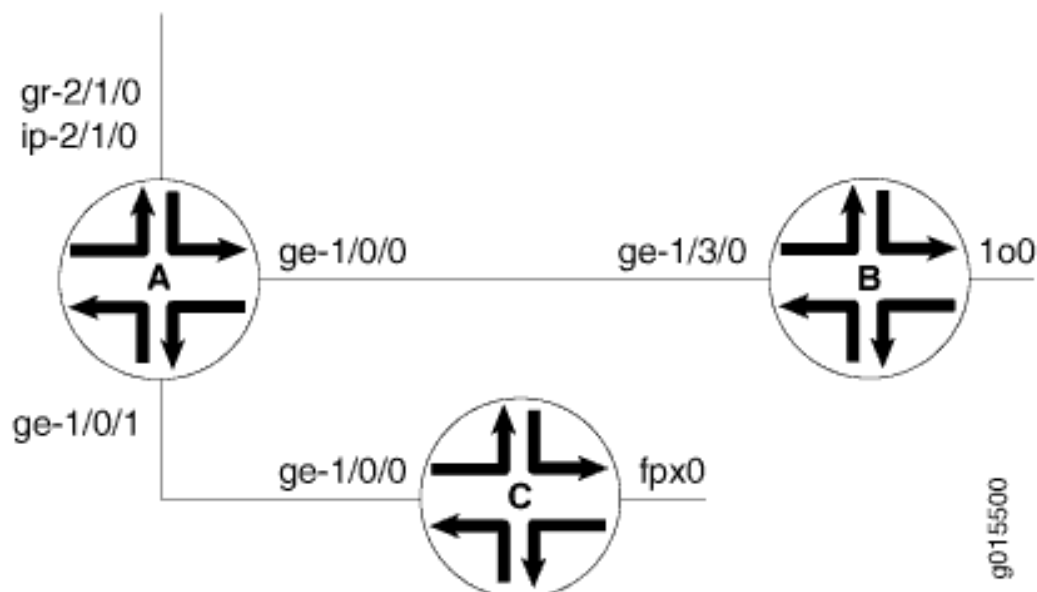
- Topology | 880

## Topology

In [Figure 62 on page 881](#), Router A acts as a tunnel ingress device. The link between interfaces `ge-1/0/0` in Router A and `ge-1/3/0` in Router B is the GRE or IP-IP tunnel. Router A monitors the traffic received from interface `ge-1/3/0`. By way of interface `ge-1/0/0`, Router C generates traffic to Router B.



Figure 62: CoS with a Tunnel Configuration



## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 881](#)

To configure this example, perform these tasks:

### CLI Quick Configuration

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

#### Router A

```
set interfaces ge-1/0/0 unit 0 family inet address 10.80.0.2/24
set interfaces ge-1/0/1 unit 0 family inet filter input zf-catch-all
set interfaces ge-1/0/1 unit 0 family inet address 10.90.0.2/24
set interfaces gr-2/1/0 unit 0 tunnel source 10.11.11.11 destination 10.255.245.46
set interfaces gr-2/1/0 unit 0 family inet address 10.21.21.21/24
```



```

set interfaces ip-2/1/0 unit 0 tunnel source 10.12.12.12 destination 10.255.245.46
set interfaces ip-2/1/0 unit 0 family inet address 10.22.22.22/24
set routing-options static route 10.1.1.1/32 next-hop gr-2/1/0.0
set routing-options static route 10.2.2.2/32 next-hop ip-2/1/0.0
set class-of-service interfaces ge-1/0/0 unit 0 rewrite-rules inet-precedence zf-tun-rw-ipprec-00
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class best-
effort loss-priority low code-point 000
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class best-
effort loss-priority high code-point 001
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class
expedited-forwarding loss-priority low code-point 010
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class
expedited-forwarding loss-priority high code-point 011
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class assured-
forwarding loss-priority low code-point 100
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class assured-
forwarding loss-priority high code-point 101
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class network-
control loss-priority low code-point 110
set class-of-service rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class network-
control loss-priority high code-point 111
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class best-effort loss-
priority low code-point 000000
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class best-effort loss-
priority high code-point 001001
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class expedited-forwarding
loss-priority low code-point 010010
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class expedited-forwarding
loss-priority high code-point 011011
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class assured-forwarding
loss-priority low code-point 100100
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class assured-forwarding
loss-priority high code-point 101101
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class network-control loss-
priority low code-point 110110
set class-of-service rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class network-control loss-
priority high code-point 111111
set firewall filter zf-catch-all term term1 then loss-priority high
set firewall filter zf-catch-all term term1 then forwarding-class network-control

```



## Router B

```
user@router-B# set interfaces ge-1/3/0 unit 0 family inet address 10.80.0.1/24
user@router-B# set interfaces lo0 unit 0 family inet address 10.255.245.46/32
```

## Router C

```
set interfaces ge-1/0/0 unit 0 family inet address 10.90.0.1/24
set routing-options static route 10.1.1.1/32 next-hop 10.90.0.2
set routing-options static route 10.2.2.2/32 next-hop 10.90.0.2
```

## Configuring Router A

### IN THIS SECTION

- [Procedure | 883](#)

## Procedure

### Step-by-Step Procedure

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

To configure router A:

1. Configure the device interfaces.

```
[edit interfaces]
user@router-A# set ge-1/0/0 unit 0 family inet address 10.80.0.2/24
user@router-A# set ge-1/0/1 unit 0 family inet filter input zf-catch-all
user@router-A# set ge-1/0/1 unit 0 family inet address address 10.90.0.2/24
user@router-A# set gr-2/1/0 unit 0 tunnel source 10.11.11.11 destination 10.255.245.46
user@router-A# set gr-2/1/0 unit 0 family inet address 10.21.21.21/24
user@router-A# set ip-2/1/0 unit 0 tunnel source 10.12.12.12 destination 10.255.245.46
user@router-A# set ip-2/1/0 unit 0 family inet address 10.22.22.22/24
```



## 2. Configure the static routes.

```
[edit routing-options static]
user@router-A# set static route 10.1.1.1/32 next-hop gr-2/1/0.0
user@router-A# set static route 10.2.2.2/32 next-hop ip-2/1/0.0
```

## 3. Apply the rewrite rule to the interface.

```
[edit class-of-service]
user@router-A# set interfaces ge-1/0/0 unit 0 rewrite-rules inet-precedence zf-tun-rw-ipprec-00
ipprec-00
```

## 4. Define the rewrite rules.

```
[edit class-of-service]
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class best-effort loss-priority low code-point 000
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class best-effort loss-priority high code-point 001
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class expedited-forwarding loss-priority low code-point 010
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class expedited-forwarding loss-priority high code-point 011
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class assured-forwarding loss-priority low code-point 100
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class assured-forwarding loss-priority high code-point 101
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class network-control loss-priority low code-point 110
user@router-A# set rewrite-rules inet-precedence zf-tun-rw-ipprec-00 forwarding-class network-control loss-priority high code-point 111
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class best-effort loss-priority low code-point 000000
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class best-effort loss-priority high code-point 001001
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class expedited-forwarding loss-priority low code-point 010010
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class expedited-forwarding loss-priority high code-point 011011
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class assured-forwarding
```



```

loss-priority low code-point 100100
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class assured-forwarding
loss-priority high code-point 101101
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class network-control loss-
priority low code-point 110110
user@router-A# set rewrite-rules dscp zf-tun-rw-dscp-00 forwarding-class network-control loss-
priority high code-point 111111

```

## 5. Configure the firewall filter.

```

[edit firewall]
user@router-A# set filter zf-catch-all term term1 then loss-priority high
user@router-A# set filter zf-catch-all term term1 then forwarding-class network-control

```

## Results

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

```

user@router-A# show interfaces
ge-1/0/0 {
  unit 0 {
    family inet {
      address 10.80.0.2/24;
    }
  }
}
ge-1/0/1 {
  unit 0 {
    family inet {
      filter {
        input zf-catch-all;
      }
      address 10.90.0.2/24;
    }
  }
}
gr-2/1/0 {
  unit 0 {

```



```

        tunnel {
            source 10.11.11.11;
            destination 10.255.245.46;
        }
        family inet {
            address 10.21.21.21/24;
        }
    }
}
ip-2/1/0 {
    unit 0 {
        tunnel {
            source 10.12.12.12;
            destination 10.255.245.46;
        }
        family inet {
            address 10.22.22.22/24;
        }
    }
}
}

```

```

user@router-A# show routing-options
static {
    route 10.1.1.1/32 next-hop gr-2/1/0.0;
    route 10.2.2.2/32 next-hop ip-2/1/0.0;
}

```

```

user@router-A# show class-of-service
interfaces {
    ge-1/0/0 {
        unit 0 {
            rewrite-rules {
                inet-precedence zf-tun-rw-ipprec-00;
            }
        }
    }
}
rewrite-rules {
    inet-precedence zf-tun-rw-ipprec-00 {
        forwarding-class best-effort {

```



```

        loss-priority low code-point 000;
        loss-priority high code-point 001;
    }
    forwarding-class expedited-forwarding {
        loss-priority low code-point 010;
        loss-priority high code-point 011;
    }
    forwarding-class assured-forwarding {
        loss-priority low code-point 100;
        loss-priority high code-point 101;
    }
    forwarding-class network-control {
        loss-priority low code-point 110;
        loss-priority high code-point 111;
    }
}
dscp zf-tun-rw-dscp-00 {
    forwarding-class best-effort {
        loss-priority low code-point 000000;
        loss-priority high code-point 001001;
    }
    forwarding-class expedited-forwarding {
        loss-priority low code-point 010010;
        loss-priority high code-point 011011;
    }
    forwarding-class assured-forwarding {
        loss-priority low code-point 100100;
        loss-priority high code-point 101101;
    }
    forwarding-class network-control {
        loss-priority low code-point 110110;
        loss-priority high code-point 111111;
    }
}

```

```

user@router-A# show firewall
filter zf-catch-all {
    term term1 {
        then {
            loss-priority high;

```



```

        forwarding-class network-control;
    }
}
}

```

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

## Configuring Router B

### IN THIS SECTION

- Procedure | 888

### Procedure

#### Step-by-Step Procedure

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

To configure router B:

1. Configure the device interfaces.

```

[edit interfaces]
user@router-B# set ge-1/3/0 unit 0 family inet address 10.80.0.1/24
user@router-B# set lo0 unit 0 family inet address 10.255.245.46/32

```

### Results

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

#### Router B

```

user@router-B# show interfaces
ge-1/3/0 {

```



```

    unit 0 {
        family inet {
            address 10.80.0.1/24;
        }
    }
}
lo0 {
    unit 0 {
        family inet {
            address 10.255.245.46/32;
        }
    }
}
}

```

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

## Configuring Router C

### IN THIS SECTION

- [Procedure | 889](#)

### Procedure

#### Step-by-Step Procedure

To configure router C:

1. Configure the device interfaces.

```

[edit interfaces]
user@router-B# set ge-1/0/0 unit 0 family inet address 10.90.0.1/24

```

2. Configure the static routes.

```

[edit routing-options static]
user@router-A# set static route 10.1.1.1/32 next-hop 10.90.0.2
user@router-A# set static route 10.2.2.2/32 next-hop 10.90.0.2

```



## Results

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

### Router C

```
user@router-C show interfaces
ge-1/0/0 {
  unit 0 {
    family inet {
      address 10.90.0.1/24;
    }
  }
}

user@router-C show routing-options
static {
  route 10.1.1.1/32 next-hop 10.90.0.2;
  route 10.2.2.2/32 next-hop 10.90.0.2;
}
```

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

## Verification

To verify the configuration, run the following commands:

- *show class-of-service rewrite-rule*
- *show firewall*

## RELATED DOCUMENTATION

---

[CoS for Tunnels Overview | 876](#)

---

[Configuring CoS for GRE and IP-IP Tunnels | 878](#)

---

[Tunneling and BA Classifiers | 878](#)



## Configuring a GRE Tunnel to Copy ToS Bits to the Outer IP Header

Unlike IP-IP tunnels, GRE tunnels do not copy the ToS bits to the outer IP header by default. To copy the inner ToS bits to the outer IP header (which is required for some tunneled routing protocols) on packets sent by the Routing Engine, include the `copy-tos-to-outer-ip-header` statement at the logical unit hierarchy level of a GRE interface. To copy the inner ToS bits to the outer IP header on packets transiting the device (MX Series routers with MPCs only), include the `copy-tos-to-outer-ip-header-transit` statement at the logical unit hierarchy level of a GRE interface.

To copy the inner ToS bits to the outer IP header on a GRE tunnel:

- Specify the interface on which to enable the inner IP header's ToS bits to be copied to the outer IP packet header

```
[edit]
user@host# edit interfaces
user@host# set gr-0/0/0 unit 0 copy-tos-to-outer-ip-header
user@host# set gr-0/0/0 unit 0 copy-tos-to-outer-ip-header-transit
user@host# set gr-0/0/0 unit 0 family inet
```

You can also copy the inner ToS bits to the outer IP header on packets transiting the device on a global basis for all GRE interfaces on MPCs by including the `copy-tos-to-outer service-type gre` statement at the `[edit chassis]` hierarchy level. This statement affects all GRE interfaces on MPCs and takes precedence over the `copy-tos-to-outer-ip-header-transit` statement. Once committed, this configuration statement only affects new `gr-` interfaces. To affect an existing `gr-` interface, you must delete and re-add the interface.

To verify that this option is enabled at the interface level, use the `show interfaces interface-name detail` command.

### RELATED DOCUMENTATION

[CoS for Tunnels Overview | 876](#)

[Configuring CoS for GRE and IP-IP Tunnels | 878](#)

*force-control-packets-on-transit-path*



# Configuring Class of Service on Services PICs

## IN THIS CHAPTER

- CoS on Services PICs Overview | 892
- Configuring CoS Rules on Services PICs | 894
- Configuring CoS Rule Sets on Services PICs | 901
- Example: Configuring CoS Rules on Services PICs | 902
- Packet Rewriting on Services Interfaces | 904
- Multiservices PIC ToS Translation | 904
- Fragmentation by Forwarding Class Overview | 905
- Configuring Fragmentation by Forwarding Class | 906
- Configuring Drop Timeout Interval for Fragmentation by Forwarding Class | 907
- Example: Configuring Fragmentation by Forwarding Class | 910
- Allocating Excess Bandwidth Among Frame Relay DLCIs on Multiservices PICs | 916
- Configuring Rate Limiting and Sharing of Excess Bandwidth on Multiservices PICs | 918

## CoS on Services PICs Overview

On Multiservices PICs with `lsq-` interfaces, there are additional features you can configure. One such feature is an additional method of classifying traffic flows based on applications, for example stateful firewalls and network address translation (NAT).

Application-based traffic flow classification enables you to configure a rule-based service that provides DiffServ code point (DSCP) marking and forwarding-class assignments for traffic transiting the Multiservices PIC. The service enables you to specify matching by application, application set, source, destination address, and match direction, and uses a similar structure to other rule-based services such as stateful firewall. The service actions allow you to associate the DSCP alias or value, forwarding-class name, system log activity, or a preconfigured application profile with the matched packet flows.





**NOTE:** If you configure a forwarding class map associating a forwarding class with a queue number, these maps are not supported on Multiservices link services intelligent queuing (lsq-) interfaces.

To configure class-of-service (CoS) features on the Multiservices PIC, include the `cos` statement at the [edit services] hierarchy level:

```
[edit services]
cos {
  application-profile profile-name {
    ftp {
      data {
        dscp (alias | bits);
        forwarding-class class-name;
      }
    }
    sip {
      video {
        dscp (alias | bits);
        forwarding-class class-name;
      }
      voice {
        dscp (alias | bits);
        forwarding-class class-name;
      }
    }
  }
  rule rule-name {
    match-direction (input | output | input-output);
    term term-name {
      from {
        application-sets [ set-names ];
        applications [ application-names ];
        destination-address address;
        destination-prefix-list list-name <except>;
        source-address address;
        source-address-range source-address-range low minimum-value high maximum-value
<except>;
        source-prefix-list list-name <except>;
      }
    }
  }
}
```



```

        then {
            application-profile profile-name;
            dscp (alias | bits);
            forwarding-class class-name;
            syslog;
            reflexive; | revert; | reverse { {
                application-profile profile-name;
                dscp (alias | bits);
                forwarding-class class-name;
                syslog;
            }
        }
    }
}
rule-set rule-set-name {
    [ rule rule-names ];
}
}

```

## RELATED DOCUMENTATION

[Configuring CoS Rules on Services PICs | 894](#)

[Configuring CoS Rule Sets on Services PICs | 901](#)

[Example: Configuring CoS Rules on Services PICs | 902](#)

## Configuring CoS Rules on Services PICs

### IN THIS SECTION

- [Configuring Match Conditions in a CoS Rule | 896](#)
- [Configuring Actions in a CoS Rule | 898](#)

This topic describes how to configure CoS rules on Services PICs.



Each CoS rule consists of a set of terms, similar to those in a firewall filter configuration. A term consists of the following:

- `from` statement—Specifies the match conditions and applications that are included and excluded.
- `then` statement—Specifies the actions and action modifiers to be performed by the router software.

If you omit the `from` term, the router accepts all traffic and the default protocol handlers take effect:

- User Datagram Protocol (UDP), Transmission Control Protocol (TCP), and Internet Control Message Protocol (ICMP) create a bidirectional flow with a predicted reverse flow.
- IP creates a unidirectional flow.

In addition, each rule must include a `match-direction` statement that specifies the direction in which the rule match is applied. To configure where the match is applied, include the `match-direction` statement at the `[edit services cos rule rule-name]` hierarchy level:

```
match-direction (input | output | input-output);
```

If you configure `match-direction input-output`, bidirectional rule creation is allowed.

The match direction is used with respect to the traffic flow through the Services PIC. When a packet is sent to the Services PIC, direction information is carried along with it.

On interface service sets, packet direction is determined by whether a packet is entering or leaving the interface on which the service set is applied.

With a next-hop service set, packet direction is determined by the interface used to route the packet to the Services PIC. If the inside interface is used to route the packet, the packet direction is input. If the outside interface is used to direct the packet to the Services PIC, the packet direction is output. For more information on inside and outside interfaces, see *Configuring Service Sets to be Applied to Services Interfaces*.

On the Services PIC, a flow lookup is performed. If no flow is found, rule processing is performed. All rules in the service set are considered. During rule processing, the packet direction is compared against rule directions. Only rules with direction information that matches the packet direction are considered.

You can use either the source address or the destination address as a match condition, in the same way that you would configure a firewall filter; for more information, see the [Routing Policies, Firewall Filters, and Traffic Policers User Guide](#).

You can also include application protocol definitions that you have configured at the `[edit applications]` hierarchy level; for more information, see the [Junos OS Services Interfaces Library for Routing Devices](#).

- To apply one or more specific application protocol definitions, include the `applications` statement at the `[edit services cos rule rule-name term term-name from]` hierarchy level.



- To apply one or more sets of application protocol definitions you have defined, include the `application-sets` statement at the `[edit services cos rule rule-name term term-name from]` hierarchy level.



**NOTE:** If you include a statement that specifies application protocols, the router derives port and protocol information from the corresponding configuration at the `[edit applications]` hierarchy level; you cannot specify these properties as match conditions.

The following sections describe how to configure CoS rules in more detail:

## Configuring Match Conditions in a CoS Rule

This topic describes how to configure the match conditions for CoS rules.

Before you begin, make sure you have completed the following tasks:

- Configure the application protocol definitions at the `[edit applications]` hierarchy level; for more information, see the `application` and [Junos OS Services Interfaces Library for Routing Devices](#).
- Configure a destination prefix list by including the `prefix-list` statement at the `[edit policy-options]` hierarchy level.
- Configure a source prefix list by including the `prefix-list` statement at the `[edit policy-options]` hierarchy level.

To configure the match conditions for a CoS rule:

1. Create the CoS rule by specifying a name for it.

```
[edit]
user@host# edit services cos rule rule-name
```

2. Specify the direction in which the rule match is applied.

```
[edit services cos rule rule-name]
user@host# set match-direction (input | output | input-output)
```

3. Specify the input conditions for the CoS term:

- a. Define one or more target application sets.

```
[edit services cos rule rule-name]
user@host# set term term-name from application-sets [ set-names ]
```





**NOTE:** You must configure the application protocol definitions at the [edit applications] hierarchy level; for more information, see the [Junos OS Services Interfaces Library for Routing Devices](#).

- b. Define one or more applications to which the CoS services apply.

```
[edit services cos rule rule-name]
user@host# set term term-name from applications [ application-names ]
```

- c. Specify the destination address for rule matching.

```
[edit services cos rule rule-name]
user@host# set term term-name from destination-address address
```

- d. Specify the name of the destination prefix list for rule matching.

```
[edit services cos rule rule-name]
user@host# set term term-name from destination-prefix-list list-name <except>
```



**NOTE:** You must configure the destination prefix list by including the prefix-list statement at the [edit policy-options] hierarchy level.

- e. Specify the source address for rule matching.

```
[edit services cos rule rule-name]
user@host# set term term-name from source-address address
```

- f. Specify the source address range for rule matching.

```
[edit services cos rule rule-name]
user@host# set term term-name from source-address-range source-address-range low minimum-value high maximum-value <except>
```



- g. Specify the source prefix list for rule matching.

```
[edit services cos rule rule-name]
user@host# set term term-name from source-prefix-list list-name <except>
```



**NOTE:** You must configure the source prefix list by including the `prefix-list` statement at the `[edit policy-options]` hierarchy level.

## Configuring Actions in a CoS Rule

### IN THIS SECTION

- [Configuring Application Profiles | 898](#)
- [Configuring Reflexive and Reverse CoS Actions | 899](#)

The principal CoS actions are:

- `dscp`—Marks the packet with the specified DiffServ code point (DSCP) value or alias.
- `forwarding-class`—Assigns the packet to the specified forwarding class.

This section describes how to configure these CoS actions and includes the following topics:

### Configuring Application Profiles

You can optionally define one or more application profiles for inclusion in CoS actions.

The `application-profile` statement includes two main components and three traffic types: `ftp` with the data traffic type and `sip` with the video and voice traffic types. You can set the appropriate `dscp` and `forwarding-class` values for each component within the application profile.



**NOTE:** The `ftp` and `sip` statements are not supported on Juniper Network MX Series 5G Universal Routing Platforms.

You can apply the application profile to a CoS configuration by including it at the `[edit services cos rule rule-name term term-name then]` hierarchy level.

To configure an application profile for inclusion in CoS actions:



1. Specify the application-profile statement at the [edit services cos] hierarchy level.

```
[edit]
user@host# edit services cos application-profile profile-name
```

2. Specify the appropriate dscp and forwarding-class value for FTP traffic.

```
[edit services cos application-profile profile-name]
user@host# set ftp data dscp (alias | bits)
user@host# set ftp data forwarding-class class-name
```

3. Specify the appropriate dscp and forwarding-class value for SIP video traffic.

```
[edit services cos application-profile profile-name]
user@host# set sip video dscp (alias | bits)
user@host# set sip video forwarding-class class-name
```

4. Specify the appropriate dscp and forwarding-class value for SIP voice traffic.

```
[edit services cos application-profile profile-name]
user@host# set sip voice dscp (alias | bits)
user@host# set sip voice forwarding-class class-name
```

## Configuring Reflexive and Reverse CoS Actions

It is important to understand that CoS services are unidirectional. It might be necessary to specify different treatments for flows in opposite directions.

Regardless of whether a packet matches the input, output, or input-output direction, flows in both directions are created. The difference is that a forward, reverse, or forward-and-reverse CoS action is associated with each flow. You should bear in mind that the flow in the opposite direction might end up having a CoS action associated with it, which you have not specifically configured.

To control the direction in which service is applied, separate from the direction in which the rule match is applied, you can configure the reflexive or reverse statement at the [edit services cos rule *rule-name* term *term-name* then] hierarchy level.

These two actions are mutually exclusive. If nothing is specified, data flows inherit the CoS behavior of the forward control flow.

- reflexive causes the equivalent reverse CoS action to be applied to flows in the opposite direction.



- reverse allows you to define the CoS behavior for flows in the reverse direction.

To control the direction in which a service is applied:

1. Define the CoS term actions.

```
[edit]
user@host# edit services cos rule rule-name term term-name then
```

2. Specify the action.

```
[edit services cos rule rule-name term term-name then]
user@host# set reflexive; | revert; | reverse {
```

3. Specify the application profile name.

```
[edit services cos rule rule-name term term-name then]
user@host# set application-profile profile-name
```

4. Define the Differentiated Services code point (DSCP) mapping that is applied to the packets.

```
[edit services cos rule rule-name term term-name then]
user@host# set dscp (alias | bits)
```

5. Define the forwarding class to which packets are assigned.

```
[edit services cos rule rule-name term term-name then]
user@host# set forwarding-class class-name
```

6. (Optional) Set the configuration to record information in the system logging facility.

Define the forwarding class to which packets are assigned.

```
[edit services cos rule rule-name term term-name then]
user@host# set syslog
```

## RELATED DOCUMENTATION

*Class of Service Overview*



*Restrictions and Cautions for CoS Configuration on Services Interfaces*

*Configuring CoS Rule Sets*

*Examples: Configuring CoS on Services Interfaces*

[CoS on Services PICs Overview | 892](#)

[Configuring CoS Rule Sets on Services PICs | 901](#)

[Example: Configuring CoS Rules on Services PICs | 902](#)

## Configuring CoS Rule Sets on Services PICs

You can define a collection of CoS rules that determine what actions the router software performs on packets in the data stream. Junos OS processes the rules in the order in which you specify them in the configuration. If a term in a rule matches the packet, the router performs the corresponding action and the rule processing stops. If no term in a rule matches the packet, processing continues to the next rule in the rule set. If none of the rules match the packet, the packet is dropped by default. The `rule-set` statement defines a collection of CoS rules that determine what actions the router software performs on packets in the data stream. You define each rule by specifying a rule name and configuring terms. You then specify the order of the rules by including the `rule-set` statement at the `[edit services cos]` hierarchy level:

This topic explains how to configure a set of CoS rules.

Before starting this procedure, make sure you define the terms and actions for the CoS rules in this rule set under the `[edit services cos rule rule-name]` hierarchy level.

To configure a CoS rule set:

1. Specify a name for the CoS rule set.

```
[edit]
user@host# edit services cos rule-set rule-set-name
```

2. Specify the name of each rule you want included in the rule set.





**NOTE:** Junos OS processes the rules in the order in which you specify them in the configuration.

```
[edit services cos rule-set rule-set-name]
user@host# set rule rule-name1
user@host# set rule rule-name2
```

## RELATED DOCUMENTATION

[CoS on Services PICs Overview | 892](#)

[Configuring CoS Rules on Services PICs | 894](#)

[Example: Configuring CoS Rules on Services PICs | 902](#)

## Example: Configuring CoS Rules on Services PICs

The following example show a CoS configuration containing two rules, one for input matching on a specified application set and the other for output matching on a specified source address:

```
[edit services]
cos {
  application-profile cosprofile {
    ftp {
      data {
        dscp af11;
        forwarding-class 1;
      }
    }
  }
  application-profile cosrevprofile {
    ftp {
      data {
        dscp af22;
      }
    }
  }
}
```



```

rule cosrule {
    match-direction input;
    term costerm {
        from {
            source-address {
                any-unicast;
            }
            applications junos-ftp;
        }
        then {
            dscp af33;
            forwarding-class 3;
            application-profile cosprofile;
            reverse {
                dscp af43;
                application-profile cosrevprofile;
            }
        }
    }
}

stateful-firewall {
    rule r1 {
        match-direction input;
        term t1 {
            from {
                application-sets junos-algs-outbound;
            }
            then {
                accept;
            }
        }
        term t2 {
            then {
                accept;
            }
        }
    }
}

service-set test {
    stateful-firewall-rules r1;
    cos-rules cosrule;
    interface-service {
        service-interface sp-1/3/0;
    }
}

```



```

    }
  }
}

```

## RELATED DOCUMENTATION

[Configuring CoS Rules on Services PICs | 894](#)

[Configuring CoS Rule Sets on Services PICs | 901](#)

## Packet Rewriting on Services Interfaces

On M Series routers, you can configure *rewrite rules* to change packet header information and attach it to an output interface. Because these rules can possibly overwrite the DSCP marking configured on Multiservices and Services PICs, it is important to create system-wide configurations carefully.

For example, knowing that the Services or Multiservices PICs can mark packets with any ToS or DSCP value and the output interface is restricted to only eight DSCP values, rewrite rules on the output interface condense the mapping from 64 to 8 values with overall loss of granularity. In this case, you have the following options:

- Remove rewrite rules in the output interface.
- Configure the output interface to include the most important mappings.

## RELATED DOCUMENTATION

[Rewriting Packet Headers to Ensure Forwarding Behavior | 532](#)

[Configuring Rewrite Rules | 538](#)

## Multiservices PIC ToS Translation

By default, all logical (lsq-) interfaces on a Multiservices PIC preserve the type-of-service (ToS) bits in an incoming packet header.

However, you can use the `translation-table` statement at the `[edit class-of-service]` hierarchy level to replace the arriving ToS bit pattern with a user-defined value.



This feature follows exactly the same configuration rules as the Enhanced IQ PIC. For configuration details, see ["Configuring ToS Translation Tables" on page 952](#).

## Fragmentation by Forwarding Class Overview

For Multiservices and Services *Physical Interface Card* (PIC) link services IQ (LSQ) and virtual LSQ redundancy (rlsq-) interfaces, you can specify fragmentation properties for specific forwarding classes. Traffic on each forwarding class can be either multilink fragmented or interleaved. By default, traffic in all forwarding classes is fragmented.

If you do not configure fragmentation properties for particular forwarding classes in multilink Point-to-Point Protocol (MLPPP) interfaces, the fragmentation threshold you set at the [edit interfaces *interface-name* unit *logical-unit-number* fragment-threshold] hierarchy level is used for all forwarding classes within the MLPPP interface. For multilink Frame Relay (MLFR) FRF.16 interfaces, the fragmentation threshold you set at the [edit interfaces *interface-name* mlfr-uni-nni-bundle-options fragment-threshold] hierarchy level is used for all forwarding classes within the MLFR FRF.16 interface. If you do not set a maximum fragment size anywhere in the configuration, packets are still fragmented if they exceed the smallest maximum transmission unit (MTU) of all the links in the bundle.

To configure fragmentation by forwarding class, include the following statements at the [edit class-of-service] hierarchy level:

```
[edit class-of-service]
fragmentation-maps {
  map-name {
    forwarding-class class-name {
      drop-timeout milliseconds;
      fragment-threshold bytes;
      multilink-class number;
      no-fragmentation;
    }
  }
}
interfaces {
  interface-name {
    unit logical-unit-number {
      fragmentation-map map-name;
    }
  }
}
```



## RELATED DOCUMENTATION

[Configuring Fragmentation by Forwarding Class | 906](#)

[Example: Configuring Fragmentation by Forwarding Class | 910](#)

[Configuring Drop Timeout Interval for Fragmentation by Forwarding Class | 907](#)

*fragmentation-map*

*fragmentation-maps*

## Configuring Fragmentation by Forwarding Class

For Multiservices and Services PIC link services IQ (LSQ) and virtual LSQ redundancy (rlsq-) interfaces only, you can configure fragmentation properties on a particular forwarding class.

To configure fragmentation properties on a specific forwarding class:

1. Specify the name of the fragmentation map and forwarding class.

```
[edit]
user@host# edit class-of-service fragmentation-maps map-name forwarding-class class-name
```

2. Specify how many milliseconds to wait for fragments.

```
[edit class-of-service fragmentation-maps map-name forwarding-class class-name]
user@host# set drop-timeout milliseconds
```



**NOTE:** If you set this value, you must also include a multilink-class value for resequencing fragments.

3. (Optional) Specify the maximum size, in bytes, for multilink packet fragments.

```
[edit class-of-service fragmentation-maps map-name forwarding-class class-name]
user@host# set fragment-threshold bytes
```



**NOTE:** If you set the option, you cannot configure no-fragmentation for the forwarding class.



4. Specify the multilink class assigned to this forwarding class.

```
[edit class-of-service fragmentation-maps map-name forwarding-class class-name]
user@host# set multilink-class number
```

5. (Optional) Specify that the traffic on this particular forwarding class is interleaved, rather than fragmented.

```
[edit class-of-service fragmentation-maps map-name forwarding-class class-name]
user@host# set no-fragmentation
```

6. Apply the fragmentation map to the logical interface.

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

## RELATED DOCUMENTATION

[Fragmentation by Forwarding Class Overview | 905](#)

[Example: Configuring Fragmentation by Forwarding Class | 910](#)

[Configuring Drop Timeout Interval for Fragmentation by Forwarding Class | 907](#)

[fragmentation-map](#)

[fragmentation-maps](#)

## Configuring Drop Timeout Interval for Fragmentation by Forwarding Class

For LSQ interfaces configured for multiclass MLPPP, you can change the drop timeout interval that the interface waits for fragment resequencing by forwarding class. This feature is mutually exclusive with the no-fragmentation statement configured for a forwarding class.

You can also disable the fragment resequencing function altogether by forwarding class. You do this by setting the drop-timeout interval to 0.

The drop-timeout interval can also be set at the bundle level. When the drop-timeout interval is set to 0 at the bundle level, *none* of the individual classes forward fragmented packets. Sequencing is ignored also,



and packets are forwarded in the order in which they were received. The drop-timeout interval value configured at the bundle level overrides the values configured at the class level.

This example configures a logical unit on an LSQ interface with a fragmentation map setting different drop timeout values for each forwarding class:

- Best effort (BE)—The value of 0 means that no resequencing of fragments takes place for BE traffic.
- Expedited Forwarding (EF)—The value of 800 ms means that the multiclass MLPPP waits 800 ms for fragment to arrive on the link for EF traffic.
- Assured Forwarding (AF)—The absence of the timeout statements means that the default timeouts of 500 ms for links at T1 and higher speeds and 1500 ms for lower speeds are in effect for AF traffic.
- Network Control (NC)—The value of 100 ms means that the multiclass MLPPP waits 100 ms for fragment to arrive on the link for NC traffic.

To configure the drop timeout interval:

1. Define the fragmentation properties for each forwarding class.

```
[edit]
user@host# edit class-of-service fragmentation-maps Timeout_Frag_Map
user@host# set forwarding-class BE drop-timeout 0 multilink-class 3 fragment-threshold 128
user@host# set forwarding-class EF drop-timeout 800 multilink-class 2
user@host# set forwarding-class NC drop-timeout 100 multilink-class 0 fragment-threshold 512
user@host# set forwarding-class AF multilink-class 1 fragment-threshold 256
```

2. Apply the fragmentation map to the logical interface.

```
[edit class-of-service]
user@host# set interfaces lsq-1/0/0 unit 1 fragmentation-map Timeout_Frag_Map
```

3. Verify the configuration.

### Verify the Configuration of the Fragmentation Map Properties

```
[edit class-of-service fragmentation-maps Timeout_Frag_Map]
user@host# show
forwarding-class {
    BE {
        fragment-threshold 128;
        multilink-class 3;
        drop-timeout 0; # no resequencing for this forwarding class
```



```

}
EF {
    multilink-class 2;
    drop-timeout 800;
}
NC {
    fragment-threshold 512;
    multilink-class 0;
    drop-timeout 100;
}
AF {
    fragment-threshold 256; # Default timeout in effect for this class
    multilink-class 1;
}
}

```

#### Verify the Fragmentation Map is Applied to the Logical Interface

```

[edit class-of-service]
user@host# show
interfaces {
    lsq-1/0/0 {
        unit 1 {
            fragmentation-map Tineout_frag_Map;
        }
    }
}

```

#### 4. Save the configuration.

```

[edit]
user@host# commit

```

## RELATED DOCUMENTATION

[Example: Configuring Fragmentation by Forwarding Class | 910](#)

[Configuring Fragmentation by Forwarding Class | 906](#)



## Example: Configuring Fragmentation by Forwarding Class

### IN THIS SECTION

- [Requirements | 910](#)
- [Overview | 910](#)
- [Configuration | 910](#)
- [Verification | 914](#)

This example shows you how to configure fragmentation maps for specific forwarding classes on Multiservices PICs or Services PICs.

### Requirements

This example uses the following hardware and software components:

- Multiservices PIC or Services PIC.

### Overview

Configure two logical units on an LSQ interface. The logical units use two different fragmentation maps.

### Configuration

#### IN THIS SECTION

- [CLI Quick Configuration | 911](#)
- [Define the Fragmentation Maps | 911](#)
- [Associate the Fragmentation Maps with the MLPPP Interface | 914](#)

To configure fragmentation maps for specific forwarding classes, perform these tasks:



## CLI Quick Configuration

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

### Define the Fragmentation Maps

```
set class-of-service fragmentation-maps frag-map-A forwarding-class AF no-fragmentation
set class-of-service fragmentation-maps frag-map-A forwarding-class EF no-fragmentation
set class-of-service fragmentation-maps frag-map-A forwarding-class BE fragment-threshold 100
set class-of-service fragmentation-maps frag-map-B forwarding-class EF fragment-threshold 200
set class-of-service fragmentation-maps frag-map-B forwarding-class BE fragment-threshold 200
set class-of-service fragmentation-maps frag-map-B forwarding-class AF fragment-threshold 200
```

### Associate the Fragmentation Map with an Interface

```
set class-of-service interfaces lsq-1/0/0 unit 1 fragmentation-map frag-map-A
set class-of-service interfaces lsq-1/0/0 unit 2 fragmentation-map frag-map-B
```

## Define the Fragmentation Maps

### Step-by-Step Procedure

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

To define the fragmentation maps:

1. Specify a name for the first fragmentation map.

```
[edit]
user@host# edit class-of-service fragmentation-maps frag-map-A
```

2. Define the first fragmentation map.



- a. Define the fragmentation properties for the AF forwarding class to be interleaved, rather than fragmented.

```
[edit class-of-service fragmentation-maps frag-map-A]
user@ost# set forwarding-class AF no-fragmentation
```

- b. Define the fragmentation properties for the EF forwarding class to be interleaved, rather than fragmented.

```
[edit class-of-service fragmentation-maps frag-map-A]
user@ost# set forwarding-class EF no-fragmentation
```

- c. Define the fragmentation properties for the BE forwarding class to be fragmented.

```
[edit class-of-service fragmentation-maps frag-map-A]
user@ost# set forwarding-class BE fragment-threshold 100
```

### 3. Define the second fragmentation map.

- a. Specify a name for the second fragmentation map.

```
[edit class-of-service fragmentation-maps]
user@host# edit frag-map-B
```

- b. Define the fragmentation properties for the EF forwarding class to be fragmented.

```
[edit class-of-service fragmentation-maps frag-map-B]
user@ost# set forwarding-class EF fragment-threshold 200
```

- c. Define the fragmentation properties for the BE forwarding class to be fragmented.

```
[edit class-of-service fragmentation-maps frag-map-B]
user@ost# set forwarding-class BE fragment-threshold 200
```



- d. Define the fragmentation properties for the AF forwarding class to be fragmented.

```
[edit class-of-service fragmentation-maps frag-map-B]
user@ost# set forwarding-class AF fragment-threshold 200
```

## Results

Verify the configuration of the fragmentation maps and forwarding classes.

```
[edit class-of-service fragmentation-maps]
user@host# show
frag-map-A {
  forwarding-class {
    AF {
      no-fragmentation;
    }
    EF {
      no-fragmentation;
    }
    BE {
      fragment-threshold 100;
    }
  }
}
frag-map-B {
  forwarding-class {
    EF {
      fragment-threshold 200;
    }
    BE {
      fragment-threshold 200;
    }
    AF {
      fragment-threshold 200;
    }
  }
}
```



## Associate the Fragmentation Maps with the MLPPP Interface

### Step-by-Step Procedure

To associate a fragmentation map with an interface:

1. Associate each fragmentation map with a logical interface.

```
[edit]
user@host# edit class-of-service interfaces lsq-1/0/0
user@host# set unit 1 fragmentation-map frag-map-A
user@host# set unit 2 fragmentation-map frag-map-B
```

### Results

Verify that the fragmentation maps are associated with the interfaces.

```
[edit class-of-service]
user@host# show
interfaces {
  lsq-1/0/0 {
    unit 1 {
      fragmentation-map frag-map-A;
    }
    unit 2 {
      fragmentation-map frag-map-B;
    }
  }
}
```

### Verification

#### IN THIS SECTION

- [Verifying the Fragmentation Properties](#) | 915



## Verifying the Fragmentation Properties

### Purpose

Verify the fragmentation properties for specific forwarding classes.

### Action

The following output displays the fragmentation properties and forwarding class association.

```
user@host> show class-of-service fragmentation-map
Fragmentation map: frag-map-A, Index: 19801
  Forwarding class: AF
  No Fragmentation

  Forwarding class: EF
  No Fragmentation

  Forwarding class: BE
  Fragmentation threshold: 100

Fragmentation map: frag-map-B, Index: 19855
  Forwarding class: EF
  Fragmentation threshold: 200

  Forwarding class: BE
  Fragmentation threshold: 200

  Forwarding class: AF
  Fragmentation threshold: 200
```

### Meaning

The output shows the forwarding class associated with each fragmentation map, as well as the fragmentation properties associated with the forwarding class.

## RELATED DOCUMENTATION

---

[Fragmentation by Forwarding Class Overview | 905](#)

[Configuring Fragmentation by Forwarding Class | 906](#)



## Allocating Excess Bandwidth Among Frame Relay DLCIs on Multiservices PICs

By default, all logical (lsq-) interfaces on a Multiservices PIC share bandwidth equally in the excess region (that is, bandwidth available once these interfaces have exhausted their committed information rate (CIR)).

However, you can include the `excess-rate` statement to control an independent set of parameters for bandwidth sharing in the excess region of a frame relay data-link connection identifier (DLCI) on a Multiservices PIC. Include the `excess-rate` statement at the `[edit class-of-service traffic-control-profile traffic-control-profile-name]` hierarchy level.

There are several limitations to this feature:

- The excess bandwidth comes from bandwidth not used by any DLCIs (that is, bandwidth above the CIR). Therefore, only FRF.16 is supported.
- Only CIR mode is supported (you must configure a CIR on at least one DLCI).
- Only the percent option is supported for lsq- interfaces. The priority option is not supported for DLCIs.
- You cannot configure this feature if you also include one of the following statements in the configuration:
  - `scheduler-map`
  - `shaping-rate`
- If you oversubscribe the DLCIs, then the bandwidth can only be distributed equally.
- The `excess-priority` statement is not supported. However, for consistency, this statement will not result in a commit error.
- This feature is only supported on the Multiservices 100, Multiservices 400, and Multiservices 500 PICs.

The following procedure configures excess bandwidth sharing in the ratio of 70 to 30 percent for two frame relay DLCIs. Only FRF.16 interfaces are supported.

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



1. Enable the association of scheduler map names with logical interfaces.

```
[edit]
user@host# edit interfaces lsq-1/3/0:0
user@host# set per-unit-scheduler unit 0 dlci 100
user@host# set per-unit-scheduler unit 1dlci 200
```

2. Configure the traffic control profiles.



**NOTE:** Only the percent option is supported.

```
[edit class-of-service]
user@host# set traffic-control-profiles tc_70 excess-rate percent 70
user@host# set traffic-control-profiles tc_30 excess-rate percent 30
```

3. Apply the traffic control profiles to the logical interface.



**NOTE:** Only FRF.16 is supported.

```
[edit]
user@host# edit interfaces lsq-1/3/0
user@host# set unit 0 output-traffic-control-profile tc_70
user@host# set unit 1 output-traffic-control-profile tc_30
```

4. Verify the configuration.

```
[edit interfaces lsq-1/3/0:0]
user@host# show
per-unit-scheduler;
unit 0 {
    dlci 100;
}
unit 1 {
```



```

    dlci 200;
}

```

```

[edit class-of-service]
user@host# show
traffic-control-profiles {
  tc_70 {
    excess-rate percent 70;
  }
  tc_30 {
    excess-rate percent 30;
  }
}

```

```

[edit interfaces]
user@host# show
lsq-1/3/0 {
  unit 0 {
    output-traffic-control-profile tc_70;
  }
  unit 1 {
    output-traffic-control-profile tc_30;
  }
}

```

## RELATED DOCUMENTATION

[Allocating Excess Bandwidth Among Frame Relay DLCIs on Multiservices PICs](#) | 916

## Configuring Rate Limiting and Sharing of Excess Bandwidth on Multiservices PICs

On Multiservices PICs, you can limit the transmit rate of a logical interface (lsq-) in the same way as other types of queuing PICs. You can also assign a percentage of the excess bandwidth to the logical interfaces. As with other types of PICs, the strict-high queue (voice) can “starve” low and medium priority queues. To prevent the strict-high queue from starving other queues, rate-limit the queue.



To rate-limit logical interfaces on a Multiservices PIC, include the `transmit-rate` statement with the `rate-limit` option at the `[edit class-of-service schedulers scheduler-name]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
transmit-rate (rate | percent percentage | remainder) rate-limit;
```

You can also make the excess strict-high bandwidth available for other queues. You can split the excess bandwidth among multiple queues, but the total excess bandwidth assigned to these queues can only add up to 100 percent. The `excess-bandwidth priority` statement option is not supported on the Multiservices PIC. For more information about excess bandwidth sharing, see ["Configuring Excess Bandwidth Sharing on IQE PICs" on page 959](#).

To share excess bandwidth among Multiservices PICs, include the `excess-rate` statement at the `[edit class-of-service schedulers scheduler-name]` hierarchy level.

```
[edit class-of-service schedulers scheduler-name]
excess-rate percent percentage;
```

Both of these rate-limiting and excess bandwidth sharing features apply to egress traffic only, and only for per-unit schedulers. Hierarchical schedulers and shared schedulers are not supported.

You must still complete the configuration by configuring the scheduler map and applying it to the Multiservices PIC interface.

This example configures a rate limit and excess bandwidth sharing for a Multiservices PIC interface.

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

1. Specify the scheduler name and parameter values.

```
[edit]
user@host# edit class-of-service
user@host# set schedulers scheduler0 transmit-rate percent 10 rate-limit
user@host# set schedulers scheduler0 priority strict-high excess-rate percent 30
user@host# set schedulers scheduler1 transmit-rate percent 1 rate-limit
user@host# set schedulers scheduler1 priority high excess-rate percent 70
```



2. Specify a scheduler map name and associate it with the scheduler configuration and forwarding class.

```
[edit class-of-service]
user@host# set scheduler-maps scheduler0 forwarding-class ef scheduler scheduler0
user@host# set scheduler-maps scheduler0 forwarding-class af scheduler scheduler1
```

3. Associate the scheduler map name with the interface.

```
[edit class-of-service]
user@host# set interfaces lsq-1/3/0 unit 0 scheduler-map scheduler0
user@host# set interfaces lsq-1/3/0 unit 1 scheduler-map scheduler1
```

4. Verify the configuration.

```
scheduler0 {
  transmit-rate {
    percent 10;
    rate-limit;
  }
  excess-rate percent 30;
  priority strict-high;
}
scheduler1 {
  transmit-rate {
    percent 1;
    rate-limit;
  }
  excess-rate percent 70;
  priority high;
}
```

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

```
interfaces {
  lsq-1/3/0 {
    unit 0 {
      scheduler-map scheduler0;
```



```
    }  
  }
```

```
scheduler-maps {  
  scheduler0 {  
    forwarding-class ef scheduler scheduler0;  
    forwarding-class af scheduler scheduler1;  
  }  
}
```

## RELATED DOCUMENTATION

---

[CoS on Services PICs Overview | 892](#)

---

[Configuring Schedulers | 338](#)

---

[Configuring Scheduler Maps | 338](#)

---

[Excess Rate and Excess Priority Configuration Examples | 378](#)



# Configuring Class of Service on IQ and Enhanced IQ (IQE) PICs

## IN THIS CHAPTER

- CoS on Enhanced IQ PICs Overview | 922
- Calculation of Expected Traffic on IQE PIC Queues | 923
- Configuring the Junos OS to Support Eight Queues on IQ Interfaces for T Series and M320 Routers | 950
- BA Classifiers and ToS Translation Tables | 951
- Configuring ToS Translation Tables | 952
- Configuring Hierarchical Layer 2 Policers on IQE PICs | 957
- Configuring Excess Bandwidth Sharing on IQE PICs | 959
- Configuring Rate-Limiting Policers for High Priority Low-Latency Queues on IQE PICs | 966
- Applying Scheduler Maps and Shaping Rate to Physical Interfaces on IQ PICs | 970
- Applying Scheduler Maps to Chassis-Level Queues | 991

## CoS on Enhanced IQ PICs Overview

The Enhanced IQ (IQE) PIC family supports a series of non-channelized and channelized interfaces that run at a large variety of speeds. Sophisticated Class-of-Service (CoS) techniques are available for the IQE PICs at the channel level. These techniques include policing based on type-of-service (ToS) bits, five priority levels, two shaping rates (the guaranteed rate and shaping rate), a shared scheduling option, DiffServ code point (DSCP) rewrite on egress, and configurable delay buffers for queuing. All of these features, with numerous examples, are discussed in this chapter.

For information about CoS components that apply generally to all interfaces, see ["Understanding How Class of Service Manages Congestion and Controls Service Levels in the Network" on page 2](#). For general information about configuring interfaces, see the [Junos OS Network Interfaces Library for Routing Devices](#).

IQE PICs can be used in Juniper Networks M40e, M120, M320 Multiservice Edge Routers and T Series Core Routers to supply enhanced CoS capabilities for edge aggregation. The same interface



configuration syntax is used for basic configuration, and other CoS statements are applied at channel levels. Some configuration statements are available only in Junos OS Release 9.3 and later, as noted in this chapter.

## Calculation of Expected Traffic on IQE PIC Queues

### IN THIS SECTION

- [Excess Bandwidth Calculations Terminology | 923](#)
- [Excess Bandwidth Basics | 924](#)
- [Logical Interface Modes on IQE PICs | 926](#)
- [Default Rates for Queues on IQE PICs | 931](#)
- [Sample Calculations of Excess Bandwidth Sharing on IQE PICs | 933](#)

This topic discusses the following topics related to calculating the expected traffic flow on IQE PIC queues:

### Excess Bandwidth Calculations Terminology

The following terms are used in this discussion of IQE PIC queue calculations:

- CIR mode—A physical interface is in CIR mode when one of more of its “children” (logical interfaces in this case) have a guaranteed rate configured, but some logical interfaces have a shaping rate configured.
- Default mode—A physical interface is in default mode if none of its “children” (logical interfaces in this case) have a guaranteed rate or shaping rate configured.
- Excess mode—A physical interface is in excess mode when one of more of its “children” (logical interfaces in this case) have an excess rate configured.
- PIR mode—A physical interface is in PIR mode if none of its “children” (logical interfaces in this case) have a guaranteed rate configured, but some logical interfaces have a shaping rate configured.



## Excess Bandwidth Basics

This basic example illustrates the interaction of the guaranteed rate, the shaping rate, and the excess rate applied to four queues. The same concepts extend to logical interfaces (units) and cases in which the user does not configure an explicit value for these parameters (in that case, the system uses implicit parameters).

In this section, the term “not applicable” (NA) means that the feature is not explicitly configured. All traffic rates are in megabits per second (Mbps).

The hardware parameters derived from the configured rates are relatively straightforward except for the excess weight. The excess rate is translated into an absolute value called the excess weight. The scheduler for an interface picks a logical unit first, and then a queue within the logical unit for transmission. Logical interfaces and queues that are within their guaranteed rates are picked first, followed by those in the excess region. If the transmission rate for a *logical interface* or queue is more than the shaping rate, the scheduler skips the logical interface or queue. Scheduling in the guaranteed region uses straight round-robin, whereas scheduling in the excess region uses weighed round-robin (WRR) based on the excess weights. The excess weights are in the range from 1 to 127, but they are transparent to the user and subject to change with implementation. The weights used in this example are for illustration only.

This example uses a logical interface with a transmit rate (CIR) of 10 Mbps and a shaping rate (PIR) of 10 Mbps. The user has also configured percentage values of transmit rate (CIR), shaping rate (PIR), and excess rate as shown in [Table 76 on page 924](#).

**Table 76: Basic Example of Excess Bandwidth**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 5%                  | 5%                 | 10%         | 10 Mbps               |
| Q1    | 30%                 | 80%                | 50%         | 10 Mbps               |
| Q2    | 10%                 | 15%                | 30%         | 10 Mbps               |
| Q3    | 15%                 | 35%                | 30%         | 10 Mbps               |

The values used by the hardware based on these parameters are shown in [Table 77 on page 925](#).



**Table 77: Hardware Use of Basic Example Parameters**

| Queue   | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Weight | Expected Traffic Rate    |
|---------|---------------------|--------------------|---------------|--------------------------|
| Q0      | 0.5 Mbps            | 0.5 Mbps           | 10            | 0.5 Mbps                 |
| Q1      | 3 Mbps              | 8 Mbps             | 50            | 5.19 Mbps                |
| Q2      | 1 Mbps              | 1.5 Mbps           | 30            | 1.5 Mbps                 |
| Q3      | 1.5 Mbps            | 3.5 Mbps           | 30            | 2.81 Mbps                |
| Totals: | 6 Mbps              | 13.5 Mbps          | 120           | 10 Mbps (maximum output) |

There are a number of important points regarding excess bandwidth calculations:

- The guaranteed rates should add up to less than the logical interface guaranteed rate (10 Mbps).
- Shaping rates (PIRs) can be oversubscribed.
- Excess rates can be oversubscribed. This rate is only a ratio at which the sharing occurs.
- Each queue receives the minimum of the guaranteed bandwidth because each queue is transmitting at its full burst if it can.
- The excess (remaining) bandwidth is shared among the queues in the ratio of their excess rates. In this case, the excess bandwidth is the logical interface bandwidth minus the sum of the queue transmit rates, or  $10 \text{ Mbps} - 6 \text{ Mbps} = 4 \text{ Mbps}$ .
- However, transmission rates are capped at the shaping rate (PIR) of the queue. For example, Queue 0 gets 0.5 Mbps.
- Queue 0 also gets a guaranteed transmit rate (CIR) of 0.5 Mbps and is eligible for excess bandwidth calculated as 4 Mbps ( $10 \text{ Mbps} - 6 \text{ Mbps}$ ) multiplied by  $10/127$ . However, because the shaping rate (PIR) for Queue 0 is 0.5 Mbps, the expected traffic rate is capped at 0.5 Mbps.
- Queue 1 gets its guaranteed transmit rate (CIR) of 3 Mbps. Because Queue 0 has already been dealt with, Queue 1 is eligible for sharing the excess bandwidth along with Queue 2 and Queue 3. So Queue 1 is entitled to an excess bandwidth of 4 Mbps multiplied by  $50 / (30 + 30 + 50)$ , or 1.81 Mbps.
- In the same way, Queue 2 is eligible for its guaranteed transmit rate (CIR) of 1 Mbps and an excess bandwidth of 4 Mbps multiplied by  $30 / (30 + 30 + 50)$ , or 1.09 Mbps. However, because Queue 2



has a shaping rate (PIR) of 1.5 Mbps, the bandwidth of Queue 2 is capped at 1.5 Mbps. The additional 0.59 Mbps can be shared by Queue 1 and Queue 3.

- Queue 3 is eligible for an excess of 4 Mbps multiplied by  $30 / (30 + 30 + 50)$ , or 1.09 Mbps. This total of 2.59 Mbps is still below the shaping rate (PIR) for Queue 3 (3.5 Mbps).
- The remaining bandwidth of 0.59 Mbps (which Queue 2 could not use) is shared between Queue 1 and Queue 3 in the ratio 50/30. So Queue 3 can get 0.59 multiplied by  $30 / (50 + 30)$ , or 0.22 Mbps. This gives a total of 2.81 Mbps.
- Therefore, Queue 1 gets 3 Mbps + 1.82 Mbps +  $(0.59 \text{ Mbps} * 50 / (50 + 30))$ , or approximately 5.19 Mbps.

## Logical Interface Modes on IQE PICs

On IQE PICs, scheduling occurs level-by-level. That is, based on the parameters configured on the logical interface, the scheduler first picks a logical interface to transmit from. Then, based on the configuration of the underlying queues, the IQE PIC selects one of the queues to transmit from. Therefore, it is important to understand how different logical interface parameters are configured or derived (not explicitly configured), and also how the same values are established at the queue level.

In the following examples, assume that the bandwidth available at the physical interface level is 400 Mbps and there are four logical interfaces (units) configured. A per-unit scheduler is configured, so the logical interfaces operate in different modes depending on the parameters configured.

If no class-of-service parameters are configured on any of the logical interfaces, the interface is in default mode. In default mode, the guaranteed rate (CIR) available at the physical interface (400 Mbps) is divided equally among the four logical interfaces. Each of the four gets a guaranteed rate (CIR) of 100 Mbps. Because none of the four logical interfaces have a shaping rate (PIR) configured, each logical interface can transmit up to the maximum of the entire 400 Mbps. Because there is no excess rate configured on any of the logical interfaces, each of the four gets an equal, minimum excess weight of 1. The configured and hardware-derived bandwidths for this default mode example are shown in [Table 78 on page 926](#).

**Table 78: Default Mode Example for IQE PICs**

| Logical Interface | Configured            |                    |             | Hardware        |              |               |
|-------------------|-----------------------|--------------------|-------------|-----------------|--------------|---------------|
|                   | Guaranteed rate (CIR) | Shaping Rate (PIR) | Excess Rate | Guaranteed Rate | Shaping Rate | Excess Weight |
| Unit 0            | NA                    | NA                 | NA          | 100 Mbps        | 400 Mbps     | 1             |



**Table 78: Default Mode Example for IQE PICs (Continued)**

| Logical Interface | Configured            |                    |             | Hardware        |              |               |
|-------------------|-----------------------|--------------------|-------------|-----------------|--------------|---------------|
|                   | Guaranteed rate (CIR) | Shaping Rate (PIR) | Excess Rate | Guaranteed Rate | Shaping Rate | Excess Weight |
| Unit 1            | NA                    | NA                 | NA          | 100 Mbps        | 400 Mbps     | 1             |
| Unit 2            | NA                    | NA                 | NA          | 100 Mbps        | 400 Mbps     | 1             |
| Unit 3            | NA                    | NA                 | NA          | 100 Mbps        | 400 Mbps     | 1             |

If a subset of the logical interfaces (units) have a shaping rate (PIR) configured, but none of them have a guaranteed rate (CIR) or excess rate, then the physical interface is in PIR mode. Furthermore, if the sum of the shaping rates on the logical interfaces is less than or equal to the physical interface bandwidth, the physical interface is in undersubscribed PIR mode. If the sum of the shaping rates on the logical interfaces is more than the physical interface bandwidth, the physical interface is in oversubscribed PIR mode. These modes are the same as on other PICs, where only a shaping rate and guaranteed rate can be configured.

In undersubscribed PIR mode, the logical interfaces with a configured shaping rate receive preferential treatment over those without a configured shaping rate. For logical interfaces with a shaping rate configured, the guaranteed rate is set to the shaping rate. For the logical interfaces without a shaping rate, the remaining logical interface bandwidth is distributed equally among them. Excess weights for the logical interfaces with a shaping rate are set to an implementation-dependent value proportional to the shaping rate. Excess weights for the logical interfaces without a shaping rate are set to the minimum weight (1). However, although the excess weights for the configured logical interfaces are never used because the logical interfaces cannot transmit above their guaranteed rates, the excess weights are still determined for consistency with oversubscribed mode. Also, logical interfaces without a configured shaping rate can transmit up to a maximum of the physical bandwidth of the other queues that are not transmitting. Therefore, the shaping rate (PIR) is set to the physical interface bandwidth on these interfaces.

The configured and hardware-derived bandwidths for the undersubscribed PIR mode example are shown in [Table 79 on page 928](#). Note that the sum of the shaping rates configured on the logical interfaces (500 Mbps) is more than the physical interface bandwidth (400 Mbps).



**Table 79: Undersubscribed PIR Mode Example for IQE PICs**

| Logical Interface | Configured            |                    |             | Hardware        |              |               |
|-------------------|-----------------------|--------------------|-------------|-----------------|--------------|---------------|
|                   | Guaranteed rate (CIR) | Shaping Rate (PIR) | Excess Rate | Guaranteed Rate | Shaping Rate | Excess Weight |
| Unit 0            | NA                    | 100 Mbps           | NA          | 100 Mbps        | 100 Mbps     | 127           |
| Unit 1            | NA                    | 200 Mbps           | NA          | 200 Mbps        | 200 Mbps     | 63            |
| Unit 2            | NA                    | NA                 | NA          | 50 Mbps         | 400 Mbps     | 1             |
| Unit 3            | NA                    | NA                 | NA          | 50 Mbps         | 400 Mbps     | 1             |

In the oversubscribed PIR mode, where the sum of the configured shaping rates on the logical interfaces exceeds the physical interface bandwidth, we cannot set the guaranteed rate to the shaping rate because this might result in the sum of the guaranteed rates exceeding the physical interface bandwidth, which is not possible. In this mode, we want the logical interfaces with shaping rates configured to share the traffic proportionally when these logical interfaces are transmitting at full capacity. This could not happen if the guaranteed rate was set to the shaping rate. Instead, in hardware, we set the guaranteed rates to a “scaled down” shaping rate, so that the sum of the guaranteed rates of the logical interfaces do not exceed the physical interface bandwidth. Because there is no remaining bandwidth once this is done, the other logical interfaces receive a guaranteed rate of 0. Excess weights are set proportionally to the shaping rates and for logical interfaces without a shaping rate, the excess weight is set to a minimum value (1). Finally, the shaping rate is set to the shaping rate configured on the logical interface or to the physical interface bandwidth otherwise.



**NOTE:** When the sum of shaping rate at a logical interface is greater than the interface's bandwidth and a rate limit is applied to one of the logical interface queues, the bandwidth limit for the queue is based on a scaled down logical interface shaping rate value rather than the configured logical interface shaping rate.

The configured and hardware-derived bandwidths for the oversubscribed PIR mode example are shown in [Table 80 on page 929](#). Note that the sum of the shaping rates configured on the logical interfaces (300 Mbps) is less than the physical interface bandwidth (400 Mbps).



**Table 80: Oversubscribed PIR Mode Example for IQE PICs**

| Logical Interface | Configured            |                    |             | Hardware        |              |               |
|-------------------|-----------------------|--------------------|-------------|-----------------|--------------|---------------|
|                   | Guaranteed rate (CIR) | Shaping Rate (PIR) | Excess Rate | Guaranteed Rate | Shaping Rate | Excess Weight |
| Unit 0            | NA                    | 100 Mbps           | NA          | 80 Mbps         | 100 Mbps     | 50            |
| Unit 1            | NA                    | 150 Mbps           | NA          | 120 Mbps        | 150 Mbps     | 76            |
| Unit 2            | NA                    | 250 Mbps           | NA          | 200 Mbps        | 250 Mbps     | 127           |
| Unit 3            | NA                    | NA                 | NA          | 0 Mbps          | 400 Mbps     | 1             |

If none of the logical interfaces have an excess rate configured, but at least one of the logical interfaces has a guaranteed rate (CIR) configured, then the physical interface is in CIR mode. In this case, the guaranteed rates are set in hardware to the configured guaranteed rate on the logical interface. For logical interfaces that do not have a guaranteed rate configured, the guaranteed rate is set to 0. The hardware shaping rate is set to the value configured on the logical interface or to the full physical interface bandwidth otherwise. The excess weight is calculated proportional to the configured guaranteed rates. Logical interfaces without a configured guaranteed rate receive a minimum excess weight of 1.

The configured and hardware-derived bandwidths for the CIR mode example are shown in [Table 81 on page 929](#). In CIR mode, the shaping rates are ignored in the excess weight calculations. So although logical unit 1 has an explicitly configured PIR and logical unit 3 does not, they both receive the minimum excess weight of 1.

**Table 81: CIR Mode Example for IQE PICs**

| Logical Interface | Configured            |                    |             | Hardware        |              |               |
|-------------------|-----------------------|--------------------|-------------|-----------------|--------------|---------------|
|                   | Guaranteed rate (CIR) | Shaping Rate (PIR) | Excess Rate | Guaranteed Rate | Shaping Rate | Excess Weight |
| Unit 0            | 50 Mbps               | 100 Mbps           | NA          | 50 Mbps         | 100 Mbps     | 127           |



Table 81: CIR Mode Example for IQE PICs (Continued)

| Logical Interface | Configured            |                    |             | Hardware        |              |               |
|-------------------|-----------------------|--------------------|-------------|-----------------|--------------|---------------|
|                   | Guaranteed rate (CIR) | Shaping Rate (PIR) | Excess Rate | Guaranteed Rate | Shaping Rate | Excess Weight |
| Unit 1            | NA                    | 150 Mbps           | NA          | 0 Mbps          | 150 Mbps     | 1             |
| Unit 2            | 100 Mbps              | NA                 | NA          | 100 Mbps        | 400 Mbps     | 63            |
| Unit 3            | NA                    | NA                 | NA          | 0 Mbps          | 400 Mbps     | 1             |

If one of the logical interfaces has an excess rate configured, then the physical interface is in excess rate mode. Strictly speaking, this mode only matters for the calculation of excess weights on the logical interface. The hardware guaranteed and shaping rates are determined as described previously. In excess rate mode, the excess weights are set to a value based on the configured excess rate. Logical interfaces which do not have excess rates configured receive a minimum excess weight of 1.



**NOTE:** Because the excess rate only makes sense above the guaranteed rate, you cannot configure an excess rate in PIR mode (PIR mode has only shaping rates configured). You must configure at least one guaranteed rate (CIR) on a logical interface to configure an excess rate.

The excess rate is configured as a percentage in the range from 1 through 100. The configured value is used to determine the excess weight in the range from 1 through 127.

The configured and hardware-derived bandwidths for the excess rate mode example are shown in [Table 82 on page 931](#). When an excess rate is configured on one or more logical interfaces, the shaping rate and the guaranteed rate are both ignored in the excess weight calculations. So logical unit 2 gets a minimum excess weight of 1, even though it has a guaranteed rate configured.



**Table 82: Excess Rate Mode Example for IQE PICs**

| Logical Interface | Configured            |                    |             | Hardware        |              |               |
|-------------------|-----------------------|--------------------|-------------|-----------------|--------------|---------------|
|                   | Guaranteed rate (CIR) | Shaping Rate (PIR) | Excess Rate | Guaranteed Rate | Shaping Rate | Excess Weight |
| Unit 0            | 50 Mbps               | 100 Mbps           | 20%         | 50 Mbps         | 100 Mbps     | 50            |
| Unit 1            | NA                    | 150 Mbps           | 50%         | 0 Mbps          | 150 Mbps     | 127           |
| Unit 2            | 100 Mbps              | NA                 | NA          | 100 Mbps        | 400 Mbps     | 1             |
| Unit 3            | NA                    | NA                 | 50%         | 0 Mbps          | 400 Mbps     | 127           |

## Default Rates for Queues on IQE PICs

The IQE PIC operates at the queue level as well as at the logical unit level. This section discusses how the IQE PIC derives hardware values from the user configuration parameters. First, the default behavior without explicit configuration is investigated, along with the rules used to derive hardware parameters from the scheduler map configuration of the transmit rate, shaping rate, and excess rate. For more information about configuring schedulers and scheduler maps, see ["How Schedulers Define Output Queue Properties" on page 332](#).

When you do not configure any CoS parameters, a default scheduler map is used to establish four queues: best-effort, expedited-forwarding, assured-forwarding, and network-control. Each queue has the default transmit rate, shaping rate, and excess rate shown in [Table 83 on page 931](#).

**Table 83: Default Queue Rates on the IQE PIC**

| Queue                     | Transmit Rate | Shaping Rate | Excess Rate |
|---------------------------|---------------|--------------|-------------|
| best-effort (Q0)          | 95%           | 100%         | 95%         |
| expedited-forwarding (Q1) | 0%            | 100%         | 0%          |
| assured-forwarding (Q2)   | 0%            | 100%         | 0%          |



**Table 83: Default Queue Rates on the IQE PIC (Continued)**

| Queue                | Transmit Rate | Shaping Rate | Excess Rate |
|----------------------|---------------|--------------|-------------|
| network-control (Q3) | 5%            | 100%         | 5%          |

When you configure a scheduler map to change the defaults, the IQE PIC hardware derives the values for each of the three major parameters: transmit rate, shaping rate, and excess rate.

The transmit rate is determined as follows:

- If a transmit rate is configured, then:
  - If the transmit rate is configured as an absolute bandwidth value, the configured value is used by the hardware.
  - If the transmit rate is configured as a percentage, then the percentage is used to calculate an absolute value used by the hardware, based on the guaranteed rate (CIR) configured at the logical interface or physical interface level. The CIR itself can be a default, configured, or derived value.
  - If the transmit rate is configured as a remainder, then the remaining value of the logical interface (unit) guaranteed rate (CIR) is divided equally among the queues configured as remainder.
- If a transmit rate is not configured, then the default transmit rate is derived based on remainder (for backward compatibility).
- If an excess rate is configured on any of the queues in a scheduler map, then the transmit rate on the queue is set to 0.

The shaping rate is determined as follows:

- If a shaping rate is configured:
  - If the shaping rate is configured as an absolute bandwidth value, the configured value is used by the hardware.
  - If the shaping rate is configured as a percentage, then the percentage is used to calculate an absolute value used by the hardware, based on the guaranteed rate (CIR) configured at the logical interface or physical interface level. Although it seems odd to base a shaping rate (PIR) on the CIR instead of a PIR, this is done so the shaping rate can be derived on the same basis as the transmit rate.
- If a shaping rate is not configured, then the default shaping rate is set to the shaping rate configured at the logical interface or physical interface level.

The excess rate is determined as follows:



- If an excess rate is configured on a queue, the value is used to derive an excess weight used by the IQE PIC hardware. The excess weight determines the proportional share of the excess bandwidth for which each queue can contend. The excess rate can be:
  - Percentage in the range from 1 through 100. This value is scaled to a hardware excess weight. Excess rates can add up to more than 100% for all queues under a logical or physical interface.
- If an excess rate is not configured on a queue, then the default excess rate is one of the following:
  - If a transmit rate is configured on any of the queues, then the excess weight is proportional to the transmit rates. Queues that do not have a transmit rate configured receive a minimum weight of 1.
  - If a transmit rate is not configured on any of the queues, but some queues have a shaping rate, then the excess weight is proportional to the shaping rates. Queues that do not have a shaping rate configured receive a minimum weight of 1.
  - If no parameters are configured on a queue, then the queue receives a minimum weight of 1.

### Sample Calculations of Excess Bandwidth Sharing on IQE PICs

The following four examples show calculations for the PIR mode. In PIR mode, the transmit rate and shaping rate calculations are based on the shaping rate of the logical interface. All calculations assume that one logical interface (unit) is configured with a shaping rate (PIR) of 10 Mbps and a scheduler map with four queues.

The first example has only a shaping rate (PIR) configured on the queues, as shown in [Table 84 on page 933](#).

**Table 84: PIR Mode with No Excess Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | NA                  | 80%                | NA          | 10 Mbps               |
| Q1    | NA                  | 50%                | NA          | 1 Mbps                |
| Q2    | NA                  | 40%                | NA          | 0 Mbps                |
| Q3    | NA                  | 30%                | NA          | 5 Mbps                |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 85 on page 934](#).



**Table 85: PIR Mode with No Excess Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 2.5 Mbps      | 8.0 Mbps     | 50            | 6 Mbps               |
| Q1    | 2.5 Mbps      | 5.0 Mbps     | 31            | 1 Mbps               |
| Q2    | 2.5 Mbps      | 4.0 Mbps     | 25            | 0 Mbps               |
| Q3    | 2.5 Mbps      | 3.0 Mbps     | 19            | 3 Mbps               |

In this first example, all four queues are initially serviced round-robin. Because there are no transmit rates configured on any of the queues, they receive a default “remainder” transmit rate of 2.5 Mbps per queue. But because there are shaping rates configured, the excess weights are calculated based on the shaping rates. For the traffic sent to each queue, Queue 0 and Queue 3 get their transmit rates of 2.5 Mbps and Queue 1 gets 1 Mbps. The remaining 4 Mbps is excess bandwidth and is divided between Queue 0 and Queue 3 in the ratio of the shaping rates (80/30). So Queue 3 expects an excess bandwidth of  $4 \text{ Mbps} * (30\% / (80\% + 30\%)) = 1.09 \text{ Mbps}$ . However, because the shaping rate on Queue 3 is 3 Mbps, Queue 3 can transmit only 3 Mbps and Queue 0 receives the remaining excess bandwidth and can transmit at 6 Mbps.

Note that if there were equal transmit rates explicitly configured, such as 2.5 Mbps for each queue, the excess bandwidth would be split based on the transmit rate (equal in this case), as long as the result in below the shaping rate for the queue.

The second example has a shaping rate (PIR) and transmit rate (CIR) configured on the queues, as shown in [Table 86 on page 934](#).

**Table 86: PIR Mode with Transmit Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 50%                 | 80%                | NA          | 10 Mbps               |
| Q1    | 40%                 | 50%                | NA          | 5 Mbps                |
| Q2    | 10%                 | 20%                | NA          | 5 Mbps                |



**Table 86: PIR Mode with Transmit Rate Configuration (Continued)**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q3    | NA                  | 5%                 | NA          | 1 Mbps                |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 87 on page 935](#).

**Table 87: PIR Mode with Transmit Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 5.0 Mbps      | 8.0 Mbps     | 63            | 5 Mbps               |
| Q1    | 4.0 Mbps      | 5.0 Mbps     | 50            | 4 Mbps               |
| Q2    | 1.0 Mbps      | 2.0 Mbps     | 12            | 1 Mbps               |
| Q3    | 0.0 Mbps      | 0.5 Mbps     | 1             | 0.0 Mbps             |

In this second example, because the transmit rates are less than the shaping rates, each queue receives its transmit rate.

The third example also has a shaping rate (PIR) and transmit rate (CIR) configured on the queues, as shown in [Table 88 on page 935](#).

**Table 88: Second PIR Mode with Transmit Rate Configuration Example**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 50%                 | 80%                | NA          | 10 Mbps               |
| Q1    | 40%                 | 50%                | NA          | 5 Mbps                |
| Q2    | 5%                  | 20%                | NA          | 0 Mbps                |
| Q3    | NA                  | 5%                 | NA          | 1 Mbps                |



The way that the IQE PIC hardware interprets these parameters is shown in [Table 89 on page 936](#).

**Table 89: Second PIR Mode with Transmit Rate Hardware Behavior Example**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 5.0 Mbps      | 8.0 Mbps     | 66            | 5.27 Mbps            |
| Q1    | 4.0 Mbps      | 5.0 Mbps     | 53            | 4.23 Mbps            |
| Q2    | 0.5 Mbps      | 2.0 Mbps     | 13            | 0.0 Mbps             |
| Q3    | 0.5 Mbps      | 0.5 Mbps     | 1             | 0.5 Mbps             |

In this third example, all four queues are initially serviced round-robin. However, Queue 2 has no traffic sent to its queue. So Queue 0, Queue 1, and Queue 3 all get their respective transmit rates, a total of 9.5 Mbps. The remaining 0.5 Mbps is used by Queue 3, because the transmit rate is the same as the shaping rate. Once this traffic is sent, Queue 0 and Queue 1 share the excess bandwidth in the ratio of their transmit rates, which total 9 Mbps. In this case, Queue 0 = 5 Mbps + (0.5 Mbps \* 5/9) = 5.27 Mbps. Queue 1 = 4 Mbps + (0.5 Mbps \* 4/9) = 4.23 Mbps.

The fourth example has a shaping rate (PIR), transmit rate (CIR), and excess rate configured on the queues, as shown in [Table 90 on page 936](#).

**Table 90: PIR Mode with Transmit Rate and Excess Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 30%                 | 80%                | 50%         | 10 Mbps               |
| Q1    | 25%                 | 50%                | 10%         | 5 Mbps                |
| Q2    | 10%                 | 20%                | 30%         | 0 Mbps                |
| Q3    | 5%                  | 5%                 | NA          | 1 Mbps                |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 91 on page 937](#).



**Table 91: PIR Mode with Transmit Rate and Excess Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 3.0 Mbps      | 8.0 Mbps     | 70            | 6.33 Mbps            |
| Q1    | 2.5 Mbps      | 5.0 Mbps     | 14            | 3.17 Mbps            |
| Q2    | 1.0 Mbps      | 2.0 Mbps     | 42            | 0.0 Mbps             |
| Q3    | 0.5 Mbps      | 0.5 Mbps     | 1             | 0.5 Mbps             |

In this fourth example, all four queues are initially serviced round-robin. Queue 3 gets 0.5 Mbps of guaranteed bandwidth but cannot transmit more because the shaping rate is the same. Queue 2 has no traffic to worry about at all. Queue 0 and Queue 1 get the respective transmit rates of 3.0 Mbps and 2.5 Mbps. The excess bandwidth of 4 Mbps is divided between Queue 0 and Queue 1 in the ratio on their excess rates. So Queue 1 gets 2.5 Mbps (the guaranteed rate) + 4 Mbps (the excess) +  $(10\% / (50\% + 10\%)) = 3.17$  Mbps. Queue 0 gets the rest, for a total of 6.33 Mbps.

You can configure only an excess rate on the queues, as shown in [Table 92 on page 937](#).

**Table 92: Excess Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | NA                  | NA                 | 50%         | 10 Mbps               |
| Q1    | NA                  | NA                 | 40%         | 10 Mbps               |
| Q2    | NA                  | NA                 | 30%         | 10 Mbps               |
| Q3    | NA                  | NA                 | 20%         | 10 Mbps               |

The way that the IQE PIC hardware interprets these excess rate parameters is shown in [Table 93 on page 938](#).



**Table 93: Excess Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 0 Mbps        | 10.0 Mbps    | 45            | 3.57 Mbps            |
| Q1    | 0 Mbps        | 10.0 Mbps    | 40            | 2.86 Mbps            |
| Q2    | 0 Mbps        | 10.0 Mbps    | 30            | 2.14 Mbps            |
| Q3    | 0 Mbps        | 10.0 Mbps    | 20            | 1.43 Mbps            |

In this excess rate example, there are no transmit or shaping rates configured on any of the queues, only excess rates, so bandwidth division happens only on the basis of the excess rates. Note that all the transmit (guaranteed) rates are set to 0. Usually, when there are no excess rates configured, the queue transmit rate is calculated by default. But when there is an excess rate configured on any of the queues, the transmit rate is set to 0. The excess bandwidth (all bandwidths in this case) is shared in the ratio of the excess weights. So Queue 0 receives  $10 \text{ Mbps} * (50 / (50 + 40 + 30 + 20)) = 3.57 \text{ Mbps}$ .

It is possible to configure rate limits that result in error conditions. For example, consider the configuration shown in [Table 94 on page 938](#).

**Table 94: PIR Mode Generating Error Condition**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | NA                  | 80%                | NA          | 10 Mbps               |
| Q1    | NA                  | 50%                | NA          | 5 Mbps                |
| Q2    | NA                  | 20%                | NA          | 5 Mbps                |
| Q3    | NA                  | 5%                 | NA          | 1 Mbps                |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 95 on page 939](#).



**Table 95: PIR Mode Generating Error Condition Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 2.5 Mbps      | 8.0 Mbps     | 818           | 4.03 Mbps            |
| Q1    | 2.5 Mbps      | 5.0 Mbps     | 511           | 3.47 Mbps            |
| Q2    | 2.5 Mbps      | 2.0 Mbps     | 255           | 2 Mbps               |
| Q3    | 2.5 Mbps      | 0.5 Mbps     | 51            | 0.1 Mbps             |

In the error example, note that the shaping rates calculated on Queue 2 and Queue 3 are less than the transmit rates on those queues (2.0 Mbps and 0.5 Mbps are each less than 2.5 Mbps). This is an error condition and results in a syslog error message.

The following set of five examples involve the IQE PIC operating in CIR mode. In CIR mode, the transmit rate and shaping rate calculations are based on the transmit rate of the logical interface. All calculations assume that the logical interface has a shaping rate (PIR) of 20 Mbps and a transmit rate (CIR) of 10 Mbps. The scheduler map has four queues.

The first example has only a shaping rate (PIR) with no excess rate configured on the queues, as shown in [Table 96 on page 939](#).

**Table 96: CIR Mode with No Excess Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | NA                  | 80%                | NA          | 10 Mbps               |
| Q1    | NA                  | 70%                | NA          | 10 Mbps               |
| Q2    | NA                  | 40%                | NA          | 10 Mbps               |
| Q3    | NA                  | 30%                | NA          | 10 Mbps               |





**NOTE:** The transmit rate (CIR) of 10 Mbps is configured on the logical interface (unit) not the queues in the scheduler map. This is why the queue transmit rates are labeled NA.

The way that the IQE PIC hardware interprets these parameters is shown in [Table 97 on page 940](#).

**Table 97: CIR Mode with No Excess Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 2.5 Mbps      | 8.0 Mbps     | 50            | 6.76 Mbps            |
| Q1    | 2.5 Mbps      | 7.0 Mbps     | 31            | 6.23 Mbps            |
| Q2    | 2.5 Mbps      | 4.0 Mbps     | 25            | 4.0 Mbps             |
| Q3    | 2.5 Mbps      | 3.0 Mbps     | 19            | 3.0 Mbps             |

In this first example, all four queues split the 10-Mbps transmit rate equally and each get a transmit rate of 2.5 Mbps. However, the shaping rate on the interface is 20 Mbps. The 10-Mbps excess bandwidth is divided among the queues in the ratio of their shaping rates. But Queue 2 and Queue 3 are shaped at 3.0 and 4.0 Mbps, respectively, so they cannot use more bandwidth and get those rates. This accounts for 2 Mbps (the 7 Mbps shaped bandwidth minus the 5 Mbps guaranteed bandwidth for Queue 2 and Queue 3) of the 10-Mbps excess, leaving 8 Mbps for Queue 0 and Queue 1. So Queue 0 and Queue 1 share the 8-Mbps excess bandwidth in the ratio of their shaping rates, which total 15 Mbps. In this case, Queue 0 =  $8.0 \text{ Mbps} \times 8/15 = 4.26 \text{ Mbps}$ , for a total of  $2.5 \text{ Mbps} + 4.26 \text{ Mbps} = 6.76 \text{ Mbps}$ . Queue 1 =  $8.0 \text{ Mbps} \times 7/15 = 3.73 \text{ Mbps}$ , for a total of  $2.5 \text{ Mbps} + 3.73 \text{ Mbps} = 6.23 \text{ Mbps}$ .

The second example has only a few shaping rates (PIR) with no excess rate configured on the queues, as shown in [Table 98 on page 940](#).

**Table 98: CIR Mode with Some Shaping Rates and No Excess Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | NA                  | 80%                | NA          | 10 Mbps               |
| Q1    | NA                  | 50%                | NA          | 5 Mbps                |



**Table 98: CIR Mode with Some Shaping Rates and No Excess Rate Configuration (Continued)**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q2    | NA                  | NA                 | NA          | 10 Mbps               |
| Q3    | NA                  | NA                 | NA          | 1 Mbps                |



**NOTE:** If a configuration results in the calculated transmit rate of the queue exceeding the shaping rate of the queue, an error message is generated. For example, setting the shaping rate on Queue 2 and Queue 3 in the above example to 20 percent and 5 percent, respectively, generates an error message because the calculated transmit rate for these queues (2.5 Mbps) is more than their calculated shaping rates (2.0 Mbps and 0.5 Mbps).

The way that the IQE PIC hardware interprets these parameters is shown in [Table 99 on page 941](#).

**Table 99: CIR Mode with Some Shaping Rates and No Excess Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 2.5 Mbps      | 8.0 Mbps     | 78            | 8.0 Mbps             |
| Q1    | 2.5 Mbps      | 5.0 Mbps     | 48            | 5.0 Mbps             |
| Q2    | 2.5 Mbps      | 20 Mbps      | 1             | 6.0 Mbps             |
| Q3    | 2.5 Mbps      | 20 Mbps      | 1             | 1.0 Mbps             |

In this second example, all four queues split the 10-Mbps transmit rate equally and each get a transmit rate of 2.5 Mbps. Because of their configured queue shaping rates, Queue 0 and Queue 1 receive preference over Queue 2 and Queue 3 for the excess bandwidth. Queue 0 (8.0 Mbps) and Queue 1 (5.0 Mbps) account for 13 Mbps of the 20 Mbps shaping rate on the logical interface. The remaining 7 Mbps is divided equally between Queue 2 and Queue 3. However, because Queue 3 only has 1 Mbps to send, Queue 2 uses the remaining 6 Mbps.

The third example has shaping rates (PIR) and transmit rates with no excess rate configured on the queues, as shown in [Table 100 on page 942](#).



**Table 100: CIR Mode with Shaping Rates and Transmit Rates and No Excess Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 50%                 | 80%                | NA          | 10 Mbps               |
| Q1    | 40%                 | 50%                | NA          | 5 Mbps                |
| Q2    | 10%                 | 20%                | NA          | 5 Mbps                |
| Q3    | NA                  | 10%                | NA          | 1 Mbps                |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 101 on page 942](#).

**Table 101: CIR Mode with Shaping Rates and Transmit Rates and No Excess Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 5.0 Mbps      | 8.0 Mbps     | 63            | 8.0 Mbps             |
| Q1    | 4.0 Mbps      | 5.0 Mbps     | 50            | 5.0 Mbps             |
| Q2    | 1.0 Mbps      | 2.0 Mbps     | 12            | 2.0 Mbps             |
| Q3    | 0.0 Mbps      | 0.5 Mbps     | 1             | 0.5 Mbps             |

In this third example, the first three queues get their configured transmit rates and are serviced in round-robin fashion. This adds up to 10 Mbps, leaving a 10-Mbps excess from the logical interface shaping rate of 20 Mbps. The excess is shared in the ratio of the transmit rates, or 5:4:1:0. Therefore, Queue 0 receives  $5 \text{ Mbps} + (5 * 10/10) = 10 \text{ Mbps}$ . This value is greater than the 8 Mbps shaping rate on Queue 0, so Queue 0 is limited to 8 Mbps. Queue 1 receives  $4 \text{ Mbps} + (4 * 10/10) = 8 \text{ Mbps}$ . This value is greater than the 5 Mbps shaping rate on Queue 1, so Queue 1 is limited to 5 Mbps. Queue 2 receives  $1 \text{ Mbps} + (1 * 10/10) = 2 \text{ Mbps}$ . This value is equal to the 2 Mbps shaping rate on Queue 2, so Queue 2 receives 2 Mbps. This still leaves 5 Mbps excess bandwidth, which can be used by Queue 3. Note that in this example bandwidth usage never reaches the shaping rate configured on the logical interface (20 Mbps).

The fourth example has shaping rates (PIR) and transmit rates with no excess rate configured on the queues. However, in this case the sum of the shaping rate percentages configured on the queues



multiplied by the transmit rate configured on the logical interface is greater than the shaping rate configured on the logical interface. The configuration is shown in [Table 102 on page 943](#).

**Table 102: CIR Mode with Shaping Rates Greater Than Logical Interface Shaping Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 50%                 | 80%                | NA          | 10 Mbps               |
| Q1    | 40%                 | 70%                | NA          | 10 Mbps               |
| Q2    | 10%                 | 50%                | NA          | 10 Mbps               |
| Q3    | NA                  | 50%                | NA          | 10 Mbps               |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 103 on page 943](#).

**Table 103: CIR Mode with Shaping Rates Greater Than Logical Interface Shaping Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 5.0 Mbps      | 8.0 Mbps     | 63            | 8.0 Mbps             |
| Q1    | 4.0 Mbps      | 7.0 Mbps     | 50            | 7.0 Mbps             |
| Q2    | 1.0 Mbps      | 5.0 Mbps     | 12            | 5.0 Mbps             |
| Q3    | 0.0 Mbps      | 5.0 Mbps     | 1             | 0.0 Mbps             |

In this fourth example, the first three queues get their configured transmit rates and are serviced in round-robin fashion. This adds up to 10 Mbps, leaving a 10-Mbps excess from the logical interface shaping rate of 20 Mbps. The excess is shared in the ratio of the transmit rates, or 5:4:1:0. Therefore, Queue 0 receives  $5 \text{ Mbps} + (5 * 10/10) = 10 \text{ Mbps}$ . This value is greater than the 8 Mbps shaping rate on Queue 0, so Queue 0 is limited to 8 Mbps. Queue 1 receives  $4 \text{ Mbps} + (4 * 10/10) = 8 \text{ Mbps}$ . This value is greater than the 7 Mbps shaping rate on Queue 1, so Queue 1 is limited to 7 Mbps. Queue 2 receives  $1 \text{ Mbps} + (1 * 10/10) = 2 \text{ Mbps}$ . This value is less than the 5 Mbps shaping rate on Queue 2, so Queue 2 receives 2 Mbps. This still leaves 3 Mbps excess bandwidth, which can be used by Queue 2 (below its shaping rate) and Queue 3 (also below its shaping rate) in the ratio 1:0 (because of the



transmit rate configuration). But 1:0 means Queue 3 cannot use this bandwidth, and Queue 2 utilizes  $2 \text{ Mbps} + (3 \text{ Mbps} * 1/1) = 5 \text{ Mbps}$ . This is equal to the shaping rate of 5 Mbps, so Queue 2 receives 5 Mbps.

The fifth example has excess rates and transmit rates, but no shaping rates (PIR) configured on the queues. The configuration is shown in [Table 104 on page 944](#).

**Table 104: CIR Mode with Excess Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 30%                 | NA                 | 50%         | 10 Mbps               |
| Q1    | 25%                 | NA                 | 10%         | 10 Mbps               |
| Q2    | NA                  | NA                 | 30%         | 10 Mbps               |
| Q3    | 10%                 | NA                 | NA          | 10 Mbps               |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 105 on page 944](#).

**Table 105: CIR Mode with Excess Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 3.0 Mbps      | 20 Mbps      | 70            | 10.5 Mbps            |
| Q1    | 2.5 Mbps      | 20 Mbps      | 14            | 4.0 Mbps             |
| Q2    | 0.0 Mbps      | 20 Mbps      | 42            | 4.5 Mbps             |
| Q3    | 1.0 Mbps      | 20 Mbps      | 1             | 1.0 Mbps             |

In this fifth example, Queue 2 does not have a transmit rate configured. If there were no excess rates configured, then Queue 2 would get a transmit rate equal to the remainder of the bandwidth (3.5 Mbps in this case). However, because there is an excess rate configured on some of the queues on this logical interface, the transmit rate for Queue 2 is set to 0 Mbps. The others queues get their transmit rates and there leaves 13.5 Mbps of excess bandwidth. This bandwidth is divided among Queue 0, Queue 1, and



Queue 3 in the ratio of their excess rates. So Queue 0, for example, gets  $3.0 \text{ Mbps} + 13.5 \text{ Mbps} * (50 / (50 + 10 + 30)) = 10.5 \text{ Mbps}$ .

Four other examples calculating expected traffic distribution are of interest. The first case has three variations, so there are six more examples in all.

- Oversubscribed PIR mode at the logical interface with transmit rates, shaping rates, and excess rates configured at the queues (this example has three variations).
- CIR mode at the logical interface (a non-intuitive case is used).
- Excess priority configured.
- Default excess priority used.

The first three examples all concern oversubscribed PIR mode at the logical interface with transmit rates, shaping rates, and excess rates configured at the queues. They all use a configuration with a physical interface having a shaping rate of 40 Mbps. The physical interface has two logical units configured, logical unit 1 and logical unit 2, with a shaping rate of 30 Mbps and 20 Mbps, respectively. Because the sum of the logical interface shaping rates is more than the shaping rate on the physical interface, the physical interface is in oversubscribed PIR mode. The CIRs (transmit rates) are set to the scaled values of 24 Mbps and 16 Mbps, respectively.

Assume that logical unit 1 has 40 Mbps of traffic to be sent. The traffic is capped at 30 Mbps because of the shaping rate of 30 Mbps. Because the CIR is scaled down to 24 Mbps, the remaining 6 Mbps (30 Mbps – 24 Mbps) qualifies as excess bandwidth.

The following three examples consider different parameters configured in a scheduler map and the expected traffic distributions that result.

The first example uses oversubscribed PIR mode with only transmit rates configured on the queues. The configuration is shown in [Table 106 on page 945](#).

**Table 106: Oversubscribed PIR Mode with Transmit Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 40%                 | NA                 | NA          | 15 Mbps               |
| Q1    | 30%                 | NA                 | NA          | 10 Mbps               |
| Q2    | 25%                 | NA                 | NA          | 10 Mbps               |



**Table 106: Oversubscribed PIR Mode with Transmit Rate Configuration (Continued)**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q3    | 5%                  | NA                 | NA          | 5 Mbps                |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 107 on page 946](#).

**Table 107: Oversubscribed PIR Mode with Transmit Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 9.6 Mbps      | 30 Mbps      | 50            | 12 Mbps              |
| Q1    | 7.2 Mbps      | 30 Mbps      | 38            | 9 Mbps               |
| Q2    | 6.0 Mbps      | 30 Mbps      | 31            | 7.5 Mbps             |
| Q3    | 1.2 Mbps      | 30 Mbps      | 6             | 1.5 Mbps             |

The first example has hardware queue transmit rates based on the parent (logical interface unit 1) transmit rate (CIR) value of 24 Mbps. Because there are no excess rates configured, the excess weights are determined by the transmit rates. Therefore, both the logical interface CIR and excess bandwidth are divided in the ratio of the transmit rates. This is essentially the same as the undersubscribed PIR mode and the traffic distribution should be the same. The only difference is that the result is achieved as a combination of guaranteed rate (CIR) and excess rate sharing.

The second example also uses oversubscribed PIR mode, but this time with only excess rate configured on the queues. In other words, the same ratios are established with excess rate percentages instead of transmit rate percentages. In this case, when excess rates are configured, queues without a specific transmit rate are set to 0 Mbps. So the entire bandwidth qualifies as excess at the queue level and the bandwidth distribution is based on the configured excess rates. The expected output rate results are exactly the same as in the first example, except the calculation is based on different parameters.

The third example also uses oversubscribed PIR mode, but with both transmit rates and excess rates configured on the queues. The configuration is shown in [Table 108 on page 947](#).



**Table 108: Oversubscribed PIR Mode with Transmit Rate and Excess Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 40%                 | NA                 | 50%         | 15 Mbps               |
| Q1    | 30%                 | NA                 | 50%         | 12 Mbps               |
| Q2    | 25%                 | NA                 | NA          | 8 Mbps                |
| Q3    | 5%                  | NA                 | NA          | 5 Mbps                |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 109 on page 947](#).

**Table 109: Oversubscribed PIR Mode with Transmit Rate and Excess Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 9.6 Mbps      | 30 Mbps      | 63            | 12.6 Mbps            |
| Q1    | 7.2 Mbps      | 30 Mbps      | 63            | 10.2 Mbps            |
| Q2    | 6.0 Mbps      | 30 Mbps      | 1             | 6.0 Mbps             |
| Q3    | 1.2 Mbps      | 30 Mbps      | 1             | 1.2 Mbps             |

The third example has the configured queue transmit rate (CIR) divided according to the ratio of the transmit rates based on the logical interface unit 1 CIR of 25 Mbps. The rest of the excess bandwidth divided according the ratio of the excess rates. The excess 6-Mbps bandwidth is divided equally between Queue 0 and Queue 1 because the excess rates are both configured at 50%. This type of configuration is not recommended, however, because the CIR on the logical interface is a system-derived value based on the PIRs of the other logical units and the traffic distribution at the queue level is based on this value and, therefore, not under direct user control. We recommend that you either configure excess rates without transmit rates at the queue level when in PIR mode, or also define a CIR at the logical interface if you want to configure a combination of transmit rates and excess rates at the queue level. That is, you should use configurations of the CIR mode with excess rates types.

The fourth example uses CIR mode at the logical interface. For this example, assume that a physical interface is configured with a 40-Mbps shaping rate and logical interfaces unit 1 and unit 2. Logical



interface unit 1 has a PIR of 30 Mbps and logical interface unit 2 has a PIR of 20 Mbps and a CIR of 10 Mbps. The configuration at the queue level of logical interface unit 1 is shown in [Table 110 on page 948](#).

**Table 110: CIR Mode with Transmit Rate and Excess Rate Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 40%                 | NA                 | 50%         | 15 Mbps               |
| Q1    | 30%                 | NA                 | 50%         | 12 Mbps               |
| Q2    | 25%                 | NA                 | NA          | 8 Mbps                |
| Q3    | 5%                  | NA                 | NA          | 5 Mbps                |

The way that the IQE PIC hardware interprets these parameters is shown in [Table 111 on page 948](#).

**Table 111: CIR Mode with Transmit Rate and Excess Rate Hardware Behavior**

| Queue | Transmit Rate | Shaping Rate | Excess Weight | Expected Output Rate |
|-------|---------------|--------------|---------------|----------------------|
| Q0    | 0 Mbps        | 30 Mbps      | 63            | 15 Mbps              |
| Q1    | 0 Mbps        | 30 Mbps      | 63            | 12 Mbps              |
| Q2    | 0 Mbps        | 30 Mbps      | 1             | 1.5 Mbps             |
| Q3    | 0 Mbps        | 30 Mbps      | 1             | 1.5 Mbps             |

The fourth example might be expected to divide the 40 Mbps of traffic between the two logical units in the ratio of the configured transmit rates. But note that because the logical interfaces are in CIR mode, and logical interface unit 1 does not have a CIR configured, the hardware CIR is set to 0 Mbps at the queue level. Bandwidth distribution happens based only on the excess weights. So Queue 0 and Queue 1 get to transmit up to 15 Mbps and 12 Mbps, respectively, while the remaining 3 Mbps is divided equally by Queue 2 and Queue 3.





**NOTE:** We recommend configuring a CIR value explicitly for the logical interface if you are configuring transmit rates and excess rates for the queues.

The fifth example associates an excess priority with the queues. Priorities are associated with every queue and propagated to the parent node (logical or physical interface). That is, when the scheduler picks a logical interface, the scheduler considers the logical interface priority as the priority of the highest priority queue under that logical interface. On the IQE PIC, you can configure an excess priority for every queue. The excess priority can differ from the priority used for guaranteed traffic and applies only to traffic in the excess region. The IQE PIC has three “regular” priorities and two excess priorities (high and low, which is the default). The excess priorities are lower than the regular priorities. For more information about configuring excess bandwidth sharing and priorities, see ["Configuring Excess Bandwidth Sharing on IQE PICs" on page 959](#).

Consider a logical interface configured with a shaping rate of 10 Mbps and a guaranteed rate of 10 Mbps. At the queue level, parameters are configured as shown in [Table 112 on page 949](#).

**Table 112: Excess Priority Configuration**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Excess Rate | Traffic Sent To Queue |
|-------|---------------------|--------------------|-------------|-----------------------|
| Q0    | 40%                 | NA                 | 50%         | 10 Mbps               |
| Q1    | 30%                 | NA                 | 50%         | 10 Mbps               |
| Q2    | 25%                 | NA                 | NA          | 0 Mbps                |
| Q3    | 5%                  | NA                 | NA          | 1 Mbps                |

In this fifth example, Queue 0 is configured with an excess priority of `high` and all other queues have the default excess priority (`low`). Because there is no traffic on Queue 2, there is an excess bandwidth of 2.5 Mbps. Because Queue 0 has a higher excess priority, Queue 0 gets the entire excess bandwidth. So the expected output rates on the queues are 4 Mbps+ 2.5 Mbps= 6.5 Mbps for Queue 0, 3 Mbps for Queue 1, 0 Mbps for Queue 2, and 0.5 Mbps for Queue 3. Note that this behavior is different than regular priorities. With regular priorities, the transmission is still governed by transmit rates and the priority controls only the order in which the packets are picked up by the scheduler. So without excess configuration, if Queue 0 had a regular priority of `high` and there was 10 Mbps of traffic on all four queues, the traffic distribution would be 4 Mbps for Queue 0, 3 Mbps for Queue 1, 2.5 Mbps for Queue 2, and 0.5 Mbps for Queue 3 instead of giving all 10 Mbps to Queue 0. Excess priority traffic distributions are governed first by the excess priority and then by the excess rates. Also note that in this



example, although the queues are in the excess region because they are transmitting above their configured transmit rates, the logical interface is still within its guaranteed rate. So at the logical interface level, the priority of the queues get promoted to a regular priority and this priority is used by the scheduler at the logical interface level.

The sixth and final example considers the effects of the default excess priority. When the excess priority for a queue is not configured explicitly, the excess priority is based on the regular priority. A regular priority of high maps to an excess priority of high. All other regular priorities map to an excess priority of low. When there is no regular priority configured, the regular and excess priorities are both set to low.

## Configuring the Junos OS to Support Eight Queues on IQ Interfaces for T Series and M320 Routers

By default, IQ PICs on T Series and M320 routers are restricted to a maximum of four egress queues per interface. To configure a maximum of eight egress queues on IQ interfaces, include the `max-queues-per-interface` statement at the `[edit chassis fpc slot-number pic pic-number]` hierarchy level:

```
[edit chassis fpc slot-number pic pic-number]
max-queues-per-interface (8 | 4);
```

On a TX Matrix or TX Matrix Plus router, include the `max-queues-per-interface` statement at the `[edit chassis lcc number fpc slot-number pic pic-number]` hierarchy level:

```
[edit chassis lcc number fpc slot-number pic pic-number]
max-queues-per-interface (8 | 4);
```



**NOTE:** The configuration at the `[edit class-of-service]` hierarchy level must also 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.

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.



**NOTE:** Starting from Junos OS Release 14.1R8, 14.2R6, 15.1F6, 15.1R3, 15.1R4, and 16.1R1, the restricted queue PICs without the `max-queues-per-interface` configuration boot up with a maximum of eight queues per port and two operational ports (port 0 and 2). PICs with restricted queues include Quad T3 PIC, Quad E3 PIC, 4-port SONET/SDH OC3c/STM1 PIC, and 4-Port OC3 and 1-port OC12 PICs with SFP.

When you include the `max-queues-per-interface` statement and commit the configuration, all physical interfaces on the IQ PIC are deleted and readded. 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.

## RELATED DOCUMENTATION

[Configuring the Junos OS to Support ILMI for Cell Relay Encapsulation on an ATM2 IQ PIC](#)

[Configuring the Junos OS to Enable Larger Delay Buffers for T1, E1, and DS0 Interfaces Configured on Channelized IQ PICs](#)

## BA Classifiers and ToS Translation Tables

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

On Juniper Networks M40e, M120, M320 Multiservice Edge Routers, and T Series Core Routers with Enhanced IQ (IQE) PICs, or on any router or switch with IQ2 or Enhanced IQ2 (IQ2E) PICs, you can replace the type-of-service (ToS) bit value on the incoming packet header on a logical interface with a user-defined value. The new ToS value is used for all class-of-service processing and is applied before any other class-of-service or firewall treatment of the packet. The PIC uses the `translation-table` statement to determine the new ToS bit values.



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

## RELATED DOCUMENTATION

[Configuring ToS Translation Tables](#) | 952

## Configuring ToS Translation Tables

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

On M40e, M120, M320, and T Series routers with IQE PICs, or on any device with IQ2 or Enhanced IQ2 PICs, you can replace the ToS bit value on the incoming packet header on a logical interface with a user-defined value. The new ToS value is used for all class-of-service processing and is applied before any other class-of-service or firewall treatment of the packet. On the IQE PIC, the values configured with the `translation-table` statement determines the new ToS bit values.

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

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

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



On the IQE PIC, incoming ToS bit translation is subject to the following rules:

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



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

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

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

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



```
}
}
```

ToS translation on the IQE PIC is a form of behavior aggregate (BA) classification. The IQE PIC does not support multifield classification of packets at the PIC level.

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

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

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

1. Create and configure the translation table.

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

2. Apply the translation table to the logical interface input on the Enhanced IQ PIC

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

3. Verify the configuration.



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

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

```
Translation Table: dscp-trans-table, Translation table type: dscp-to-dscp, Index: 6761
```

| From Code point | To Code Point |
|-----------------|---------------|
| 000000          | 000111        |
| 000001          | 000111        |
| 000010          | 000111        |
| 000011          | 000111        |
| 000100          | 000111        |
| 000101          | 000111        |
| 000110          | 000111        |
| 000111          | 111000        |
| 001000          | 000111        |
| 001001          | 000111        |
| 001010          | 000111        |
| 001011          | 000111        |
| 001100          | 000111        |
| 001101          | 000111        |
| 001110          | 000111        |
| 001111          | 000111        |
| 010000          | 000111        |
| 010001          | 000111        |
| 010010          | 000111        |
| 010011          | 000111        |
| 010100          | 000111        |
| 010101          | 000111        |
| 010110          | 000111        |
| 010111          | 000111        |
| 011000          | 000111        |
| 011001          | 000111        |
| 011010          | 000111        |
| 011011          | 000111        |
| 011100          | 000111        |
| 011101          | 000111        |
| 011110          | 000111        |
| 011111          | 000111        |
| 100000          | 000111        |
| 100001          | 000111        |
| 100010          | 000111        |



|        |        |
|--------|--------|
| 100011 | 000111 |
| 100100 | 000111 |
| 100101 | 000111 |
| 100110 | 000111 |
| 100111 | 111000 |
| 101000 | 000111 |
| 101001 | 000111 |
| 101010 | 000111 |
| 101011 | 000111 |
| 101100 | 000111 |
| 101101 | 000111 |
| 101110 | 000111 |
| 101111 | 000111 |
| 110000 | 000111 |
| 110001 | 000111 |
| 110010 | 000111 |
| 110011 | 000111 |
| 110100 | 000111 |
| 110101 | 000111 |
| 110110 | 000111 |
| 110111 | 000111 |
| 111000 | 000111 |
| 111001 | 000111 |
| 111010 | 000111 |
| 111011 | 000111 |
| 111100 | 000111 |
| 111101 | 000111 |
| 111110 | 000001 |
| 111111 | 000000 |

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

```

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

Logical interface: so-2/3/0.0, Index: 68
  Object          Name          Type          Index

```



|                   |                 |                |       |
|-------------------|-----------------|----------------|-------|
| Rewrite           | exp-default     | exp (mpls-any) | 29    |
| Classifier        | dscp-default    | dscp           | 7     |
| Classifier        | exp-default     | exp            | 10    |
| Translation Table | exp-trans-table | EXP_TO_EXP     | 61925 |

#### 4. Save the configuration.

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

## RELATED DOCUMENTATION

[Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 44](#)

[Overview of Assigning Service Levels to Packets Based on Multiple Packet Header Fields | 132](#)

## Configuring Hierarchical Layer 2 Policers on IQE PICs

The IQE PIC can police traffic at Layer 2 in a hierarchical manner. *Policing* is the practice of making sure that different streams of incoming traffic conform to certain parameters and limits. If the incoming traffic exceeds the established boundaries, that traffic can be marked or even ignored, depending on configuration. Hierarchical policing maintains two rates: an aggregate rate and a high-priority rate. The traffic is marked differently depending on service class (currently, the classes are expedited forwarding and nonexpedited forwarding). The expedited traffic has an additional rate configured, the guaranteed rate (CIR), which is only marked above that limit. If there is no expedited traffic present, then the non-expedited traffic is able to use the aggregate bandwidth rate before being marked with a packet loss priority. When expedited traffic is present, it is marked above the guaranteed rate, but also uses bandwidth from the nonexpedited range.

For example, consider an aggregate rate of 10 Mbps and a high-priority rate of 2 Mbps of a Fast Ethernet interface. The guaranteed rate is also set at 2 Mbps for expedited forwarding traffic. If there is no expedited traffic present, then nonexpedited traffic can use up to 10 Mbps before being marked. When expedited forwarding traffic is present, the expedited traffic is guaranteed 2 Mbps (of the 10 Mbps) without being marked, but is marked above the 2 Mbps limit. In this case, the nonexpedited forwarding traffic can use the remaining 8 Mbps before being marked.

Layer 2 policers configured on IQE PICs have the following limitations:

- Only one kind of policer is supported on a physical or logical interface. For example, a hierarchical or two- or three-color policer in the same direction on the same logical interface is not supported.



- Applying policers to both physical port and logical interface (policer chaining) is not supported.
- If there is no behavior aggregate classification, there is a limit of 64 policers per interface. (Usually, there will be a single policer per DLCI in frame relay and other logical interface types.)
- The policer should be independent of behavior aggregate classification. (Without a behavior aggregate, all traffic is treated as either expedited or non-expedited forwarding, depending on configuration.)
- With a behavior aggregate, traffic not matching any classification bits (such as DSCP or EXP) is policed as nonexpedited forwarding traffic.
- Only two levels of traffic policing are supported: aggregate and premium.

To configure Layer 2 policing on the IQE PIC, for each forwarding class:

1. Enable configuration of the forwarding classes.

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

2. Define the forwarding classes.

```
set class fc1 queue-num 0 priority high policing-priority premium
set class fc2 queue-num 1 priority low policing-priority normal
set class fc3 queue-num 2 priority low policing-priority normal
set class fc4 queue-num 3 priority low policing-priority normal
```

3. Configure the hierarchical policer.

- a. Enable configuration of the hierarchical policer.

```
[edit]
user@host# edit firewall hierarchical-policer hier_example1
```

- b. Configure the aggregate policer.

```
[edit firewall hierarchical-policer hier_example1 ]
user@host# set aggregate if-exceeding bandwidth-limit 70m burst-size-limit 1800
user@host# set aggregate then discard
```



- c. Configure the premium policer.

```
[edit firewall hierarchical-policer hier_example1 ]
user@host# set premium if-exceeding bandwidth-limit 70m burst-size-limit 3600
user@host# set premium then discard
```

4. Apply the policer to the logical on the IQE PIC.

```
[edit]
user@host# edit interfaces so-6/0/0 unit 0
user@host# set layer2-policer input-hierarchical-policer hier_example1
user@host# set family inet address 10.0.22.1/30
user@host# set family iso
user@host# set family mpls
```

Alternatively, to hierarchically rate-limit Layer 2 ingress traffic for all protocol families and for all logical interfaces configured on physical interface so-6/0/0, you could reference the policer from the physical interface configuration.

## RELATED DOCUMENTATION

*Controlling Network Access Using Traffic Policing Overview*

[Configuring a Custom Forwarding Class for Each Queue | 281](#)

## Configuring Excess Bandwidth Sharing on IQE PICs

The IQE PIC gives users more control over excess bandwidth sharing. You can set a shaping rate and a guaranteed rate on a queue or logical interface and control the excess bandwidth (if any) that can be used after all bandwidth guarantees have been satisfied. This section discusses the following topics related to excess bandwidth sharing on the IQE PIC:

On some types of PICs, including the IQ and IQ2, and Enhanced Queuing DPCs, you can configure either a committed information rate (CIR) using the `guaranteed-rate` statement or a peak information rate (PIR) using the `shaping-rate` statement. You can configure both a PIR and CIR, and in most cases the CIR is less than the value of PIR. For bursty traffic, the CIR represents the average rate of traffic per unit time and the PIR represents the maximum amount of traffic that can be transmitted in a given interval. In other words, the PIR (`shaping-rate`) establishes the maximum bandwidth available. The CIR (`guaranteed-rate`) establishes the minimum bandwidth available if all sources are active at the same time. Theoretically, the



PIR or CIR can be established at the queue, logical interface, or physical interface level. In this section, the PIRs or CIRs apply at the queue or logical interface (or both) levels.



**NOTE:** You can configure a shaping rate at the physical interface, logical interface, or queue level. You can configure a guaranteed rate or excess rate only at the logical interface and queue level.

Once all of the bandwidth guarantees (the sum of the CIRs at that level) are met, there could still be some excess bandwidth available for use. In existing PICs, you have no control over how this excess bandwidth is used. For example, consider the situation shown in [Table 113 on page 960](#) regarding a 10-Mbps physical interface. This example assumes that all queues are of the same priority. Also, if you do not specify a priority for the excess bandwidth, the excess priority is the same as the normal priority.

**Table 113: Default Handling of Excess Traffic**

| Queue | Transmit Rate (CIR) | Shaping Rate (PIR) | Traffic Rate | Guaranteed Rate (Total = 6 Mbps) | Maximum Rate | Excess Bandwidth (Part of 4 Mbps Excess) | Expected Transmit Rate (Guarantee + Excess) |
|-------|---------------------|--------------------|--------------|----------------------------------|--------------|------------------------------------------|---------------------------------------------|
| Q0    | 10%                 | 80%                | 10 Mbps      | 1 Mbps                           | 8 Mbps       | 0.73 Mbps                                | 1.73 Mbps                                   |
| Q1    | 20%                 | 50%                | 10 Mbps      | 2 Mbps                           | 5 Mbps       | 1.45 Mbps                                | 3.45 Mbps                                   |
| Q2    | 5%                  | 5%                 | 10 Mbps      | 0.5 Mbps                         | 0.5 Mbps     | 0 Mbps                                   | 0.5 Mbps                                    |
| Q3    | 25%                 | NA ("100%")        | 10 Mbps      | 2.5 Mbps                         | 10 Mbps      | 1.82 Mbps                                | 4.32 Mbps                                   |

A 10-Mbps interface (the Traffic Rate column) has four queues, and the guaranteed rates are shown as percentages (Transmit Rate column) and in bits per second (Guaranteed Rate column). The table also shows the shaping rate (PIR) as a percentage (Shaping Rate column) and the actual maximum possible transmitted rate (Traffic Rate column) on the oversubscribed interface. Note the guaranteed rates (CIRs) add up to 60 percent of the physical port speed or 6 Mbps. This means that there are 4 Mbps of "excess" bandwidth that can be used by the queues. This excess bandwidth is used as shown in the last two columns. One column (the Excess Bandwidth column) shows the bandwidth partitioned to each queue as a part of the 4-Mbps excess. The excess 4 Mbps bandwidth is shared in the ratio of the transmit rate (CIR) percentages of 10, 20, 5, and 25, adjusted for granularity. The last column shows the



transmit rate the users can expect: the sum of the guaranteed rate plus the proportion of the excess bandwidth assigned to the queue.

Note that on PICs other than the IQE PICs the user has no control over the partitioning of the excess bandwidth. Excess bandwidth partitioning is automatic, simply assuming that the distribution and priorities of the excess bandwidth should be the same as the distribution and priorities of the other traffic. However, this might not always be the case and the user might want more control over excess bandwidth usage.

For more information on how excess bandwidth sharing is handled on the Enhanced Queuing DPC, see ["Configuring Excess Bandwidth Sharing" on page 1081](#).

On PICs other than IQE PICs, you can limit a queue's transmission rate by including the `transmit-rate` statement with the `exact` option at the `[edit class-of-service schedulers scheduler-name]` hierarchy level. However, on the IQE PIC, you can set a shaping rate independent of the transmit rate by including the `shaping-rate` statement at the `[edit class-of-service schedulers scheduler-name]` hierarchy level. Also, other PICs share excess bandwidth (bandwidth left over once the guaranteed transmit rate is met) in an automatic, nonconfigurable fashion. You cannot configure the priority of the queues for the excess traffic on other PICs either.

To share excess bandwidth on IQE PICs, include the `excess-rate` statement along with the `guaranteed-rate` statement (to define the CIR) and the `shaping-rate` statement (to define the PIR):

```
[edit class-of-service traffic-control-profile profile-name]
[edit class-of-service schedulers scheduler-name]
excess-rate percent percentage;
guaranteed-rate (percent percentage | rate);
shaping-rate (percent percentage | rate);
```

To apply these limits to a logical interface, configure the statements at the `[edit class-of-service traffic-control-profile profile-name]` hierarchy level. To apply these limits to a specific queue, configure the statements at the `[edit class-of-service schedulers scheduler-name]` hierarchy level. You must also complete the configuration by applying the scheduler map or traffic control profile correctly.

You configure the excess rate as a percentage from 1 through 100. By default, excess bandwidth is automatically distributed as on other PIC types.

You can also configure a high or low priority for excess bandwidth by including the `excess-priority` statement with the `high` or `low` option at the `[edit class-of-service schedulers scheduler-name]` hierarchy level.



This statement establishes the priority at the queue level, which then applies also at the logical and physical interface levels.

```
[edit class-of-service schedulers scheduler-name]
excess-priority (high | low);
```



**NOTE:** You cannot configure an excess rate for a logical interface if there is no guaranteed rate configured on any logical interface belonging to the physical interface.

The following example configures excess bandwidth sharing on logical interfaces of an IQE PIC by using twotraffic control profile:

1. Create the first traffic control profile called: for-unit-0-percent and specify the associated parameters for sharing the excess bandwidth.

- a. Specify a name for the traffic control profile to create it.

```
[edit]
user@host$ edit class-of-service traffic-control-profiles for-unit-0-percent
```

- b. Configure the maximum usage rate.

```
[edit class-of-service traffic-control-profiles for-unit-0-percent]
user@host$ set shaping-rate 10k
```

- c. Specify the guaranteed-rate (to define the CIR)

```
[edit class-of-service traffic-control-profiles for-unit-0-percent]
user@host$ guaranteed-rate 1k
```

- d. Specify the excess-rate

```
[edit class-of-service traffic-control-profiles for-unit-0-percent]
user@host$ excess-rate percent 30
```

2. Create the second traffic control profile called: for-unit-1-proportion and specify the associated parameters for sharing the excess bandwidth.



- a. Specify a name for the traffic control profile to create it.

```
[edit]
user@host$ edit class-of-service traffic-control-profiles for-unit-1-proportion
```

- b. Configure the maximum usage rate.

```
[edit class-of-service traffic-control-profiles for-unit-1-proportion]
user@host$ set shaping-rate 5m
```

- c. Specify the percentage or proportion of excess bandwidth traffic to share.

```
[edit class-of-service traffic-control-profiles for-unit-1-proportion]
user@host$ excess-rate percent 30
```

- d. Specify the priority for excess bandwidth.

3. Verify the configuration.

```
user@host> show class-of-service traffic-control-profile
for-unit-0-percent {
    shaping-rate 10k;
    guaranteed-rate 1k;
    excess-rate percent 30;
}
for-unit-1-proportion {
    shaping-rate 20k;
    guaranteed-rate 10k;
    excess-rate percent 35;
}
```

The following example configures the excess rate in a scheduler:

1. Create the first scheduler called: scheduler-for-excess-low and specify the associated parameters for sharing the excess bandwidth.



- a. Specify a name for the scheduler to create it.

```
[edit]
user@host$ edit class-of-service schedulers scheduler-for-excess-low
```

- b. Specify the transmit rate for the scheduler.

```
[edit class-of-service schedulers scheduler-for-excess-low]
user@host$ set transmit-rate 1m
```

- c. Configure the maximum usage rate.

```
[edit class-of-service schedulers scheduler-for-excess-low]
user@host$ set shaping-rate 5m
```

- d. Specify the percentage or proportion of excess bandwidth traffic to share.

```
[edit class-of-service schedulers scheduler-for-excess-low]
user@host$ excess-rate percent 30
```

- e. Specify the priority of excess bandwidth traffic on the scheduler.

```
[edit class-of-service schedulers scheduler-for-excess-low]
user@host$ excess-priority low
```

2. Create the second scheduler called: scheduler-for-excess-high and specify the associated parameters for sharing the excess bandwidth.

- a. Specify a name for the traffic control profile to create it.

```
[edit]
user@host$ edit class-of-service schedulers scheduler-for-excess-high
```



- b. Specify the transmit rate for the scheduler.

```
[edit class-of-service schedulers scheduler-for-excess-low]
user@host$ set transmit-rate percent 20
```

- c. Configure the maximum usage rate.

```
[edit class-of-service schedulers scheduler-for-excess-high]
user@host$ set shaping-rate percent 30
```

- d. Specify the percentage or proportion of excess bandwidth traffic to share.

```
[edit class-of-service schedulers scheduler-for-excess-high]
user@host$ excess-rate percent 25
```

- e. Specify the priority of excess bandwidth traffic on the scheduler.

```
[edit class-of-service schedulers scheduler-for-excess-high]
user@host$ excess-priority high
```

3. Verify the configuration.

```
user@host> show class-of-service schedulers
scheduler-for-excess-low {
    transmit-rate 1m;
    shaping-rate 5m;
    excess-rate percent 30;
    excess-priority low;
}
scheduler-for-excess-high {
    transmit-rate percent 20;
    shaping-rate percent 30;
    excess-rate percent 25;
    excess-priority high;
}
```





**NOTE:** All of these parameters apply to egress traffic only and only for per-unit schedulers. That is, there is no hierarchical or shared scheduler support.

## RELATED DOCUMENTATION

[Configuring Schedulers | 338](#)

[Excess Rate and Excess Priority Configuration Examples | 378](#)

## Configuring Rate-Limiting Policers for High Priority Low-Latency Queues on IQE PICs

You can rate-limit the strict-high and high queues on the IQE PIC. Without this limiting, traffic that requires low latency (delay) such as voice can block the transmission of medium-priority and low-priority packets. Unless limited, high and strict-high traffic is always sent before lower priority traffic, causing the lower priority queues to “starve” and cause timeouts and unnecessarily resent packets.

On the IQE PIC you can rate-limit queues before the packets are queued for output. All packets exceeding the configured rate limit are dropped, so care is required when establishing this limit. This model is also supported on IQ2 PICs and is the only way to perform egress policing on IQE PICs. This feature introduces no new configuration statements.

Although intended for low-latency traffic classes such as voice, the configuration allows any queue to be rate-limited. However, the configuration requires the rate-limited queue to have either a high or strict-high priority.



**NOTE:** You can configure a low-latency static policer for only one rate-limited queue per scheduler map. You can configure up to 1024 low-latency static policers.

This example limits the transmit rate of a strict-high expedited-forwarding queue to 1 Mbps. The scheduler and scheduler map are defined, and then applied to the traffic at the [edit interfaces] and [edit class-of-service] hierarchy levels:

1. Define the scheduler:



- a. Specify a name for the scheduler to create it.

```
[edit]
user@host# edit class-of-service schedulers scheduler-1
```

- b. Specify the transmit rate.

```
[edit class-of-service schedulers scheduler-1]
user@host# set transmit-rate 1m rate-limit
```

- c. Specify the priority of the scheduler.

```
[edit class-of-service schedulers scheduler-1]
user@host# set priority strict-high
```

## 2. Define the scheduler map:

- a. Specify a name for the scheduler map to create it.

```
[edit]
user@host# edit class-of-service scheduler-maps scheduler-map1
```

- b. Map the EF forwarding class to the scheduler.

```
[edit class-of-service scheduler-maps scheduler-map-1]
user@host# set forwarding-class expedited-forwarding scheduler scheduler-1
```

## 3. Configure the physical interface.

This example uses Frame Relay encapsulation and enables per-unit scheduling, which enables you to apply scheduling to the Frame Relay DLCI.

- a. Specify the physical interface of the interface.

```
[edit]
user@host# edit interfaces so-2/0/0
```



- b. Enable the association of scheduler map names with logical interfaces.

```
[edit interfaces s0-2/0/0]
user@host# set per-unit-scheduler
```

- c. Specify the encapsulation type.

```
[edit interfaces s0-2/0/0]
user@host# set encapsulation frame-relay
```

#### 4. Configure the logical interface and specify the Frame Relay DLCI.

- a. Specify the logical interface number.

```
[edit interfaces s0-2/0/0]
user@host# edit unit 0
```

- b. Specify the data-link connection identifier (DLCI).

```
[edit interfaces s0-2/0/0 unit 0]
user@host# set dlci 1
```

#### 5. Apply the scheduler map to the logical interface:

- a. Specify the physical and logical interface to which you want to apply the scheduler map.

```
[edit]
user@host# edit class-of-service interfaces so-2/0/0 unit 0
```

- b. Specify the name of the scheduler map you created.

```
[edit class-of-service interfaces so-2/0/0 unit 0]
user@host# set scheduler-map scheduler-map1
```

- c. Specify the amount of bandwidth to be allocated for the logical interface.

```
[edit class-of-service interfaces so-2/0/0 unit 0]
user@host# set shaping-rate 2m
```



6. You can issue the following operational mode commands to verify your configuration (the first shows the rate limit in effect):

- `show class-of-service scheduler-map scheduler-map scheduler-map-1`
- `show class-of-service interface so-2/0/0`

```
[edit class-of-service]
schedulers {
  scheduler-1 {
    transmit-rate 1m rate-limit;
    priority strict-high;
  }
}
scheduler-maps {
  scheduler-map1 {
    forwarding-class expedited-forwarding scheduler scheduler-1;
  }
}
[edit interfaces]
s0-2/0/0 {
  per-unit-scheduler;
  encapsulation frame-relay;
  unit 0 {
    dlci 1;
  }
}
[edit class-of-service]
interfaces {
  so-2/0/0 {
    unit 0 {
      scheduler-map scheduler-map1;
      shaping-rate 2m;
    }
  }
}
```

## RELATED DOCUMENTATION

[Configuring Schedulers | 338](#)

[Configuring Scheduler Maps | 338](#)



| `show class-of-service scheduler-map`

## Applying Scheduler Maps and Shaping Rate to Physical Interfaces on IQ PICs

### IN THIS SECTION

- [Examples: Applying a Shaping Rate | 971](#)
- [Examples: Applying a Scheduler Map and Shaping Rate to Physical Interfaces on IQ PICs | 976](#)

This topic describes how to configure and apply scheduler maps and shaping rates to physical interfaces on various types of IQ PICs.

You can specify a peak bandwidth rate (shaping rate) in bps, either as a complete decimal number or as a decimal number followed by the abbreviation *k* (1000), *m* (1,000,000), or *g* (1,000,000,000). For physical interfaces, the range is from 1000 through 6,400,000,000,000 bps. 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 logical interfaces, the range is 1000 through 32,000,000,000 bps.) The sum of the bandwidths you allocate to all physical interfaces on a PIC must not exceed the bandwidth of the PIC.



**NOTE:** For MX Series routers, the shaping rate value for the physical interface at the [edit class-of-service interfaces *interface-name*] hierarchy level must be a minimum of 160 Kbps.

If you configure a shaping rate that exceeds the physical interface bandwidth, the new configuration is ignored, and the previous configuration remains in effect. For example, if you configure a shaping rate that is 80 percent of the physical interface bandwidth, then change the configuration to 120 percent of the physical interface bandwidth, the 80 percent setting remains in effect. This holds true unless the PIC is restarted, in which case the default bandwidth goes into effect. As stated previously, the default bandwidth is based on the channel bandwidth and the time slot allocation.

Optionally, you can instead configure scheduling and rate shaping on logical interfaces, as described in ["Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs" on page 391](#). In general, logical and physical interface traffic shaping is mutually exclusive. You can include the `shaping-rate` statement at the [edit class-of-service interfaces *interface-name*] hierarchy level or the [edit class-of-service interfaces *interface-name* unit *logical-unit-number*] hierarchy level, but not both. For Gigabit Ethernet IQ2 and IQ2E



PICs, you can configure hierarchical traffic shaping, meaning the shaping is performed on both the physical interface and the logical interface. For more information, see ["Configuring Input Shaping Rates for Both Physical and Logical Interfaces" on page 428](#).

For more information, see the following sections:

Examples: Applying a Shaping Rate

IN THIS SECTION

Shaping Rate Calculations | 971


Example: Applying a Shaping Rate to a Clear-Channel T1 Interface on a Channelized T1 IQ PIC | 972

Example: Applying a Shaping Rate to a Clear-Channel E1 Interface on a Channelized E1 IQ PIC | 974

This topic shows you how to calculate shaping rates and provides two examples of how to configure and apply a shaping rate. It includes the following sections:

Shaping Rate Calculations

For shaping rate and WRR, the information included in the calculations varies by PIC type, as shown in [Table 114 on page 971](#).



**NOTE:** The 10-port 10-Gigabit Oversubscribed Ethernet (OSE) PICs and Gigabit Ethernet IQ2 PICs are unique in supporting ingress scheduling and shaping. The calculations shown for 10-port 10-Gigabit OSE and Gigabit Ethernet IQ2 PICs apply to both ingress and egress scheduling and shaping. For other PICs, the calculations apply to egress scheduling and shaping only.

For more information, see ["CoS on Enhanced IQ2 PICs Overview" on page 1005](#).

Table 114: Shaping Rate and WRR Calculations by PIC Type

| PIC Type                   | Platform              | Shaping Rate and WRR Calculations Include                                                                            |
|----------------------------|-----------------------|----------------------------------------------------------------------------------------------------------------------|
| 10-port 10-Gigabit OSE PIC | T Series Core Routers | For ingress and egress:<br><br>L3 header + L2 header + frame check sequence (FCS) + interpacket gap (IPG) + preamble |



**Table 114: Shaping Rate and WRR Calculations by PIC Type (Continued)**

| PIC Type                          | Platform | Shaping Rate and WRR Calculations Include                                         |
|-----------------------------------|----------|-----------------------------------------------------------------------------------|
| Gigabit Ethernet IQ2 PIC          | All      | For ingress and egress:<br><br>L3 header + L2 header + frame check sequence (FCS) |
| Gigabit Ethernet IQ PIC           | All      | L3 header + L2 header + FCS                                                       |
| IQ PIC with a SONET/SDH interface | All      | L3 header+ L2 header + FCS                                                        |

**Example: Applying a Shaping Rate to a Clear-Channel T1 Interface on a Channelized T1 IQ PIC**

To apply a shaping rate to a clear-channel T1 interface on a channelized T1 IQ PIC:

1. Configure the physical and logical interfaces.
  - a. Specify the physical interface to configure.

```
[edit]
user@host# edit interfaces ct1-2/1/0
```

- b. Configure the channelized T1 IQ PIC as unpartitioned, clear channel.

```
[edit interfaces ct1-2/1/0]
user@host# set no-partition interface-type t1
```

- c. Specify the logical interface to configure.

```
[edit interfaces]
user@host# edit t1-2/1/0 unit 0
```



- d. Specify the IPv4 family and IP address

```
[edit interfaces ]
user@host# set family inet address 10.40.1.1/30
```

2. Configure traffic shaping by specifying the amount of bandwidth to be allocated to the logical interface.

- a. Specify the interface to configure.

```
[edit]
user@host# edit class-of-service interfaces t1-2/1/0
```

- b. Specify the shaping rate for the interface.

```
[edit class-of-service interfaces t1-2/1/0]
user@host# set shaping-rate 3000
```

3. Verify the configuration.

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

ct1-2/1/0 {
    no-partition interface-type t1;
}
t1-2/1/0 {
    unit 0 {
        family inet {
            address 10.40.1.1/30;
        }
    }
}
```

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

t1-2/1/0 {
```



```
    shaping-rate 160k;
}
```

4. Save your configuration.

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

### Example: Applying a Shaping Rate to a Clear-Channel E1 Interface on a Channelized E1 IQ PIC

To apply a shaping rate to a clear-channel E1 interface on a channelized E1 IQ PIC:

1. Configure the physical and logical interfaces.
  - a. Specify the physical interface to configure.

```
[edit]
user@host# edit interfaces ce1-2/1/0
```

- b. Configure the channelized T1 IQ PIC as unpartitioned, clear channel.

```
[edit interfaces ce1-2/1/0]
user@host# set no-partition interface-type e1
```

- c. Specify the logical interface to configure.

```
[edit interfaces]
user@host# edit e1-2/1/0 unit 0
```

- d. Specify the IPv4 family and IP address

```
[edit interfaces ]
user@host# set family inet address 10.40.1.1/30
```

2. Configure traffic shaping by specifying the amount of bandwidth to be allocated to the logical interface.



- a. Specify the interface to configure.

```
[edit]
user@host# edit class-of-service interfaces e1-2/1/0
user@host#
```

- b. Specify the shaping rate for the interface.

```
[edit class-of-service interfaces e1-2/1/0]
user@host# set shaping-rate 4000
```

3. Verify the configuration.

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

ce1-2/1/0 {
    no-partition interface-type e1;
}
e1-2/1/0 unit 0 {
    unit 0 {
        family inet {
            address 10.40.1.1/30;
        }
    }
}
```

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

e1-2/1/0 {
    shaping-rate 160k;
}
```

4. Save your configuration.

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



## Examples: Applying a Scheduler Map and Shaping Rate to Physical Interfaces on IQ PICs

### IN THIS SECTION

- [Example: Applying a Scheduler Map and Shaping Rate to a DS0 Channel of a Channelized T1 Interface on a Channelized T1 IQ PIC | 976](#)
- [Example: Applying a Scheduler Map and Shaping Rate to DS0 Channels of a Channelized E1 Interface on a Channelized E1 IQ PIC | 978](#)
- [Applying a Scheduler Map and Shaping Rate to a Clear-Channel T3 Interface on a Channelized DS3 IQ PIC | 982](#)
- [Applying a Scheduler Map and Shaping Rate to Fractional T1 Interfaces on a Channelized DS3 IQ PIC | 984](#)
- [Applying a Scheduler Map and Shaping Rate to a DS0 Channel of a T1 Interface in a Channelized T3 Interface on a Channelized DS3 IQ PIC | 989](#)

The following sections provide examples of how to configure and apply a scheduler map and shaping rate to various physical interface types.

The following examples are included:

### Example: Applying a Scheduler Map and Shaping Rate to a DS0 Channel of a Channelized T1 Interface on a Channelized T1 IQ PIC

To apply a scheduler map and shaping rate to a clear-channel T1 interface on a channelized T1 IQ PIC:

For this procedure, you must also configure a scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).

1. Configure the physical interface.
  - a. Specify the physical interface to configure.

```
[edit]
user@host# edit interfaces ct1-0/0/9
```



- b. Configure the channelized T1 IQ PIC as unpartitioned, clear channel.

```
[edit interfaces ct1-2/1/0]
user@host# set partition 1 timeslots 1-2 interface-type ds
```

## 2. Configure the logical interface.

- a. Specify the logical interface (DS0) to configure.

```
[edit interfaces]
user@host# edit ds-0/0/9:1
```

- b. Disable the sending of keepalives on the interface.

```
[edit interfaces ds-0/0/0:1]
set no-keepalives
```

- c. Specify the IPv4 family and IP address for the logical interface.

```
[edit interfaces ds-0/0/0:1]
user@host# set unit 0 family inet address 10.10.1.1/30
```

## 3. Apply the scheduler map and shaping rate to the interface.



**NOTE:** Be sure you have previously configured the scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).

- a. Specify the interface to configure.

```
[edit]
user@host# edit class-of-service interfaces ds-0/0/9:1
```

- b. Specify the name of the scheduler map to apply to the interface.

```
[edit class-of-service interfaces ds-0/0/9:1]
user@host# set scheduler-map sched_port_1
```



- c. Specify the amount of bandwidth to allocate to the interface.

```
set shaping-rate 2000
```

4. Verify the configuration.

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

ct1-0/0/9 {
    partition 1 timeslots 1-2 interface-type ds;
}
ds-0/0/9:1 {
    no-keepalives;
    unit 0 {
        family inet {
            address 10.10.1.1/30;
        }
    }
}
```

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

ds-0/0/0:1 {
    scheduler-map sched_port_1;
    shaping-rate 160k;
}
```

5. Save your configuration.

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

#### Example: Applying a Scheduler Map and Shaping Rate to DS0 Channels of a Channelized E1 Interface on a Channelized E1 IQ PIC

To apply a scheduler map and shaping rate to a clear-channel E1 interface on a channelized E1 IQ PIC:



For this procedure, you must also configure a scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).

**1. Configure the physical.**

- a.** Specify the physical interface to configure.

```
[edit]
user@host# edit interfaces ce1-1/3/1
```

- b.** Configure the channelized E1 IQ PIC as unpartitioned, clear channel.

```
[edit interfaces ce1-1/3/1]
user@host# set partition 1 timeslots 1-4 interface-type ds
user@host# set partition 2 timeslots 5-6 interface-type ds
```

**2. Configure the first logical interface (DS0).**

- a.** Specify the logical interface to configure.

```
[edit interfaces]
user@host# edit ds-1/3/1:1
```

- b.** Disable the sending of keepalives on the interface.

```
[edit interfaces ds-1/3/1:1]
set no-keepalives
```

- c.** Specify the IPv4 family and IP address for the logical interface.

```
[edit interfaces ds-1/3/1:1]
user@host# set unit 0 family inet address 10.10.1.1/30
```

**3. Configure the second logical interface (DS0).**

- a.** Specify the logical interface to configure.

```
[edit interfaces]
user@host# edit ds-1/3/1:2
```



- b. Disable the sending of keepalives on the interface.

```
[edit interfaces ds-1/3/1:2]
set no-keepalives
```

- c. Specify the IPv4 family and IP address for the logical interface.

```
[edit interfaces ds-1/3/1:2]
user@host# set unit 0 family inet address 10.10.1.5/30
```

4. Apply the scheduler map and shaping rate to the first logical interface.



**NOTE:** Be sure you have previously configured the scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).

- a. Specify the logical interface to configure.

```
[edit]
user@host# edit class-of-service interfaces ds-1/3/1:1
```

- b. Specify the name of the scheduler map to apply to the logical interface.

```
[edit class-of-service interfaces ds-1/3/1:1]
user@host# set scheduler-map sched_port_1
```

- c. Specify the amount of bandwidth to allocate to the interface.

```
[edit class-of-service interfaces ds-1/3/1:1]
set shaping-rate 1000
```

5. Apply the scheduler map and shaping rate to the second logical interface.



**NOTE:** Be sure you have previously configured the scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).



- a. Specify the logical interface on which you want to apply the scheduler.

```
[edit]
user@host# edit class-of-service interfaces ds-1/3/1:2
```

- b. Specify the name of the scheduler map to apply to the interface.

```
[edit class-of-service interfaces ds-1/3/1:2]
user@host# set scheduler-map sched_port_1
```

- c. Specify the amount of bandwidth to allocate to the interface.

```
[edit class-of-service interfaces ds-1/3/1:2]
set shaping-rate 1500
```

6. Verify the configuration.

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

ce1-1/3/1 {
    partition 1 timeslots 1-4 interface-type ds;
    partition 2 timeslots 5-6 interface-type ds;
}
ds-1/3/1:1 {
    no-keepalives;
    unit 0 {
        family inet {
            address 10.10.1.1/30;
        }
    }
}
ds-1/3/1:2 {
    no-keepalives;
    unit 0 {
        family inet {
            address 10.10.1.5/30;
        }
    }
}
```



```
    }
}
```

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

interfaces {
  ds-1/3/1:1 {
    scheduler-map sched_port_1;
    shaping-rate 1000;
  }
  ds-1/3/1:2 {
    scheduler-map sched_port_1;
    shaping-rate 1500;
  }
}
```

7. Save your configuration.

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

### Applying a Scheduler Map and Shaping Rate to a Clear-Channel T3 Interface on a Channelized DS3 IQ PIC

To apply a scheduler map and shaping rate to a clear-channel T3 interface on a channelized DS3 IQ PIC:

For this procedure, you must also configure a scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).

1. Configure the channelized T3 IQ interface.
  - a. Specify the physical interface to configure.

```
[edit]
user@host# edit interfaces ct3-2/1/0
```



- b. Configure the interface as unpartitioned.

```
user@host# set no-partition
```

2. Configure the channelized DS3 interface.

- a. Specify the interface name.

```
[edit interface]
user@host# edit t3-2/1/0
```

- b. Specify the IPv4 family and IP address for the logical interface.

```
[edit interfaces t3-2/1/0]
user@host# set unit 0 family inet address 10.40.1.1/30
```

3. Apply the scheduler map and shaping rate on the channelized DS3 interface.



**NOTE:** Be sure you have previously configured the scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).

- a. Specify the interface name.

```
[edit]
user@host# edit class-of-service interfaces t3-2/1/0
```

- b. Specify the name of the scheduler map to apply to the interface.

```
[edit class-of-service interfaces t3-2/1/0]
user@host# set scheduler-map sched_port_1
```

- c. Specify the amount of bandwidth to allocate to the interface.

```
[edit class-of-service interfaces t3-2/1/0]
set shaping-rate 2500
```



#### 4. Verify the configuration.

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

ct3-2/1/0 {
  no-partition;
}
t3-2/1/0 {
  unit 0 {
    family inet {
      address 10.40.1.1/30;
    }
  }
}
```

```
[edit class-of-service]

user@ host# show
interfaces {
  t3-2/1/0 {
    shaping-rate 2500;
    unit 0 {
      scheduler-map sched_port_1;
    }
  }
}
```

#### 5. Save your configuration.

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

### Applying a Scheduler Map and Shaping Rate to Fractional T1 Interfaces on a Channelized DS3 IQ PIC

To apply a scheduler map and shaping rate to a fractional T1 interfaces on a channelized DS3 IQ PIC:

For this procedure, you must also configure a scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).



1. Configure the physical interface.
  - a. Specify the name of the physical interface.

```
[edit]  
user@host# edit interfaces ct3-1/1/3
```

- b. Configure the interface as a partitioned T1 interface.

```
[edit interfaces ct3-1/1/3]  
user@host# set partition 1-3 interface-type t1
```

2. Configure the first logical interface (T1 interface).
  - a. Specify the logical interface name.

```
[edit interfaces]  
user@host# edit t1-1/1/3:1
```

- b. Configure T1-specific physical interface properties.

```
[edit interfaces t1-1/1/3:1]  
set t1-options timeslots 1-2
```

- c. Specify the IPv4 family and IP address for the logical interface.

```
[edit interfaces t1-1/1/3:1]  
user@host# set unit 0 family inet address 10.10.1.1/30
```

3. Configure the second logical interface (T1 interface).
  - a. Specify the logical interface name.

```
[edit interfaces]  
user@host# edit t1-1/1/3:2
```



- b. Configure T1-specific physical interface properties.

```
[edit interfaces t1-1/1/3:2]  
set t1-options timeslots 3-6
```

- c. Specify the IPv4 family and IP address for the logical interface.

```
[edit interfaces t1-1/1/3:2]  
user@host# set unit 0 family inet address 10.10.1.5/30
```

4. Configure the third logical interface (T1 interface).

- a. Specify the logical interface name.

```
[edit interfaces]  
user@host# edit t1-1/1/3:3
```

- b. Configure T1-specific physical interface properties.

```
[edit interfaces t1-1/1/3:3]  
set t1-options timeslots 7-12
```

- c. Specify the IPv4 family and IP address for the logical interface.

```
[edit interfaces t1-1/1/3:3]  
user@host# set unit 0 family inet address 10.10.1.9/30
```

5. Apply the scheduler map and shaping rate to the T1 logical interfaces.



**NOTE:** Be sure you have previously configured the scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).



- a. Configure the first interface by specifying the interface name, applying the scheduler map to the interface, and specifying the amount of bandwidth to allocate to the interface.

```
[edit]
user@host# edit class-of-service
user@host# set interfaces t1-1/1/3:1 scheduler-map sched_port_1 shaping-rate 1200
```

- b. Configure the second interface by specifying the interface name, applying the scheduler map to the interface, and specifying the amount of bandwidth to allocate to the interface.

```
[edit class-of-service]
user@host# edit interfaces t1-1/1/3:2 scheduler-map sched_port_1 shaping-rate 1300
```

- c. Configure the third interface by specifying the interface name, applying the scheduler map to the interface, and specifying the amount of bandwidth to allocate to the interface.

```
[edit class-of-service]
user@host# edit interfaces t1-1/1/3:3 scheduler-map sched_port_1 shaping-rate 1400
```

## 6. Verify the configuration.

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

ct3-1/1/3 {
    partition 1-3 interface-type t1;
}
t1-1/1/3:1 {
    t1-options {
        timeslots 1-2;
    }
    unit 0 {
        family inet {
            address 10.10.1.5/30;
        }
    }
}
t1-1/1/3:2 {
    t1-options {
        timeslots 3-6;
```



```

    }
    unit 0 {
        family inet {
            address 10.10.1.5/30;
        }
    }
}
t1-1/1/3:3 {
    t1-options {
        timeslots 7-12;
    }
    unit 0 {
        family inet {
            address 10.10.1.9/30;
        }
    }
}
}

```

[edit class-of-service interfaces]

user@ host# **show**

```

t1-1/1/3:1 {
    scheduler-map sched_port_1;
    shaping-rate 1200;
}
t1-1/1/3:2 {
    scheduler-map sched_port_1;
    shaping-rate 1300;
}
t1-1/1/3:3 {
    scheduler-map sched_port_1;
    shaping-rate 1400;
}
}

```

## 7. Save your configuration.

[edit]

user@host# **commit**



## Applying a Scheduler Map and Shaping Rate to a DS0 Channel of a T1 Interface in a Channelized T3 Interface on a Channelized DS3 IQ PIC

To apply a scheduler map and shaping rate to a DS0 Channel of a T1 Interface in a Channelized T3 Channelized DS3 IQ PIC:

For this procedure, you must also configure a scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).

1. Configure the physical interface.
  - a. Specify the name of the physical interface.

```
[edit]
user@host# edit interfaces ct1-2/1/3
```

- b. Configure the interface as a partitioned, specify the timeslots for the interface, and specify the interface type.

```
[edit interfaces ct3-2/1/3]
user@host# set partition 1 timeslots 1-4 interface-type ds
```

2. Configure the DS0 interface.

```
[edit interfaces]
user@host# set ds-2/1/3:1:1 unit 0 family inet address 10.20.144.1/30
```

3. Apply the scheduler map and shaping rate to the logical interface.



**NOTE:** Be sure you have previously configured the scheduler map. For details on configuring the scheduler map, see ["Configuring Scheduler Maps" on page 338](#).

- a. Configure the interface by specifying the interface name, applying the scheduler map to the interface, and specifying the amount of bandwidth to allocate to the interface.

```
[edit class-of-service]
user@host# set interfaces ds-2/1/3:1:1 scheduler-map sched_port_1 shaping-rate 1100
```



#### 4. Verify the configuration.

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

ct3-2/1/3 {
  partition 1 interface-type ct1;
}
ct1-2/1/3:1 {
  partition 1 timeslots 1-4 interface-type ds;
}
ds-2/1/3:1:1 {
  unit 0 {
    family inet {
      address 10.20.144.1/30;
    }
  }
}
```

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

interfaces {
  ds-2/1/3:1:1 {
    scheduler-map sched_port_1;
    shaping-rate 1100;
  }
}
```

#### 5. Save your configuration.

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

## RELATED DOCUMENTATION

[Applying Scheduler Maps to Physical Interfaces | 340](#)

[Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs | 391](#)



## Applying Scheduler Maps to Chassis-Level Queues

### IN THIS SECTION

- [Applying Custom Schedulers to Packet Forwarding Component Queues | 993](#)
- [Examples: Scheduling Packet Forwarding Component Queues | 994](#)

On Intelligent Queuing (IQ) and Intelligent Queuing 2 (IQ2) interfaces, as well as on the 10x10GE MIC with SFP+, the traffic that is fed from the packet forwarding components into the PIC uses low packet loss priority (PLP) by default and is distributed evenly across the four chassis queues (not PIC queues), regardless of the scheduling configuration for each logical interface. This default behavior can cause traffic congestion.

The default chassis scheduler allocates resources for queue 0 through queue 3, with 25 percent of the bandwidth allocated to each queue. When you configure the chassis to use more than four queues, you must configure and apply a custom chassis scheduler to override the default chassis scheduler.

To apply a custom chassis scheduler:

1. Specify the interface on which to apply the scheduler.

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

For example:

```
[edit]
user@host# edit class-of-service interfaces so-0/1/*
```

2. Specify the name of the custom scheduler you want to apply to the interface.

```
[edit class-of-service interfaces so-0/1/0]
user@host# set scheduler-map-chassis map-name
```

To control the aggregated traffic transmitted from the chassis queues into the PIC, you can configure the chassis queues to derive their scheduling configuration from the associated logical interface's.



To configure the chassis queues to derive their scheduling from the associated logical interfaces:

1. Specify the logical interfaces from which to derive the scheduling configuration.

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

For example:

```
[edit]
user@host# edit class-of-service interfaces so-0/1/*
```

2. Specify that the scheduler configuration is derived from the specified logical interfaces.

```
[edit class-of-service interfaces so-0/1/0]
user@host# set scheduler-map-chassis derived
```



**CAUTION:** If you specify the `scheduler-map-chassis derived` statement in the configuration, packet loss might occur when you subsequently add or remove logical interfaces at the `[edit interfaces interface-name]` hierarchy level.

When fragmentation occurs on the egress interface, the first set of packet counters displayed in the output of the `show interfaces queue` command show the post-fragmentation values. The second set of packet counters (under the Packet Forwarding Engine Chassis Queues field) show the pre-fragmentation values. For more information about the `show interfaces queue` command, see the [CLI Explorer](#).

You can specify both the `scheduler-map` and the `scheduler-map-chassis derived` statements in the same interface configuration. The `scheduler-map` statement controls the scheduler inside the PIC, while the `scheduler-map-chassis derived` statement controls the aggregated traffic transmitted into the entire PIC.



**NOTE:** For the Gigabit Ethernet IQ PIC, you must specify both the `scheduler-map` and the `scheduler-map-chassis derived` statements in the interface configuration.

Generally, when you specify the `scheduler-map-chassis` statement in the configuration, you must use an interface wildcard for the interface name, as in `type-fpc/pic/*`. The wildcard must use this format—for example, `so-1/2/*`, which means all interfaces on FPC slot 1, PIC slot 2. There is one exception—you can apply the chassis scheduler map to a specific interface on the Gigabit Ethernet IQ PIC only.



According to Junos OS wildcard rules, specific interface configurations override wildcard configurations. For chassis scheduler map configurations, this rule does not apply; instead, specific interface CoS configurations are added to the chassis scheduler map configuration. For more information about how wildcards work with chassis scheduler maps, see ["Examples: Scheduling Packet Forwarding Component Queues" on page 994](#). For general information about wildcards, see the [Junos OS Administration Library for Routing Devices](#).



**NOTE:** The interface applies wildcard configuration only if you do not add any specific configuration. If you add the specific interface configuration, then the interface deletes the applied wildcard configuration and applies the specified configuration.

## Applying Custom Schedulers to Packet Forwarding Component Queues

Optionally, you can apply a custom scheduler to the chassis queues instead of configuring the chassis queues to automatically derive their scheduling configuration from the logical interfaces on the PIC.

To apply a custom chassis scheduler:

1. Specify the interface on which to apply the scheduler.

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

For example:

```
[edit]
user@host# edit class-of-service interfaces so-0/1/*
```

2. Specify the name of the custom scheduler you want to apply to the interface.

```
[edit class-of-service interfaces so-0/1/0]
user@host# set scheduler-map-chassis map-name
```

When you apply a custom scheduler map to packet forwarding component queues, or when you modify the configuration of a custom scheduler map that is already applied to packet forwarding component queues, packets already in the chassis queues might be dropped. The amount of packet loss is not deterministic and depends on the offered traffic load at the time you apply or modify the custom scheduler map.



## Examples: Scheduling Packet Forwarding Component Queues

### IN THIS SECTION

- [Example: Applying a Chassis Scheduler Map to a 2-Port IQ PIC | 994](#)
- [Example: Configuring Two T3 Interfaces on a Channelized DS3 IQ PIC | 996](#)
- [Example: Applying Normal Schedulers to Two T3 Interfaces | 998](#)
- [Example: Applying a Chassis Scheduler to Two T3 Interfaces | 1001](#)

### Example: Applying a Chassis Scheduler Map to a 2-Port IQ PIC

This example applies a chassis scheduler map to interfaces `so-0/1/0` and `so-0/1/1`.

According to customary wildcard rules, the `so-0/1/0` configuration overrides the `so-0/1/*` configuration, implying that the chassis scheduler map `MAP1` is not applied to `so-0/1/0`. However, the wildcard rule is not obeyed in this case; the chassis scheduler map applies to both interfaces `so-0/1/0` and `so-0/1/1`.

To configure the chassis queues to derive their scheduling from the associated logical interfaces:

1. Specify the logical interfaces from which to derive the scheduling configuration.

```
[edit class-of-service]
user@host# set interfaces so-0/1/0 unit 0 classifiers inet-precedence default
```

2. Using a wildcard rule, specify that the scheduler configuration is derived from the logical interfaces on `so-0/1*`.

```
[edit class-of-service]
user@host# set interfaces so-0/1/* scheduler-map-chassis derived
```

3. Review the configuration.

```
[edit]

user@host# show
class-of-service {
  interfaces {
    so-0/1/0 {
```



```

    unit 0 {
        classifiers {
            inet-precedence default;
        }
    }
}
so-0/1/* {
    scheduler-map-chassis derived;
}
}
}

```

4. Save the configuration.

```

[edit]
user@host# commit

```

### Not Recommended: Using a Wildcard for Gigabit Ethernet IQ Interfaces When Applying a Chassis Scheduler Map

On a Gigabit Ethernet IQ PIC, you can apply the chassis scheduler map at both the specific interface level and the wildcard level. We do not recommend this because the wildcard chassis scheduler map takes precedence, which might not be the desired effect. For example, if you want to apply the chassis scheduler map MAP1 to port 0 and MAP2 to port 1, we do not recommend the following:

```

[edit class-of-service]
user@host# set interfaces ge-0/1/0 scheduler-map-chassis MAP1
user@host# set interfaces ge-0/1/* scheduler-map-chassis MAP2

```

```

[edit class-of-service]

user@host# show
interfaces {
    ge-0/1/0 {
        scheduler-map-chassis MAP1;
    }
    ge-0/1/* {
        scheduler-map-chassis MAP2;
    }
}

```



## Recommended: Identifying Gigabit Ethernet IQ Interfaces Individually When Applying a Chassis Scheduler Map

Instead, we recommend this configuration:

```
[edit class-of-service]
user@host# set interfaces ge-0/1/0 scheduler-map-chassis MAP1
user@host# set interfaces ge-0/1/1 scheduler-map-chassis MAP2
```

```
[edit class-of-service]
user@host# show
interfaces {
  ge-0/1/0 {
    scheduler-map-chassis MAP1;
  }
  ge-0/1/1 {
    scheduler-map-chassis MAP2;
  }
}
```

## Example: Configuring Two T3 Interfaces on a Channelized DS3 IQ PIC

To configure two T3 interfaces on a channelized DS3 IQ PIC:

1. Configure the first channelized DS3 IQ interface.

- a. Specify the name of the interface.

```
[edit]
user@host# edit interfaces ct3-3/0/0
```

- b. Configure the interface as unpartitioned, clear channel.

```
[edit interfaces ct3-3/0/0]
user@host# set no-partition interface-type t3
```

2. Configure the second channelized DS3 IQ interface.



- a. Specify the name of the interface.

```
[edit]
user@host# edit interfaces ct3-3/0/1
```

- b. Configure the interface as unpartitioned, clear channel.

```
[edit interfaces ct3-3/0/1]
user@host# set no-partition interface-type t3
```

### 3. Configure the first T3 channel.

- a. Specify the name of the T3 interface on the DS3 IQ PIC.

```
[edit]
user@host# edit t3-3/0/0 unit 0
```

- b. Specify the protocol family and address of the interface.

```
[edit t3-3/0/0 unit 0]
user@host# set family inet address 10.0.100.1/30
```

### 4. Configure the second T3 channel.

- a. Specify the name of the T3 interface on the DS3 IQ PIC.

```
[edit]
user@host# edit t3-3/0/1 unit 0
```

- b. Specify the protocol family and address of the interface.

```
[edit t3-3/0/0 unit 0]
user@host# set family inet address 10.0.101.1/30
```

### 5. Review the configuration.

```
[edit interfaces]

user@host# show
```



```

ct3-3/0/0 {
    no-partition interface-type t3; # use entire port 0 as T3
}
ct3-3/0/1 {
    no-partition interface-type t3; # use entire port 1 as T3
}
t3-3/0/0 {
    unit 0 {
        family inet {
            address 10.0.100.1/30;
        }
    }
}
t3-3/0/1 {
    unit 0 {
        family inet {
            address 10.0.101.1/30;
        }
    }
}

```

6. Save the configuration.

```

[edit]
user@host# commit

```

### Example: Applying Normal Schedulers to Two T3 Interfaces

Configure a scheduler for the aggregated traffic transmitted into both T3 interfaces.

1. Specify the names of the scheduler maps for each interface.

```

[edit]
user@host# set class-of-service interfaces t3-3/0/0 scheduler-map sched-qct3-0
user@host# set class-of-service interfaces t3-3/0/1 scheduler-map sched-qct3-1

```

2. Specify the CoS parameters assigned to each forwarding class.

```

[edit]
user@host# set class-of-service scheduler-maps sched-qct3-0 forwarding-class best-effort
scheduler be-qct3-0

```



```

user@host# set class-of-service scheduler-maps sched-qct3-0 forwarding-class expedited-
forwarding scheduler ef-qct3-0
user@host# set class-of-service scheduler-maps sched-qct3-0 forwarding-class assured-
forwarding scheduler as-qct3-0
user@host# set class-of-service scheduler-maps sched-qct3-0 forwarding-class network-control
scheduler nc-qct3-0
user@host# set class-of-service scheduler-maps sched-qct3-1 forwarding-class best-effort
scheduler be-qct3-1
user@host# set class-of-service scheduler-maps sched-qct3-1 forwarding-class expedited-
forwarding scheduler ef-qct3-1
user@host# set class-of-service scheduler-maps sched-qct3-1 forwarding-class assured-
forwarding scheduler as-qct3-1
user@host# set class-of-service scheduler-maps sched-qct3-1 forwarding-class network-control
scheduler nc-qct3-1
user@host# set class-of-service scheduler-maps sched-chassis-to-q forwarding-class best-
effort scheduler be-chassis
user@host# set class-of-service scheduler-maps sched-chassis-to-q forwarding-class expedited-
forwarding scheduler ef-chassis
user@host# set class-of-service scheduler-maps sched-chassis-to-q forwarding-class assured-
forwarding scheduler as-chassis
user@host# set class-of-service scheduler-maps sched-chassis-to-q forwarding-class network-
control scheduler nc-chassis

```

3. Specify each scheduler name and the associated transmit rate..

```

user@host# set class-of-service schedulers be-qct3-0 transmit-rate percent 40
user@host# set class-of-service schedulers ef-qct3-0 transmit-rate percent 30
user@host# set class-of-service schedulers as-qct3-0 transmit-rate percent 20
user@host# set class-of-service schedulers nc-qct3-0 transmit-rate percent 10

```

4. Review the configuration.

```

[edit class-of-service]

user@host# show

interfaces {
    t3-3/0/0 {
        scheduler-map sched-qct3-0;
    }
    t3-3/0/1 {
        scheduler-map sched-qct3-1;
    }
}

```



```

    }
}
scheduler-maps {
    sched-qct3-0 {
        forwarding-class best-effort scheduler be-qct3-0;
        forwarding-class expedited-forwarding scheduler ef-qct3-0;
        forwarding-class assured-forwarding scheduler as-qct3-0;
        forwarding-class network-control scheduler nc-qct3-0;
    }
    sched-qct3-1 {
        forwarding-class best-effort scheduler be-qct3-1;
        forwarding-class expedited-forwarding scheduler ef-qct3-1;
        forwarding-class assured-forwarding scheduler as-qct3-1;
        forwarding-class network-control scheduler nc-qct3-1;
    }
    sched-chassis-to-q {
        forwarding-class best-effort scheduler be-chassis;
        forwarding-class expedited-forwarding scheduler ef-chassis;
        forwarding-class assured-forwarding scheduler as-chassis;
        forwarding-class network-control scheduler nc-chassis;
    }
}
schedulers {
    be-qct3-0 {
        transmit-rate percent 40;
    }
    ef-qct3-0 {
        transmit-rate percent 30;
    }
    as-qct3-0 {
        transmit-rate percent 20;
    }
    nc-qct3-0 {
        transmit-rate percent 10;
    }
}

```

##### 5. Save the configuration.

```

[edit]
user@host# commit

```



### Example: Applying a Chassis Scheduler to Two T3 Interfaces

Bind a scheduler to the aggregated traffic transmitted into the entire PIC. The chassis scheduler controls the traffic from the packet forwarding components feeding the interface t3-3/0/\*:

1. Using a wildcard rule, specify that the scheduler configuration is derived from the logical interfaces on t3-3/0/\*.

```
user@host# set class-of-service interfaces t3-3/0/* scheduler-map-chassis derived
```

2. Review the configuration.

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

interfaces {
  t3-3/0/* {
    scheduler-map-chassis derived;
  }
}
```

3. Save the configuration.

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

### Not Recommended: Using a Wildcard for Logical Interfaces When Applying a Scheduler

Do not apply a scheduler to a logical interface using a wildcard. For example, if you configure a logical interface (unit) with one parameter, and apply a scheduler map to the interface using a wildcard, the logical interface will not apply the scheduler. The following configuration will commit correctly but will not apply the scheduler map to interface so-3/0/0.0:

```
[edit]
user@host# set class-of-service interfaces so-3/0/* unit 0 scheduler-map MY_SCHED_MAP
user@host# set class-of-service interfaces so-3/0/0 unit 0 shaping-rate 100m
```

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



```

so-3/0/* {
  unit 0 {
    scheduler-map MY_SCHED_MAP;
  }
}
so-3/0/0 {
  unit 0 {
    shaping-rate 100m;
  }
}
}

```

### Recommended: Identifying Logical Interfaces Individually When Applying a Scheduler

Always apply the scheduler to a logical interface without the wildcard:

```

[edit]
user@host# set class-of-service interfaces so-3/0/0 unit 0 scheduler-map MY_SCHED_MAP
user@host# set class-of-service interfaces so-3/0/0 unit 0 shaping-rate 100m

```

```

[edit class of service]
user@host# show
interfaces {
  so-3/0/0 {
    unit 0 {
      scheduler-map MY_SCHED_MAP;
      shaping-rate 100m;
    }
  }
}

```



**NOTE:** This same wildcard behavior applies to classifiers and rewrites as well as schedulers.

## RELATED DOCUMENTATION

[Configuring Scheduler Maps | 338](#)

[Applying Scheduler Maps to Physical Interfaces | 340](#)







# Configuring Class of Service on Ethernet IQ2 and Enhanced IQ2 PICs

## IN THIS CHAPTER

- CoS on Enhanced IQ2 PICs Overview | 1005
- CoS Features and Limitations on IQ2 and IQ2E PICs (M Series and T Series) | 1006
- Differences Between Gigabit Ethernet IQ and Gigabit Ethernet IQ2 PICs | 1007
- Shaping Granularity Values for Enhanced Queuing Hardware | 1010
- Ethernet IQ2 PIC RTT Delay Buffer Values | 1013
- Configuring BA Classifiers for Bridged Ethernet | 1013
- Setting the Number of Egress Queues on IQ2 and Enhanced IQ2 PICs | 1016
- Configuring the Number of Schedulers per Port for Ethernet IQ2 PICs | 1017
- Applying Scheduler Maps to Chassis-Level Queues | 1019
- CoS for L2TP Tunnels on Ethernet Interface Overview | 1031
- Configuring CoS for L2TP Tunnels on Ethernet Interfaces | 1033
- Configuring LNS CoS for Link Redundancy | 1034
- Example: Configuring L2TP LNS CoS Support for Link Redundancy | 1035
- Configuring Shaping on 10-Gigabit Ethernet IQ2 PICs | 1040
- Configuring Per-Unit Scheduling for GRE Tunnels Using IQ2 and IQ2E PICs | 1042
- Understanding Burst Size Configuration on IQ2 and IQ2E Interfaces | 1044
- Configuring Burst Size for Shapers on IQ2 and IQ2E Interfaces | 1045
- Configuring a CIR and a PIR on Ethernet IQ2 Interfaces | 1046
- Example: Configuring Shared Resources on Ethernet IQ2 Interfaces | 1049
- Configuring and Applying IEEE 802.1ad Classifiers | 1053
- Configuring Rate Limits to Protect Lower Queues on IQ2 and Enhanced IQ2 PICs | 1055
- Simple Filters Overview | 1057
- Configuring a Simple Filter | 1058



## CoS on Enhanced IQ2 PICs Overview

Some PICs, such as the Gigabit Ethernet Intelligent Queuing 2 (IQ2) and Ethernet Enhanced IQ2 (IQ2E) PICs, have eight egress queues enabled by default on platforms that support eight queues.

The IQ2E PICs preserve all of the features of the IQ2 PICs, such as the default support for eight egress queues on platforms that support eight queues.

The IQ2E PICs add features such as the ability to perform hierarchical scheduling. You can mix IQ2 and IQ2E PICs on the same router.

The IQ2E PICs offer:

- Three levels of hierarchical CoS
- More granularity than a high priority queue
- 16,000 queues
- 2,000 schedulers with 8 queues
- 4,000 schedulers with 4 queues

The IQ2E PICs also offer automatic scheduler allocation across ports, so there is no need to reset the PIC when this changes. Random early detection (RED) keeps statistics on a per-drop-profile basis, improving the ability to perform network capacity planning.

When you include the `per-unit-scheduler` statement at the `[edit interfaces interface-name]` hierarchy level, each *logical interface* (unit) gets a dedicated scheduler (one scheduler is reserved for overflow). You can include the `per-session-scheduler` statement at the `[edit interfaces interface-name unit logical-unit-number]` hierarchy level to shape Layer 2 Tunneling Protocol (L2TP) sessions. The behavior of these two-port scheduler modes is the same as in IQ2 PICs. However, IQ2E PICs use hierarchical schedulers and not shared schedulers; IQ2E PICs do not support the `shared-scheduler` statement at the `[edit interfaces interface-name]` hierarchy level.

For more information about configuring hierarchical schedulers, including examples, see ["Configuring Hierarchical Schedulers for CoS" on page 448](#).

You can shape traffic at the physical interface (port), logical interface (unit), or interface set (set of units) levels. Shaping is not supported at the queue level. However, you can include the `transmit-rate` statement with the `rate-limit` option at the `[edit class-of-service schedulers scheduler-name]` hierarchy level to police the traffic passing through a queue (but only in the egress direction). See ["Configuring Rate Limits to Protect Lower Queues on IQ2 and Enhanced IQ2 PICs" on page 1055](#).

At the physical interface (port) level, you can configure only a shaping rate (PIR). At the logical interface (unit) and interface set levels, you can configure both a shaping rate and a guaranteed rate (CIR). Note that the guaranteed rates at any level must be consistent with the parent level's capacity. In other words,



the sum of the guaranteed rates on the logical interface (units) should be less than the guaranteed rate on the interface set, and the sum of the guaranteed rates on the logical interface (units) and interface sets should be less than the guaranteed rate on the physical interface (port).

You can control the rate of traffic that passes through the interface by configuring a policer overhead. When you configure a policer overhead, the configured policer overhead value is added to the length of the final Ethernet frame. This calculated length of the frame is used to determine the policer or the rate limit action. It does this because the policer overhead needs to be applied to policers just like shaping overhead is accounted for by shapers. The policer overhead is to be configured on the interface so that it is accounted for in the total packet length when policing traffic. See *Configuring a Policer Overhead*

The weighed RED (WRED) decision on the IQ2E PICs is done at the queue level. Once the accept or drop decision is made and the packet is queued, it is never dropped. Four drop profiles are associated with each queue: low, low-medium, medium-high, and high. WRED statistics are available for each loss priority (this feature is not supported on the IQ2 PICs). Also in contrast to the IQ2 PICs, the IQ2E PICs support WRED scaling profiles, allowing a single drop profile to be reused with a wide range of values. This practice increases the effective number of WRED drop profiles.

The IQ2E PICs provide four levels of strict priorities: strict-high, high, medium-high (medium-low) and low. In contrast to the IQ2 PICs, which support only one strict-high queue, the IQ2E PICs do not restrict the number of queues with a given priority. There is priority propagation among three levels: the logical interface, the logical interface set, and the physical port. These features are the same as those supported by Enhanced Queuing Dense Port Concentrators (DPCs) for Juniper Network MX Series 5G Universal Routing Platforms. For more information about configuring these features, see ["Enhanced Queuing DPC CoS Properties" on page 1071](#).

The IQ2E PIC's queues are serviced with modified deficit round-robin (MDRR), as with the Enhanced Queuing DPCs. Excess bandwidth (bandwidth available after all guaranteed rates have been satisfied) can be shared equally or in proportion to the guaranteed rates. For more information about excess bandwidth sharing, see ["Configuring Excess Bandwidth Sharing" on page 1081](#).

## RELATED DOCUMENTATION

*egress-policer-overhead*

*ingress-policer-overhead*

## CoS Features and Limitations on IQ2 and IQ2E PICs (M Series and T Series)

This topic describes CoS scaling and performance parameters that apply to IQ2 and IQ2E PICs on M series and T series routers.



### *Classification*

Behavior aggregate (BA) classification is done on the PIC. There are eight classifier tables of each type (ieee-802.1p, mpls-exp, inet-precedence, dscp, and dscp-ipv6) supported per PIC.

For each classifier type, one table is reserved for a default classifier. This table is used when no classifier is configured, or when the number of tables configured exceeds eight.

The following restrictions apply:

- You can only use BA classifiers for IPv4 DSCP bits for virtual private LAN service (VPLS).
- You cannot use BA classifiers for IPv4 DSCP bits for Layer 2 VPNs.
- You cannot use BA classifiers for IPv6 DSCP bits for VPLS.
- You cannot use BA classifiers for IPv6 DSCP bits for Layer 2 VPNs.

### *Rewrite Operations*

802.1p or 802.1ad rewrite operations are done on the PIC. A total of eight rewrite markers of each type are supported on the PIC. For other rewrite operations, the numbers are the same as for any other M series FPCs.

## Differences Between Gigabit Ethernet IQ and Gigabit Ethernet IQ2 PICs

Because Gigabit Ethernet IQ PICs and Gigabit Ethernet IQ2 PICs use different architectures, they differ in the following ways:

- Gigabit Ethernet IQ2 PICs support a transmission rate within a queue, but do not support an exact rate within a queue. You can apply a weight to a queue, but you cannot put an upper limit on the queue transmission rate that is less than the *logical interface* can support. Consequently, including the **exact** option with the **transmit-rate (rate | percent percent)** statement at the **[edit class-of-service schedulers scheduler-name]** hierarchy level is not supported for Gigabit Ethernet IQ2 interfaces.
- Gigabit Ethernet IQ2 PICs support only one queue in the scheduler map with **medium-high**, **high**, or **strict-high** priority. If more than one queue is configured with **medium-high**, **high**, or **strict-high** priority, the commit operation fails.
- To ensure that protocol control traffic (such as OSPF, BGP, and RIP) are not dropped at the oversubscribed ingress direction, the software puts control protocol packets into a separate control scheduler. There is one control scheduler per port. These control schedulers are implemented as strict-high priority, so they transmit traffic until they are empty.



- On Gigabit Ethernet IQ2 PICs, you can configure a single traffic-control profile to contain both a PIR (the **shaping-rate** statement) and a CIR (the **guaranteed-rate** statement). On Gigabit Ethernet IQ PICs, these statements are mutually exclusive.
- Gigabit Ethernet IQ2 PICs support only two fill levels in the RED drop profile. The recommended definition of the RED drop profile is as follows:

```
class-of-service {
  drop-profiles {
    drop-iq2-example1 {
      fill-level 20 drop-probability 0;
      fill-level 100 drop-probability 80;
    }
  }
}
```

This configuration defines a drop profile with a linear drop probability curve when the fill level is between 20 and 100 percent, and a maximum drop probability of 80 percent.

You can configure more than two fill levels in a drop profile, but the software only uses the points (**min\_fill\_level**, 0) and (**max\_fill\_level**, **max\_probability**) and ignores other fill levels. The drop probability at the minimum fill level is set to 0 percent even if you configure a non-zero drop probability value at the minimum fill level. The following example shows a sample configuration and the software implementation:

### Configuration

```
class-of-service {
  drop-profiles {
    drop-iq2-example2 {
      fill-level 30 drop-probability 10;
      fill-level 40 drop-probability 20;
      fill-level 100 drop-probability 80;
    }
  }
}
```

### Implementation

```
class-of-service {
  drop-profiles {
    drop-iq2-example2-implementation {
```



```

        fill-level 30 drop-probability 0;
        fill-level 100 drop-probability 80;
    }
}
}

```

If you configure more than two fill levels, a system log message warns you that the software supports only two fill levels and displays the drop profile that is implemented.

Though the **interpolate** statement is supported in the definition of a RED drop profile, we do not recommend using it. The following example shows a sample configuration and the software implementation:

### Configuration

```

class-of-service {
    drop-profiles {
        drop-iq2-example3 {
            interpolate {
                fill-level [ 30 50 80 ];
                drop-probability [ 10 20 40 ];
            }
        }
    }
}

```

When you use the **interpolate** statement and the maximum fill level is not 100 percent, the software adds the point (100, 100). Therefore, the drop-iq2-example3 drop profile is implemented as:

### Implementation

```

class-of-service {
    drop-profiles {
        drop-iq2-example3-implementation {
            fill-level 2 drop-probability 0;
            fill-level 100 drop-probability 100;
        }
    }
}

```

The implemented minimum fill level is not 30 percent as configured, but 2 percent because of the 64-point interpolation.



## Shaping Granularity Values for Enhanced Queuing Hardware

Due to the limits placed on shaping thresholds used in the hierarchy, there is a granularity associated with the Enhanced IQ2 (IQ2E) PIC and the Enhanced Queuing (EQ) DPC. For these hardware models, the shaper accuracies differ at various levels of the hierarchy, with shapers at the logical interface level (Level 3) being more accurate than shapers at the interface set level (Level 2) or the port level (Level 1). [Table 115 on page 1010](#) shows the accuracy of the logical interface shaper at various rates for Ethernet ports operating at 1 Gbps.

**Table 115: Shaper Accuracy of 1-Gbps Ethernet at the Logical Interface Level**

| Range of Logical Interface Shaper | Step Granularity |
|-----------------------------------|------------------|
| Up to 4.096 Mbps                  | 16 Kbps          |
| 4.096 to 8.192 Mbps               | 32 Kbps          |
| 8.192 to 16.384 Mbps              | 64 Kbps          |
| 16.384 to 32.768 Mbps             | 128 Kbps         |
| 32.768 to 65.535 Mbps             | 256 Kbps         |
| 65.535 to 131.072 Mbps            | 512 Kbps         |
| 131.072 to 262.144 Mbps           | 1024 Kbps        |
| 262.144 to 1 Gbps                 | 4096 Kbps        |

[Table 116 on page 1010](#) shows the accuracy of the logical interface shaper at various rates for Ethernet ports operating at 10 Gbps.

**Table 116: Shaper Accuracy of 10-Gbps Ethernet at the Logical Interface Level**

| Range of Logical Interface Shaper | Step Granularity |
|-----------------------------------|------------------|
| Up to 10.24 Mbps                  | 40 Kbps          |



**Table 116: Shaper Accuracy of 10-Gbps Ethernet at the Logical Interface Level (Continued)**

| Range of Logical Interface Shaper | Step Granularity |
|-----------------------------------|------------------|
| 10.24 to 20.48 Mbps               | 80 Kbps          |
| 10.48 to 40.96 Mbps               | 160 Kbps         |
| 40.96 to 81.92 Mbps               | 320 Kbps         |
| 81.92 to 163.84 Mbps              | 640 Kbps         |
| 163.84 to 327.68 Mbps             | 1280 Kbps        |
| 327.68 to 655.36 Mbps             | 2560 Kbps        |
| 655.36 to 2611.2 Mbps             | 10240 Kbps       |
| 2611.2 to 5222.4 Mbps             | 20480 Kbps       |
| 5222.4 to 10 Gbps                 | 40960 Kbps       |

[Table 117 on page 1011](#) shows the accuracy of the interface set shaper at various rates for Ethernet ports operating at 1 Gbps.

**Table 117: Shaper Accuracy of 1-Gbps Ethernet at the Interface Set Level**

| Range of Interface Set Shaper | Step Granularity |
|-------------------------------|------------------|
| Up to 20.48 Mbps              | 80 Kbps          |
| 20.48 Mbps to 81.92 Mbps      | 320 Kbps         |
| 81.92 Mbps to 327.68 Mbps     | 1.28 Mbps        |
| 327.68 Mbps to 1 Gbps         | 20.48 Mbps       |



[Table 118 on page 1012](#) shows the accuracy of the interface set shaper at various rates for Ethernet ports operating at 10 Gbps.

**Table 118: Shaper Accuracy of 10-Gbps Ethernet at the Interface Set Level**

| Range of Interface Set Shaper | Step Granularity |
|-------------------------------|------------------|
| Up to 128 Mbps                | 500 Kbps         |
| 128 Mbps to 512 Mbps          | 2 Mbps           |
| 512 Mbps to 2.048 Gbps        | 8 Mbps           |
| 2.048 Gbps to 10 Gbps         | 128 Mbps         |

[Table 119 on page 1012](#) shows the accuracy of the physical port shaper at various rates for Ethernet ports operating at 1 Gbps.

**Table 119: Shaper Accuracy of 1-Gbps Ethernet at the Physical Port Level**

| Range of Physical Port Shaper | Step Granularity |
|-------------------------------|------------------|
| Up to 64 Mbps                 | 250 Kbps         |
| 64 Mbps to 256 Mbps           | 1 Mbps           |
| 256 Mbps to 1 Gbps            | 4 Mbps           |

[Table 120 on page 1012](#) shows the accuracy of the physical port shaper at various rates for Ethernet ports operating at 10 Gbps.

**Table 120: Shaper Accuracy of 10-Gbps Ethernet at the Physical Port Level**

| Range of Physical Port Shaper | Step Granularity |
|-------------------------------|------------------|
| Up to 640 Mbps                | 2.5 Mbps         |



**Table 120: Shaper Accuracy of 10-Gbps Ethernet at the Physical Port Level (Continued)**

| Range of Physical Port Shaper | Step Granularity |
|-------------------------------|------------------|
| 640 Mbps to 2.56 Gbps         | 10 Mbps          |
| 2.56 Gbps to 10 Gbps          | 40 Mbps          |

## Ethernet IQ2 PIC RTT Delay Buffer Values

The following table shows the round-trip time (RTT) delay buffer values for IQ2 PICs, which are nonstandard and vary by PIC type and direction. The values are rounded up slightly to account for oversubscription.

**Table 121: RTT Delay Buffers for IQ2 PICs**

| IQ2 PIC Type                        | Ingress Buffer (ms) | Egress Buffer (ms) |
|-------------------------------------|---------------------|--------------------|
| 4-port Gigabit Ethernet (Type 1)    | 200                 | 300                |
| 8-port Gigabit Ethernet (Type 2)    | 175                 | 200                |
| 8-port Gigabit Ethernet (Type 3)    | 35                  | 225                |
| 1-port 10-Gigabit Ethernet (Type 3) | 25                  | 190                |

## Configuring BA Classifiers for Bridged Ethernet

On M120 and M320 routers equipped with IQ2 PICs, you can configure BA classification based on the IEEE 802.1 bits for bridged Ethernet over Asynchronous Transfer Mode (ATM), Point-to-Point Protocol (PPP), and frame relay for VPLS applications. The BA classification is applied to the first (outer) tag when tagged frames are received. Untagged frames are bypassed and a value of 000 for the classification IEEE 802.1p bits is assumed. There is no support for circuit cross-connect (CCC), and only port-mode



VPLS is supported (in port-mode VPLS, only VLANs on a single physical port are included in the VPLS instance). There is no support for multilink PPP bonding with VPLS. For bridging over frame relay, only frames that do not preserve the frame check sequence (FCS) field are supported. Frames that preserve the FCS field are silently discarded.

The bridging over PPP function is restricted:

- There is no support for “tinygram” compression and expansion.
- Frames received with preserved FCS bits are silently discarded.
- Bridge control frames are also classified based on header bit values.
- Both tagged and untagged frames are classified and forwarded. The peer must discard frame types that are not supported.

The following example applies an IEEE 802.1p classifier named `ppp-ether-vpls-classifier` to interface `(so-1/2/3)` with Ethernet VPLS over PPP encapsulation.



**NOTE:** The interface and CoS configuration must be consistent to support the feature. You must also configure the classifier and other CoS parameters such as forwarding classes.

1. Apply the CoS behavior aggregate classifier to a logical interface.

```
[edit]
user@host# set class-of-service interfaces so-1/2/3 unit 0 classifiers ieee-802.1 ppp-ether-
vpls-classifier
```

2. Configure the encapsulation and protocol family for the interface.

```
[edit]
user@host# set interfaces so-1/2/3 encapsulation ethernet-vpls-ppp unit 0 family vpls
```

3. Verify the configuration.

```
[edit class-of-service]
user@host# show
interfaces {
  so-1/2/3 {
    unit 0 {
      classifiers {
```



```

        ieee-802.1 ppp-ether-vpls-classifier;
    }
}
}
}

```

```

[edit interfaces]
user@host# show
s0-1/2/3 {
    encapsulation ether-vpls-over-ppp;
    unit 0 {
        family vpls;
    }
}

```

#### 4. Save the configuration.

```

[edit]
user@host# commit

```

On routers with IQ2 or IQ2E PICs, you can perform BA classification based on the value of the inner VLAN tag in an Ethernet frame.

1. To configure BA classification based on the inner VLAN tag value, specify the `inner` option at the `[edit class-of-service interfaces interface-name unit logical-unit-number classifiers ieee-802.1 classifier-name vlan-tag]` hierarchy level. You must also configure the inner VLAN tag for the logical interface with the `inner` option at the `[edit interfaces interface-name unit logical-interface-name vlan-tag]` hierarchy level.

```

[edit]
user@host# set class-of-service interfaces ge-2/2/2 unit 0 classifiers ieee-802.1 inner-vlan-
tag-ba-classifier
user@host# set class-of-service interfaces ge-2/2/2 unit 0 classifiers ieee-802.1 vlan-tag
inner

```

#### 2. Verify the configuration.

```

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

```



```

interfaces {
  ge-2/2/2 unit 0
    classifiers ieee-802.1 inner-vlan-tag-ba-classifier {
      vlan-tag inner;
    }
  }
}

```

### 3. Save the configuration.

```

[edit]
user@host# commit

```

## RELATED DOCUMENTATION

[CoS on Enhanced IQ2 PICs Overview | 1005](#)

[Configuring Behavior Aggregate Classifiers | 63](#)

[Default IEEE 802.1p Classifier | 53](#)

## Setting the Number of Egress Queues on IQ2 and Enhanced IQ2 PICs

Gigabit Ethernet IQ2 4-port and 8-port Type 2 PICs are oversubscribed, which means the amount of traffic coming to the PIC can be more than the maximum bandwidth from the PIC to the Flexible PIC Concentrator (FPC).

By default, PICs on M320, MX Series, and T Series routers support a maximum of four egress queues per interface. Some PICs, such as the IQ2 and IQ2E PICs, have eight egress queues enabled by default on platforms that support eight queues. You configure the number of egress queues as four or eight by including the `max-queues-per-interface` statement at the `[edit chassis fpc slot-number pic pic-slot-number]` hierarchy level:

```

[edit chassis fpc slot-number pic pic-slot-number]
max-queues-per-interface (4 | 8);

```

The numerical value can be 4 or 8.



For more information about configuring egress queues, see ["Configuring Up to 16 Custom Forwarding Classes" on page 283](#).

## Configuring the Number of Schedulers per Port for Ethernet IQ2 PICs

You can oversubscribe the Ethernet IQ2 family of PICs. Because of the bursty nature of Ethernet use, traffic received by the PIC can be several orders of magnitude greater than the maximum bandwidth leaving the PIC and entering the router. Several configuration statements apply only to Ethernet IQ2 PICs and allow the PIC to intelligently handle the oversubscribed traffic.



**NOTE:** The total of the input guaranteed rates for oversubscribed IQ2 PICs is limited to the FPC or PIC bandwidth.

By default, each Ethernet IQ2 PIC is allocated a fixed number of the 1024 available schedulers for each port during PIC initialization. For example, the 8-port Gigabit Ethernet IQ2 PIC is allocated 128 schedulers for each port. This number cannot be changed after the PIC is operational and can limit the utilization of shapers among the ports. Each of the 1024 schedulers is mapped at the logical interface (unit) level, and each scheduler map can support up to eight forwarding classes.

Schedulers are allocated in multiples of four. Three schedulers are reserved on each port. One is for control traffic, one is for port-level shaping, and the last is for unshaped logical interface traffic. These are allocated internally and automatically. The fourth scheduler is added when VLANs are configured.

When you configure schedulers for a port on an Ethernet IQ2 PIC:

- The three reserved schedulers are added to the configured value, which yields four schedulers per port.
- The configured value is adjusted upward to the nearest multiple of 4 (schedulers are allocated in multiples of 4).
- After all configured schedulers are allocated, any remaining unallocated schedulers are partitioned equally across the other ports.
- Any remaining schedulers that cannot be allocated meaningfully across the ports are allocated to the last port.

If the configured scheduler number is changed, the Ethernet IQ2 PIC is restarted when the configuration is committed.





**CAUTION:** If you deactivate and reactivate a port configured with a non-default number of schedulers, then the entire Ethernet IQ2 PIC restarts.

You can configure between 1 and 1024 schedulers on a port.

The following example allocates 100 schedulers to port 1 on an 8-port Gigabit Ethernet IQ2 PIC. The example shows the final scheduler allocation numbers for each port on the PIC. By default, each port would have been allocated  $1024 / 8 = 128$  schedulers. To configure the number of schedulers assigned to a port on an Ethernet IQ2 PIC:

- Specify the `schedulers` statement for the Ethernet IQ2 PIC interface at the `[edit interfaces ge-fpc/pic/port]` hierarchy level.

```
[edit interfaces ge-1/2/1]
user@host# set schedulers 100
```

This configuration results in the port and scheduler configuration shown in [Table 122 on page 1018](#).

**Table 122: Scheduler Allocation for an Ethernet IQ2 PIC**

| Ethernet IQ2 PIC Port | Number of Allocated Schedulers                                                           |
|-----------------------|------------------------------------------------------------------------------------------|
| 0                     | 128                                                                                      |
| 1                     | 104 (100 configured, plus 3 reserved, rounded up to multiple of 4: $100 + 3 + 1 = 104$ ) |
| 2                     | 128                                                                                      |
| 3                     | 128                                                                                      |
| 4                     | 128                                                                                      |
| 5                     | 128                                                                                      |
| 6                     | 128                                                                                      |



Table 122: Scheduler Allocation for an Ethernet IQ2 PIC (*Continued*)

| Ethernet IQ2 PIC Port | Number of Allocated Schedulers                                                       |
|-----------------------|--------------------------------------------------------------------------------------|
| 7                     | 152 (128 plus the 24 remaining that cannot be meaningfully allocated to other ports) |

## RELATED DOCUMENTATION

[How Schedulers Define Output Queue Properties](#) | 332

## Applying Scheduler Maps to Chassis-Level Queues

### IN THIS SECTION

- [Applying Custom Schedulers to Packet Forwarding Component Queues](#) | 1021
- [Examples: Scheduling Packet Forwarding Component Queues](#) | 1022

On Intelligent Queuing (IQ) and Intelligent Queuing 2 (IQ2) interfaces, as well as on the 10x10GE MIC with SFP+, the traffic that is fed from the packet forwarding components into the PIC uses low packet loss priority (PLP) by default and is distributed evenly across the four chassis queues (not PIC queues), regardless of the scheduling configuration for each logical interface. This default behavior can cause traffic congestion.

The default chassis scheduler allocates resources for queue 0 through queue 3, with 25 percent of the bandwidth allocated to each queue. When you configure the chassis to use more than four queues, you must configure and apply a custom chassis scheduler to override the default chassis scheduler.

To apply a custom chassis scheduler:

1. Specify the interface on which to apply the scheduler.

[edit]

```
user@host# edit class-of-service interfaces interface-name
```



For example:

```
[edit]
user@host# edit class-of-service interfaces so-0/1/*
```

2. Specify the name of the custom scheduler you want to apply to the interface.

```
[edit class-of-service interfaces so-0/1/0]
user@host# set scheduler-map-chassis map-name
```

To control the aggregated traffic transmitted from the chassis queues into the PIC, you can configure the chassis queues to derive their scheduling configuration from the associated logical interface's.

To configure the chassis queues to derive their scheduling from the associated logical interfaces:

1. Specify the logical interfaces from which to derive the scheduling configuration.

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

For example:

```
[edit]
user@host# edit class-of-service interfaces so-0/1/*
```

2. Specify that the scheduler configuration is derived from the specified logical interfaces.

```
[edit class-of-service interfaces so-0/1/0]
user@host# set scheduler-map-chassis derived
```



**CAUTION:** If you specify the scheduler-map-chassis derived statement in the configuration, packet loss might occur when you subsequently add or remove logical interfaces at the [edit interfaces *interface-name*] hierarchy level.

When fragmentation occurs on the egress interface, the first set of packet counters displayed in the output of the show interfaces queue command show the post-fragmentation values. The second set of packet counters (under the Packet Forwarding



Engine Chassis Queues field) show the pre-fragmentation values. For more information about the `show interfaces queue` command, see the [CLI Explorer](#).

You can specify both the `scheduler-map` and the `scheduler-map-chassis` derived statements in the same interface configuration. The `scheduler-map` statement controls the scheduler inside the PIC, while the `scheduler-map-chassis` derived statement controls the aggregated traffic transmitted into the entire PIC.



**NOTE:** For the Gigabit Ethernet IQ PIC, you must specify both the `scheduler-map` and the `scheduler-map-chassis` derived statements in the interface configuration.

Generally, when you specify the `scheduler-map-chassis` statement in the configuration, you must use an interface wildcard for the interface name, as in `type-fpc/pic/*`. The wildcard must use this format—for example, `so-1/2/*`, which means all interfaces on FPC slot 1, PIC slot 2. There is one exception—you can apply the chassis scheduler map to a specific interface on the Gigabit Ethernet IQ PIC only.

According to Junos OS wildcard rules, specific interface configurations override wildcard configurations. For chassis scheduler map configurations, this rule does not apply; instead, specific interface CoS configurations are added to the chassis scheduler map configuration. For more information about how wildcards work with chassis scheduler maps, see "[Examples: Scheduling Packet Forwarding Component Queues](#)" on page 994. For general information about wildcards, see the [Junos OS Administration Library for Routing Devices](#).



**NOTE:** The interface applies wildcard configuration only if you do not add any specific configuration. If you add the specific interface configuration, then the interface deletes the applied wildcard configuration and applies the specified configuration.

## Applying Custom Schedulers to Packet Forwarding Component Queues

Optionally, you can apply a custom scheduler to the chassis queues instead of configuring the chassis queues to automatically derive their scheduling configuration from the logical interfaces on the PIC.

To apply a custom chassis scheduler:

1. Specify the interface on which to apply the scheduler.

[edit]

```
user@host# edit class-of-service interfaces interface-name
```



For example:

```
[edit]
user@host# edit class-of-service interfaces so-0/1/*
```

2. Specify the name of the custom scheduler you want to apply to the interface.

```
[edit class-of-service interfaces so-0/1/0]
user@host# set scheduler-map-chassis map-name
```

When you apply a custom scheduler map to packet forwarding component queues, or when you modify the configuration of a custom scheduler map that is already applied to packet forwarding component queues, packets already in the chassis queues might be dropped. The amount of packet loss is not deterministic and depends on the offered traffic load at the time you apply or modify the custom scheduler map.

## Examples: Scheduling Packet Forwarding Component Queues

### IN THIS SECTION

- [Example: Applying a Chassis Scheduler Map to a 2-Port IQ PIC | 1022](#)
- [Example: Configuring Two T3 Interfaces on a Channelized DS3 IQ PIC | 1025](#)
- [Example: Applying Normal Schedulers to Two T3 Interfaces | 1027](#)
- [Example: Applying a Chassis Scheduler to Two T3 Interfaces | 1029](#)

### Example: Applying a Chassis Scheduler Map to a 2-Port IQ PIC

This example applies a chassis scheduler map to interfaces `so-0/1/0` and `so-0/1/1`.

According to customary wildcard rules, the `so-0/1/0` configuration overrides the `so-0/1/*` configuration, implying that the chassis scheduler map `MAP1` is not applied to `so-0/1/0`. However, the wildcard rule is not obeyed in this case; the chassis scheduler map applies to both interfaces `so-0/1/0` and `so-0/1/1`.

To configure the chassis queues to derive their scheduling from the associated logical interfaces:



1. Specify the logical interfaces from which to derive the scheduling configuration.

```
[edit class-of-service]
user@host# set interfaces so-0/1/0 unit 0 classifiers inet-precedence default
```

2. Using a wildcard rule, specify that the scheduler configuration is derived from the logical interfaces on so-0/1\*.

```
[edit class-of-service]
user@host# set interfaces so-0/1/* scheduler-map-chassis derived
```

3. Review the configuration.

```
[edit]

user@host# show
class-of-service {
  interfaces {
    so-0/1/0 {
      unit 0 {
        classifiers {
          inet-precedence default;
        }
      }
    }
    so-0/1/* {
      scheduler-map-chassis derived;
    }
  }
}
```

4. Save the configuration.

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

### Not Recommended: Using a Wildcard for Gigabit Ethernet IQ Interfaces When Applying a Chassis Scheduler Map

On a Gigabit Ethernet IQ PIC, you can apply the chassis scheduler map at both the specific interface level and the wildcard level. We do not recommend this because the wildcard chassis scheduler map



takes precedence, which might not be the desired effect. For example, if you want to apply the chassis scheduler map MAP1 to port 0 and MAP2 to port 1, we do not recommend the following:

```
[edit class-of-service]
user@host# set interfaces ge-0/1/0 scheduler-map-chassis MAP1
user@host# set interfaces ge-0/1/* scheduler-map-chassis MAP2
```

```
[edit class-of-service]

user@host# show
interfaces {
  ge-0/1/0 {
    scheduler-map-chassis MAP1;
  }
  ge-0/1/* {
    scheduler-map-chassis MAP2;
  }
}
```

### Recommended: Identifying Gigabit Ethernet IQ Interfaces Individually When Applying a Chassis Scheduler Map

Instead, we recommend this configuration:

```
[edit class-of-service]
user@host# set interfaces ge-0/1/0 scheduler-map-chassis MAP1
user@host# set interfaces ge-0/1/1 scheduler-map-chassis MAP2
```

```
[edit class-of-service]
user@host# show
interfaces {
  ge-0/1/0 {
    scheduler-map-chassis MAP1;
  }
  ge-0/1/1 {
    scheduler-map-chassis MAP2;
  }
}
```



### Example: Configuring Two T3 Interfaces on a Channelized DS3 IQ PIC

To configure two T3 interfaces on a channelized DS3 IQ PIC:

1. Configure the first channelized DS3 IQ interface.

- a. Specify the name of the interface.

```
[edit]  
user@host# edit interfaces ct3-3/0/0
```

- b. Configure the interface as unpartitioned, clear channel.

```
[edit interfaces ct3-3/0/0]  
user@host# set no-partition interface-type t3
```

2. Configure the second channelized DS3 IQ interface.

- a. Specify the name of the interface.

```
[edit]  
user@host# edit interfaces ct3-3/0/1
```

- b. Configure the interface as unpartitioned, clear channel.

```
[edit interfaces ct3-3/0/1]  
user@host# set no-partition interface-type t3
```

3. Configure the first T3 channel.

- a. Specify the name of the T3 interface on the DS3 IQ PIC.

```
[edit]  
user@host# edit t3-3/0/0 unit 0
```

- b. Specify the protocol family and address of the interface.

```
[edit t3-3/0/0 unit 0]  
user@host# set family inet address 10.0.100.1/30
```

4. Configure the second T3 channel.



- a. Specify the name of the T3 interface on the DS3 IQ PIC.

```
[edit]
user@host# edit t3-3/0/1 unit 0
```

- b. Specify the protocol family and address of the interface.

```
[edit t3-3/0/0 unit 0]
user@host# set family inet address 10.0.101.1/30
```

5. Review the configuration.

```
[edit interfaces]

user@host# show
ct3-3/0/0 {
    no-partition interface-type t3; # use entire port 0 as T3
}
ct3-3/0/1 {
    no-partition interface-type t3; # use entire port 1 as T3
}
t3-3/0/0 {
    unit 0 {
        family inet {
            address 10.0.100.1/30;
        }
    }
}
t3-3/0/1 {
    unit 0 {
        family inet {
            address 10.0.101.1/30;
        }
    }
}
```



6. Save the configuration.

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

### Example: Applying Normal Schedulers to Two T3 Interfaces

Configure a scheduler for the aggregated traffic transmitted into both T3 interfaces.

1. Specify the names of the scheduler maps for each interface.

```
[edit]
user@host# set class-of-service interfaces t3-3/0/0 scheduler-map sched-qct3-0
user@host# set class-of-service interfaces t3-3/0/1 scheduler-map sched-qct3-1
```

2. Specify the CoS parameters assigned to each forwarding class.

```
[edit]
user@host# set class-of-service scheduler-maps sched-qct3-0 forwarding-class best-effort
scheduler be-qct3-0
user@host# set class-of-service scheduler-maps sched-qct3-0 forwarding-class expedited-
forwarding scheduler ef-qct3-0
user@host# set class-of-service scheduler-maps sched-qct3-0 forwarding-class assured-
forwarding scheduler as-qct3-0
user@host# set class-of-service scheduler-maps sched-qct3-0 forwarding-class network-control
scheduler nc-qct3-0
user@host# set class-of-service scheduler-maps sched-qct3-1 forwarding-class best-effort
scheduler be-qct3-1
user@host# set class-of-service scheduler-maps sched-qct3-1 forwarding-class expedited-
forwarding scheduler ef-qct3-1
user@host# set class-of-service scheduler-maps sched-qct3-1 forwarding-class assured-
forwarding scheduler as-qct3-1
user@host# set class-of-service scheduler-maps sched-qct3-1 forwarding-class network-control
scheduler nc-qct3-1
user@host# set class-of-service scheduler-maps sched-chassis-to-q forwarding-class best-
effort scheduler be-chassis
user@host# set class-of-service scheduler-maps sched-chassis-to-q forwarding-class expedited-
forwarding scheduler ef-chassis
user@host# set class-of-service scheduler-maps sched-chassis-to-q forwarding-class assured-
forwarding scheduler as-chassis
```



```
user@host# set class-of-service scheduler-maps sched-chassis-to-q forwarding-class network-control scheduler nc-chassis
```

3. Specify each scheduler name and the associated transmit rate..

```
user@host# set class-of-service schedulers be-qct3-0 transmit-rate percent 40
user@host# set class-of-service schedulers ef-qct3-0 transmit-rate percent 30
user@host# set class-of-service schedulers as-qct3-0 transmit-rate percent 20
user@host# set class-of-service schedulers nc-qct3-0 transmit-rate percent 10
```

4. Review the configuration.

```
[edit class-of-service]

user@host# show

interfaces {
  t3-3/0/0 {
    scheduler-map sched-qct3-0;
  }
  t3-3/0/1 {
    scheduler-map sched-qct3-1;
  }
}
scheduler-maps {
  sched-qct3-0 {
    forwarding-class best-effort scheduler be-qct3-0;
    forwarding-class expedited-forwarding scheduler ef-qct3-0;
    forwarding-class assured-forwarding scheduler as-qct3-0;
    forwarding-class network-control scheduler nc-qct3-0;
  }
  sched-qct3-1 {
    forwarding-class best-effort scheduler be-qct3-1;
    forwarding-class expedited-forwarding scheduler ef-qct3-1;
    forwarding-class assured-forwarding scheduler as-qct3-1;
    forwarding-class network-control scheduler nc-qct3-1;
  }
}
sched-chassis-to-q {
  forwarding-class best-effort scheduler be-chassis;
  forwarding-class expedited-forwarding scheduler ef-chassis;
  forwarding-class assured-forwarding scheduler as-chassis;
  forwarding-class network-control scheduler nc-chassis;
```



```

    }
  }
  schedulers {
    be-qct3-0 {
      transmit-rate percent 40;
    }
    ef-qct3-0 {
      transmit-rate percent 30;
    }
    as-qct3-0 {
      transmit-rate percent 20;
    }
    nc-qct3-0 {
      transmit-rate percent 10;
    }
  }
}

```

5. Save the configuration.

```

[edit]
user@host# commit

```

### Example: Applying a Chassis Scheduler to Two T3 Interfaces

Bind a scheduler to the aggregated traffic transmitted into the entire PIC. The chassis scheduler controls the traffic from the packet forwarding components feeding the interface t3-3/0/\*:

1. Using a wildcard rule, specify that the scheduler configuration is derived from the logical interfaces on t3-3/0/\*.

```

user@host# set class-of-service interfaces t3-3/0/* scheduler-map-chassis derived

```

2. Review the configuration.

```

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

interfaces {
  t3-3/0/* {
    scheduler-map-chassis derived;
  }
}

```



```
}
}
```

### 3. Save the configuration.

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

### Not Recommended: Using a Wildcard for Logical Interfaces When Applying a Scheduler

Do not apply a scheduler to a logical interface using a wildcard. For example, if you configure a logical interface (unit) with one parameter, and apply a scheduler map to the interface using a wildcard, the logical interface will not apply the scheduler. The following configuration will commit correctly but will not apply the scheduler map to interface so-3/0/0.0:

```
[edit]
user@host# set class-of-service interfaces so-3/0/* unit 0 scheduler-map MY_SCHED_MAP
user@host# set class-of-service interfaces so-3/0/0 unit 0 shaping-rate 100m
```

```
[edit class of service]
user@host# show
interfaces {
  so-3/0/* {
    unit 0 {
      scheduler-map MY_SCHED_MAP;
    }
  }
  so-3/0/0 {
    unit 0 {
      shaping-rate 100m;
    }
  }
}
```

### Recommended: Identifying Logical Interfaces Individually When Applying a Scheduler



Always apply the scheduler to a logical interface without the wildcard:

```
[edit]
user@host# set class-of-service interfaces so-3/0/0 unit 0 scheduler-map MY_SCHED_MAP
user@host# set class-of-service interfaces so-3/0/0 unit 0 shaping-rate 100m
```

```
[edit class of service]
user@host# show
interfaces {
  so-3/0/0 {
    unit 0 {
      scheduler-map MY_SCHED_MAP;
      shaping-rate 100m;
    }
  }
}
```



**NOTE:** This same wildcard behavior applies to classifiers and rewrites as well as schedulers.

## RELATED DOCUMENTATION

[Configuring Scheduler Maps | 338](#)

[Applying Scheduler Maps to Physical Interfaces | 340](#)

[Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs | 391](#)

## CoS for L2TP Tunnels on Ethernet Interface Overview

For effective packet tunneling, CoS is implemented over L2TP tunnels. For Ethernet interfaces, CoS is supported for L2TP session traffic to a LAC on platforms configured as an LNS that include egress IQ2 or IQ2E PICs.

This feature is supported on the following platforms:

- EX Series switches
- M7i and M10i routers



- M120 routers

To enable session-aware CoS on an L2TP interface, include the `per-session-scheduler` statement at the `[edit interfaces unit logical-unit-number]` hierarchy level.

After CoS is configured on an L2TP tunnel, Junos OS dynamically creates a traffic shaper for the traffic-shaping-profile and the L2TP tunnel based on the tunnel identification number. This ensures that the packets are monitored at the LAC and classified to allow the traffic flow to be adjusted on congested networks.

This feature has the following limitations:

- Only 991 shapers are supported on each IQ2 or IQ2E PIC.
- For a 4-port IQ2E PIC, you can configure up to 1976 shapers for an 8-queue session and 3952 shapers for a 4-queue session.
- For an 8-port IQ2E PIC, you can configure up to 1912 shapers for an 8-queue session and up to 3824 shapers for a 4-queue session.
- Sessions in excess of the maximum supported values specified for the PICs cannot be shaped (but they can be policed).
- There is no support for PPP multilinks.
- The overall traffic rate cannot exceed the L2TP traffic rate, or else random drops result.
- There is no support for *logical interface* scheduling and shaping at the ingress because all schedulers are now reserved for L2TP.
- There is no support for physical interface rate shaping at the ingress.
- You cannot delete or deactivate the primary Ethernet interface on which the tunnel is established.

You can provide policing support for sessions with more than the maximum supported value on each IQ2 or IQ2E PIC. Each session can have four or eight different classes of traffic (queues). Each class needs its own policer; for example, one for voice and one for data traffic.

## RELATED DOCUMENTATION

[Configuring CoS for L2TP Tunnels on Ethernet Interfaces | 1033](#)

[Configuring LNS CoS for Link Redundancy | 1034](#)

[Example: Configuring L2TP LNS CoS Support for Link Redundancy | 1035](#)



## Configuring CoS for L2TP Tunnels on Ethernet Interfaces

The Layer 2 Tunneling Protocol (L2TP) is often used to carry traffic securely between an L2TP Network Server (LNS) to an L2TP Access Concentrator (LAC). CoS is supported for L2TP session traffic to a LAC on platforms configured as an LNS that include egress IQ2 and IQ2E Ethernet PICs.

This feature is supported on the following platforms:

- EX Series switches
- M7i and M10i routers
- M120 routers

To configure CoS for L2TP on Ethernet interfaces:

1. Configure L2TP services on the Ethernet interface.
2. On the Ethernet interface, enable session-aware CoS for L2TP sessions.

```
[[edit interfaces interface-name unit logical-unit-number]]
user@host# set per-session-scheduler
```

3. Configure the traffic manager in the IQ2 or IQ2E PIC to enable per-session CoS support.

```
[edit chassis fpc slot-number pic pic-number]
user@host# set traffic-manager mode-session-shaping
```

4. (Optional) To fine tune the system, you may also set the traffic-manager mode to session-shaping and configure the value of ingress-shaping-overhead parameter from 50 through 130 depending on your network requirement.

```
[edit chassis fpc slot-number pic pic-number]
user@host# set traffic-manager ingress-shaping-overhead value mode-session-shaping
```



**NOTE:** If you deactivate or delete the primary Ethernet interface on which the L2TP tunnel is configured, the tunnel with sessions having CoS is torn down.

After CoS is enabled for L2TP tunnels on Ethernet interface, you can run the **show class-of-service l2tp-session** command to verify the mapping of CoS with the configured L2TP session.



## RELATED DOCUMENTATION

*L2TP Minimum Configuration*

[CoS for L2TP Tunnels on Ethernet Interface Overview | 1031](#)

[Configuring LNS CoS for Link Redundancy | 1034](#)

[Example: Configuring L2TP LNS CoS Support for Link Redundancy | 1035](#)

*show class-of-service l2tp-session*

## Configuring LNS CoS for Link Redundancy

You can configure multiple ports on the same IQ2 and IQ2E PICs to support link redundancy for CoS on L2TP tunnels configured on an Ethernet interface. Link redundancy is useful when the active port is unavailable due to events such as:

- Disconnection of the cable
- Rebooting of the remote end system
- Traffic re-routing through a different port due to network conditions

When link redundancy is enabled in such scenarios, the L2TP tunnels and its session are maintained by switching traffic to another port configured on the same IQ2 or IQ2E PIC.

To configure multiple ports (IQ and IQ2PE PIC) on an Ethernet interface for redundancy with CoS, configure per-session-scheduler for all Ethernet ports:

```
user@host#edit interfaces ge-2/0/0 unit 0 per-session-scheduler
user@host#edit interfaces ge-2/0/1 unit 0 per-session-scheduler
```

You can similarly configure all the ports on the IQ2 or IQ2E PIC to support link redundancy for CoS on L2TP tunnels.



### NOTE:

- If one or more redundancy ports is removed from the configuration, the tunnels established through those redundancy ports also go down.



- You must configure per-session-scheduler for all the ports that are to be used for redundancy. If you do not do so, new tunnels or sessions with CoS do not get established.

## RELATED DOCUMENTATION

| [per-session-scheduler](#)

## Example: Configuring L2TP LNS CoS Support for Link Redundancy

### IN THIS SECTION

- [Requirements | 1035](#)
- [Overview | 1036](#)
- [Configuration | 1037](#)
- [Verification | 1038](#)

This example shows how link redundancy is supported when CoS for L2TP is configured on Ethernet interfaces.



**NOTE:** In this example, support for link redundancy is demonstrated by manually disabling the interface. However, link redundancy is also supported when the interface goes down due to events such as disconnection of the cable or rebooting of the remote end system.

## Requirements

Before you begin:

- Configure service and loopback interfaces.
- Configure CoS for L2TP.

This feature applies to M Series Multiservice Edge Router running Junos OS Release 12.1 or later and EX Series switches.



## Overview

### IN THIS SECTION

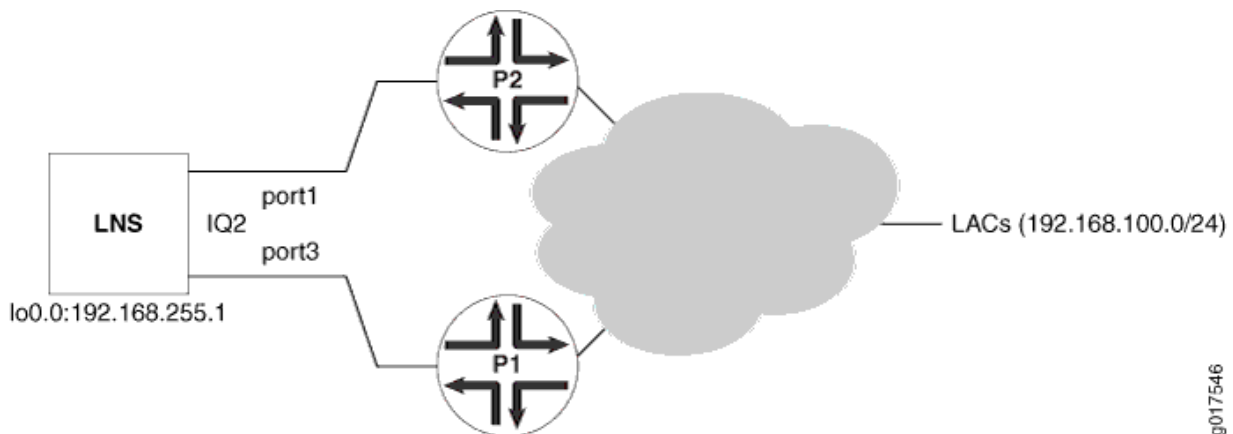
- [Topology | 1036](#)

Junos OS now supports link redundancy for CoS configured on an L2TP LNS. In this example, we verify that an L2TP tunnel does not go down when the Ethernet interface, through which the tunnels and its sessions with CoS are established, goes down.

### Topology

Figure 63 on page 1036 shows a sample scenario in which L2TP access concentrator (LAC) devices operate on one side of an L2TP tunnel. LAC devices are configured with the address range of 192.168.100.0 with a subnet mask of 24. The LAC devices are connected to two backbone routers, P1 and P2. These two routers, P1 and P2, are connected over two Gigabit Ethernet ports on a single Ethernet IQ2 PIC to an L2TP network server (LNS). The LNS device is a router running Junos OS that supports redundancy for terminating L2TP sessions configured with CoS parameters. The CoS settings are applied on the interfaces using a RADIUS server when the L2TP session is set up. One of the Gigabit Ethernet interfaces on the IQ2 PIC present on the LNS device, ge-0/3/1, is connected to P1, while the other interface, ge-0/3/3, is linked to P2. Such a method of connection enables the subscriber sessions that reach the LAC devices to be forwarded to one of the two ports of the IQ2 PIC on the LNS device.

**Figure 63: Topology to Verify Link Redundancy Support for L2TP LNS CoS**



g017546



## Configuration

### IN THIS SECTION

- Procedure | [1037](#)

## Procedure

### Step-by-Step Procedure

To configure Ethernet interfaces for redundancy:

1. Configure Gigabit Ethernet interfaces.

```
[edit interfaces]
user@host# set ge-0/3/1 unit 0 family inet address 192.168.1.1/30
user@host# set ge-0/3/3 unit 0 family inet address 192.168.1.5/30
user@host# set ge-0/3/1 unit 0 per-session-scheduler
user@host# set ge-0/3/3 unit 0 per-session-scheduler
```

2. Configure static routing options.

```
[edit routing-options]
user@host# set static route 192.168.100.0/24 next-hop [ 192.168.1.2 192.168.1.6 ]
```

### Step-by-Step Procedure

Verify that CoS is now implemented over L2TP on an Ethernet interface and the LAC is reachable.

1. Verify that LAC is reachable.

```
user@host> show route 192.168.100.1
inet.0: 14 destinations, 14 routes (14 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.100.0/24    *[Static/5] 1d 02:09:09
```



```

to 192.168.1.2 via ge-0/3/1.0
> to 192.168.1.6 via ge-0/3/3.0

```

2. Bring up an L2TP session and verify that L2TP sessions come up.

```

user@host> show services l2tp session
Interface: sp-1/3/0, Tunnel group: GEN-TUN-GRP-BIO, Tunnel local ID: 44806
Local Remote Interface State          Bundle Username
ID   ID   unit
12491 33795      1 Established          - test1

```

3. Send a traffic stream towards the subscriber.
4. Verify that the shaping at the subscriber end is as per the shaping rate configured.

```

user@host# show class-of-service l2tp-session
L2TP Session Username: test1, Index: 12491
Physical interface: ge-0/3/3, Index: 131
Queues supported: 4, Queues in use: 4
Scheduler map: GEN-SCHED-MAP-EF-65%, Index: 5212
Shaping rate: 2162200 bps
Encapsulation Overhead: 6, Cell Overhead: Enabled

```

In the output of the `show class-of-service l2tp-session` command, `ge-0/3/3`, index 131 represents the port used to establish the L2TP tunnel to which the current L2TP session belongs. It does not represent the port that was active when the L2TP session came up.

## Verification

### IN THIS SECTION

- [Bring Down ge-0/3/3 Interface Through Which the L2TP Tunnel Is Established | 1039](#)
- [Verify LAC Reachability and the Status of L2TP Sessions | 1039](#)

Verify that, when CoS is configured on an L2TP tunnel, link redundancy works if one of the ports on which the L2TP tunnel is established goes down.



## Bring Down ge-0/3/3 Interface Through Which the L2TP Tunnel Is Established

### Purpose

Bring down the interface through which the L2TP session and its tunnels are established.

### Action

```
[edit interfaces]
user@host# set ge-0/3/3 disable
user@host# commit
```

## Verify LAC Reachability and the Status of L2TP Sessions

### Purpose

Verify that link redundancy works and the L2TP session does not go down when the active port on the IQ2 PIC is down. Verify that the traffic flow is unaffected after it is switched to another port configured on the same IQ2 or IQ2E PIC.

### Action

```
user@host> show route 192.168.100.1
inet.0: 14 destinations, 14 routes (14 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

192.168.100.0/24    *[Static/5] 1d 02:35:09
                  to 192.168.1.2 via ge-0/3/1.0

user@host> show services l2tp session
Interface: sp-1/3/0, Tunnel group: GEN-TUN-GRP-BIO, Tunnel local ID: 44806
  Local Remote Interface State          Bundle Username
  ID    ID      unit
  12491 33795      1 Established          - test1
```

## RELATED DOCUMENTATION

| [Configuring LNS CoS for Link Redundancy](#) | 1034



## Configuring Shaping on 10-Gigabit Ethernet IQ2 PICs

The 10-Gigabit Ethernet IQ2 PIC (which has xe- interfaces) is unlike other Gigabit Ethernet IQ2 PICs in that it does not have oversubscription. The bandwidth from the PIC to the FPC is sufficient to transmit the full line rate. However, the 10-Gigabit Ethernet IQ2 PIC has the same hardware architecture as other Gigabit Ethernet IQ2 PICs and supports all the same class-of-service (CoS) features. For more information, see the [Ethernet Interfaces User Guide for Routing Devices](#).

To handle oversubscribed traffic on 10-Gigabit Ethernet IQ2 PICs, you can configure input shaping and scheduling based on Layer 2, MPLS, and Layer 3 packet fields. Gigabit Ethernet IQ2 PICs also support simple filters, accounting, and policing. This topic discusses input and output shaping and scheduling. For information about accounting and policing, see the [Junos OS Network Interfaces Library for Routing Devices](#).



**NOTE:** The CoS functionality supported on Gigabit Ethernet IQ2 PICs is not available across aggregated Ethernet links. However, if you configure a CoS scheduler map on the link bundle, the configuration is honored by the individual links within that bundle. Therefore, CoS behaves as configured on a per-link level, but *not* across the aggregated links.

If you configure a shaping transmit rate of 100 Mbps on an aggregated Ethernet bundle with three ports (by applying a scheduler for which the configuration includes the `transmit-rate` statement with the `exact` option at the `[edit class-of-service schedulers scheduler-name]` hierarchy level), each port is provisioned with a 33.33 Mbps shaping transmit rate.

You can configure shaping for aggregated Ethernet interfaces that use interfaces originating from Gigabit Ethernet IQ2 PICs. However, you cannot enable shaping on aggregated Ethernet interfaces when the aggregate bundle combines ports from IQ and IQ2 PICs.

By default, transmission 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. The default operation can be changed by configuring the software.

To configure input and output scheduling and shaping on 10-Gigabit Ethernet IQ2 PICs:

- 1.



Create the traffic-control profile, including the required scheduler map and shaping rate.

```
[edit class-of-service]
user@host# edit traffic-control-profiles tcp1
  user@host# set scheduler-map map1
  user@host# set shaping-rate 100
```

2. Apply the traffic-control profile to logical interface for either input scheduling and shaping and/or output scheduling and shaping:

To apply the traffic-control profile to a logical interface for input scheduling and shaping.

```
[edit class-of-service]
user@host# edit interfaces xe-1/2/1 unit 0
  user@host# set input-traffic-control-profile tcp1 shared-instance map1
```

To apply the traffic-control profile to a logical interface for output scheduling and shaping.

```
user@host# set output-traffic-control-profile tcp1 shared-instance map1
```

To apply the traffic-control profile to a logical interface for both input and output scheduling and shaping.

```
[edit class-of-service]
user@host# edit interfaces xe-1/2/1 unit 0
  user@host# set input-traffic-control-profile tcp1 shared-instance map1
  user@host# set output-traffic-control-profile tcp1 shared-instance map1
```

3. Enable per-unit scheduling (per-unit-scheduler statement) to enable the association of scheduler maps with logical interfaces.

```
[edit interfaces xe-1/2/1]
user@host# set per-unit-scheduler
```



**NOTE:** The scheduler-map and shaping-rate statements can be specified at the [edit class-of-service interfaces *interface-name* unit *logical-unit-number*] hierarchy level. However, we do not recommend this configuration. Include the output-traffic-control-profile or input-traffic-control-profile statement instead.





**NOTE:** For Gigabit Ethernet IQ2 interfaces, the logical interface egress statistics displayed in the `show interfaces` command output might not accurately reflect the traffic on the wire when output shaping is applied. Traffic management output shaping might drop packets after they are tallied by the `Output bytes` and `Output packets` logical interface counters. However, correct values display for both of these `Transit` statistics when per-unit scheduling is enabled for the Gigabit Ethernet IQ2 physical interface, or when a single logical interface is actively using a shared scheduler.

## RELATED DOCUMENTATION

*Simple Filter Overview*

*How Simple Filters Evaluate Packets*

*Guidelines for Configuring Simple Filters*

*Guidelines for Applying Simple Filters*

## Configuring Per-Unit Scheduling for GRE Tunnels Using IQ2 and IQ2E PICs

This topic describes how to configure per unit scheduling for Generic Routing Encapsulation (GRE) tunnels running on Intelligent Queuing 2 (IQ2) PICs and Intelligent Queuing 2 Enhanced (IQ2E) PICs in M7i, M10i, M120, M320, T Series and TX Matrix routers.

M7i, M10i, M120, M320, T Series and TX Matrix routers with Intelligent Queuing 2 (IQ2) PICs and Intelligent Queuing 2 Enhanced (IQ2E) PICs support per unit scheduling for Generic Routing Encapsulation (GRE) tunnels, adding all the functionality of tunnel PICs to GRE tunnels. The class of service (CoS) for the GRE tunnel traffic is applied as the traffic is looped through the IQ2 and IQ2E PIC.

Shaping is performed on full packets that pass through the GRE tunnel.

IQ2 and IQ2E PICs support all interfaces that are supported on tunnel PICs, as follows:

- `gr-fpc/pic/port`
- `vt-fpc/pic/port`
- `lt-fpc/pic/port`
- `ip-fpc/pic/port`



- `pe-fpc/pic/port`
- `pd-fpc/pic/port`
- `mt-fpc/pic/port`

The port variable is always zero.

The IQ2 and IQ2E PICs tunnel functionality is the same as that of regular tunnel PICs.

You can specify that IQ2 and IQ2E PICs work exclusively in tunnel mode or as a regular PIC. When IQ2 and IQ2E PICs work exclusively as a tunnel PIC, they support the same number of tunnel logical interfaces as regular tunnel PICs; for example each PIC can support 4,000 gr- logical interfaces. The default setting uses IQ2 and IQ2E PICs as a regular PIC.

IQ2E PIC schedulers can be dynamically allocated across ports.



**NOTE:** When IQ2 and IQ2E PICs work exclusively in tunnel mode only traffic-control-profile on gr- logical interfaces are supported. Class of Service (CoS) on gr- logical interfaces is not supported.

Also, a scheduler is allocated for a gr- logical interface only when there is a traffic-control profile configured for it.

The gr- logical interfaces without an explicit CoS configuration are not assigned a dedicated scheduler. These use a reserved scheduler meant for all unshaped tunnel traffic; that is, all traffic on gr- logical interfaces that do not have CoS configured and all traffic from other types of tunnels.

On gr- interfaces, you can configure an output traffic control profile on the logical interface:

### 1. Configuring the IQ2 and IQ2E PIC to work exclusively in tunnel mode

For example:

[edit]

```
user@host# set chassis fpc 1 pic 1 tunnel-services tunnel-only
```



**NOTE:** The PIC will be automatically bounced when the tunnel services configuration is changed.

The chassis traffic-manager mode must have the ingress traffic manager enabled in order for the tunnel-services to work correctly.

### 2. Create the traffic control profile and specify the shaping rate.



For example:

```
[edit]
user@host# set class-of-service traffic-control-profiles tcp shaping-rate 1000
user@host# set class-of-service interfaces gr-1/1/1 unit 0 output-traffic-control-profile tcp
```

3. Apply the profile to the logical interface as an output traffic control profile.

For example:

```
[edit]
user@host# set class-of-service interfaces gr-1/1/1 unit 0 output-traffic-control-profile tcp
```

4. To verify the configuration and view statistics:

- You can use the `show interfaces queue gr-fpc/pic/port` command to display statistics for the specified tunnel.
- To view the configuration and statistics for GRE tunnel logical interfaces, use the `show interfaces queue gr-` command.

## RELATED DOCUMENTATION

*tunnel-services (Chassis)*

[Configuring CoS for GRE and IP-IP Tunnels | 878](#)

[CoS on Enhanced IQ2 PICs Overview | 1005](#)

## Understanding Burst Size Configuration on IQ2 and IQ2E Interfaces

You can explicitly configure the burst size for shapers in a traffic control profile for IQ2 and IQ2E interfaces. This feature is supported on M7i, M10i, M40e, M120, M320 routers and all T Series routers.

The shaping burst size determines the maximum number of bytes that can be sent through a shaper during a burst. The guaranteed burst size determines when the scheduler moves from green to yellow.

The burst size limits the number of credits that can be accumulated for scheduling. Configuring a burst size is only useful in the case when traffic is sent after a long lull period so that credits can be accumulated until the burst size limit is reached. When traffic is continuous, credits are not accumulated, and the burst size limit is not reached.



If no burst size value is specified when the shaping rate or guaranteed rate is configured, then a default burst size (expressed as a time value) is applied. The default shaping burst size is 10 ms of the shaping rate (that is,  $10 \times \text{shaping rate} / 1000$  bytes). The minimum value is 2048 bytes to accommodate the minimum of 1 MTU.

The burst size value is adjusted and rounded off to meet the restrictions enforced by the hardware. Thus, the actual burst size in the hardware might vary slightly from the configured value.

To enable this feature, include the burst-size statement at the following hierarchy levels:

```
[edit class-of-service traffic-control-profiles shaping-rate]
[edit class-of-service traffic-control-profiles guaranteed-rate]
```



**NOTE:** The guaranteed-rate burst size value cannot be greater than the shaping-rate burst size.

## RELATED DOCUMENTATION

[Configuring Burst Size for Shapers on IQ2 and IQ2E Interfaces | 1045](#)

*guaranteed-rate*

*shaping-rate*

## Configuring Burst Size for Shapers on IQ2 and IQ2E Interfaces

This topic shows how to set the burst-size while configuring the shaping-rate and guaranteed-rate under the `[edit class-of-service traffic-control-profiles profile-name]` hierarchy level.

In following configuration for `tcp1`, the shaping-rate burst size is set to 5 KB, and the guaranteed-rate burst size is set to 3 KB under the traffic-control-profiles statement. To apply this configuration to a logical interface (ifl), the traffic-control-profile is attached to the ifl.

```
class-of-service {
  traffic-control-profiles {
    tcp1 {
      shaping-rate 100m burst-size 5k;
      guaranteed-rate 50m burst-size 3k;
    }
  }
}
```



```

    tcp2 {
        shaping-rate 100m burst-size 5k;
    }
}

interfaces {
    interface-set ifset1 {
        output-traffic-control-profile tcp1;
    }
    ge-1/2/1 {
        unit 0 {
            output-traffic-control-profile tcp1;
        }
    }
    ge-1/2/2 {
        output-traffic-control-profile tcp2;
    }
}
}

```

## RELATED DOCUMENTATION

[Understanding Burst Size Configuration on IQ2 and IQ2E Interfaces | 1044](#)

*guaranteed-rate*

*shaping-rate*

## Configuring a CIR and a PIR on Ethernet IQ2 Interfaces

On Gigabit Ethernet IQ2 interfaces, you can configure a CIR (guaranteed rate) and a PIR (shaping rate) on a single logical interface. The configured rates are gathered into a traffic control profile. If you configure a traffic control profile with a CIR (guaranteed rate) only, the PIR (shaping rate) is set to the physical interface (port) rate.

In the following example, logical unit 0 has a CIR equal to 30 Mbps and a PIR equal to 200 Mbps. Logical unit 1 has a PIR equal to 300 Mbps. Logical unit 2 has a CIR equal to 100 Mbps and a PIR that is unspecified. For logical unit 2, the software causes the PIR to be 1 Gbps (equal to the physical interface rate) because the PIR must be equal to or greater than the CIR.



*Excess bandwidth* is the leftover bandwidth on the port after meeting all the guaranteed rate requirements of the logical interfaces. For each port, excess bandwidth is shared as follows:

- Proportional to the guaranteed rate—This method is used if you configure one or more logical interfaces on a port to have a guaranteed rate.
- Proportional to the shaping rate—This method is used if you configure none of the logical interfaces on a port to have a guaranteed rate.

In this example, bandwidth is shared proportionally to the guaranteed rate because at least one logical interface has a guaranteed rate.

1. Create and configure the traffic control profile.

```
[edit]
user@host# set class-of-service traffic-control-profiles profile1 scheduler-map sched-map
user@host# set class-of-service traffic-control-profiles profile1 shaping-rate 200m
user@host# set class-of-service traffic-control-profiles profile1 guaranteed-rate 30m
user@host# set class-of-service traffic-control-profiles profile1 delay-buffer-rate 150m
user@host# set class-of-service traffic-control-profiles profile2 scheduler-map sched-map
user@host# set class-of-service traffic-control-profiles profile2 shaping-rate 300m
user@host# set class-of-service traffic-control-profiles profile2 delay-buffer-rate 500k
user@host# set class-of-service traffic-control-profiles profile3 scheduler-map sched-map
user@host# set class-of-service traffic-control-profiles profile3 guaranteed-rate 100m
```

2. Apply the traffic control profiles to the interfaces

```
[edit]
user@host# set class-of-service interfaces ge-3/0/0 unit 0 output-traffic-control-profile
profile1
user@host# set class-of-service interfaces ge-3/0/0 unit 1 output-traffic-control-profile
profile2
user@host# set class-of-service interfaces ge-3/0/0 unit 2 output-traffic-control-profile
profile3
```

3. View the configuration.

```
[edit]

user@host# show class-of-service traffic-control-profiles

profile1 {
```



```

    scheduler-map sched-map;
    shaping-rate 200m;
    guaranteed-rate 30m;
    delay-buffer-rate 150m;
}
profile2 {
    scheduler-map sched-map;
    shaping-rate 300m;
    delay-buffer-rate 500k;
}
profile3 {
    scheduler-map sched-map;
    guaranteed-rate 100m;
}

```

[edit]

```

user@host# show class-of-service interfaces
ge-3/0/0 {
    unit 0 {
        output-traffic-control-profile profile1;
    }
    unit 1 {
        output-traffic-control-profile profile2;
    }
    unit 2 {
        output-traffic-control-profile profile3;
    }
}

```

## RELATED DOCUMENTATION

[CoS on Enhanced IQ2 PICs Overview | 1005](#)

[Configuring Traffic Control Profiles for Shared Scheduling and Shaping | 340](#)



## Example: Configuring Shared Resources on Ethernet IQ2 Interfaces

For input traffic on physical interface ge-1/2/3, logical interface units 1, 2, and 3 are sharing one set of scheduler-shaper resources, defined by traffic-control profile s1. Logical interface units 4, 5, and 6 are sharing another set of scheduler-shaper resources, defined by traffic-control profile s1.

For output traffic on physical interface ge-1/2/3, logical interface units 1, 2, and 3 are sharing one set of scheduler-shaper resources, defined by traffic-control profile s2. Logical interface units 4, 5, and 6 are sharing another set scheduler-shaper resources, defined by traffic-control profile s2.

For each physical interface, the shared-instance statement creates one set of resources to be shared among units 1, 2, and 3 and another set of resources to be shared among units 4, 5, and 6. Input and output shaping rates are configured at the physical interface level, which demonstrates the hierarchical shaping capability of the Gigabit Ethernet IQ2 PIC.

```
[edit]
class-of-service {
  traffic-control-profiles {
    s1 {
      scheduler-map map1;
      shaping-rate 100k;
    }
    s2 {
      scheduler-map map1;
      shaping-rate 200k;
    }
  }
  forwarding-classes { # Map one forwarding class to one queue.
    queue 0 fc-be;
    queue 1 fc-be1;
    queue 2 fc-ef;
    queue 3 fc-ef1;
    queue 4 fc-af11;
    queue 5 fc-af12;
    queue 6 fc-nc1;
    queue 7 fc-nc2;
  }
  classifiers { # Map 802.1p bits to forwarding-class and loss-priority.
    ieee-802.1 ieee-8021p-table {
      forwarding-class fc-nc2 {
        loss-priority low code-points [111];
      }
    }
  }
}
```



```

forwarding-class fc-nc1 {
    loss-priority low code-points [110];
}
forwarding-class fc-af12 {
    loss-priority low code-points [101];
}
forwarding-class fc-af11 {
    loss-priority low code-points [100];
}
forwarding-class fc-ef1 {
    loss-priority low code-points [011];
}
forwarding-class fc-ef {
    loss-priority low code-points [010];
}
forwarding-class fc-be1 {
    loss-priority low code-points [001];
}
forwarding-class fc-be {
    loss-priority low code-points [000];
}
}
}
interfaces {
    ge-1/2/3 {
        input-shaping-rate 500m;
        shaping-rate 500m; # Output shaping rate
        unit 0 { # Apply behavior aggregate classifier to an interface.
            classifiers {
                ieee-802.1 ieee-8021p-table;
            }
        }
        unit 1 {
            input-traffic-control-profile s1 shared-instance 1;
            output-traffic-control-profile s2 shared-instance 1;
        }
        unit 2 {
            input-traffic-control-profile s1 shared-instance 1;
            output-traffic-control-profile s2 shared-instance 1;
        }
        unit 3 {
            input-traffic-control-profile s1 shared-instance 1;
            output-traffic-control-profile s2 shared-instance 1;
        }
    }
}

```



```

    }
    unit 4 {
        input-traffic-control-profile s1 shared-instance 2;
        output-traffic-control-profile s2 shared-instance 2;
    }
    unit 5 {
        input-traffic-control-profile s1 shared-instance 2;
        output-traffic-control-profile s2 shared-instance 2;
    }
    unit 6 {
        input-traffic-control-profile s1 shared-instance 2;
        output-traffic-control-profile s2 shared-instance 2;
    }
}
}
}
}

```

### Configuring a Simple Filter

Configure a simple filter that overrides the classification derived from the lookup of the Layer 2 fields.

```

firewall {
    family inet {
        simple-filter sf-1 {
            term 1 {
                source-address 172.16.0.0/24;
                destination-address 172.16.20.0/24;
                source-port 1024-9071;
            }
            then { # Action with term-1
                forwarding-class fc-be1;
                loss-priority high;
            }
            term 2 {
                source-address 172.16.10.0/24;
                destination-address 172.16.30.0/24;
            }
            then { # Action with term-2
                forwarding-class fc-ef1;
                loss-priority low;
            }
        }
    }
}

```



```

interfaces { # Apply the simple filter.
ge-1/2/3 {
    unit 0 {
        family inet {
            simple-filter {
                input sf-1;
            }
        }
    }
}

class-of-service {
    scheduler-maps { # Configure a custom scheduler map.
    map1 {
        forwarding-class fc-be scheduler sch-Q0;
        forwarding-class fc-be1 scheduler sch-Q1;
        forwarding-class fc-ef scheduler sch-Q2;
        forwarding-class fc-ef1 scheduler sch-Q3;
        forwarding-class fc-af11 scheduler sch-Q4;
        forwarding-class fc-af12 scheduler sch-Q5;
        forwarding-class fc-nc1 scheduler sch-Q6;
        forwarding-class fc-nc2 scheduler sch-Q7;
    }
}

schedulers { # Define schedulers.
sch-Q0 {
    transmit-rate percent 25;
    buffer-size percent 25;
    priority low;
    drop-profile-map loss-priority any protocol any drop-profile drop-default;
}

sch-Q1 {
    transmit-rate percent 5;
    buffer-size temporal 2000;
    priority high;
    drop-profile-map loss-priority any protocol any drop-profile drop-ef;
}

sch-Q2 {
    transmit-rate percent 35;
    buffer-size percent 35;
    priority low;
    drop-profile-map loss-priority any protocol any drop-profile drop-default;
}
}

```



```

sch-Q3 {
    transmit-rate percent 5;
    buffer-size percent 5;
    drop-profile-map loss-priority any protocol any drop-profile drop-default;
}
sch-Q4 {
    transmit-rate percent 5;
    priority high;
    drop-profile-map loss-priority any protocol any drop-profile drop-ef;
}
sch-Q5 {
    transmit-rate percent 10;
    priority high;
    drop-profile-map loss-priority any protocol any drop-profile drop-ef;
}
sch-Q6 {
    transmit-rate remainder;
    priority low;
    drop-profile-map loss-priority any protocol any drop-profile drop-default;
}
sch-Q7 {
    transmit-rate percent 5;
    priority high;
    drop-profile-map loss-priority any protocol any drop-profile drop-default;
}

```

## RELATED DOCUMENTATION

[Configuring Traffic Control Profiles for Shared Scheduling and Shaping](#) | 340

## Configuring and Applying IEEE 802.1ad Classifiers

If you apply an IEEE 802.1 classifier to a logical interface, this classifier takes precedence and is not compatible with any other classifier type. For Juniper Networks MX Series 5G Universal Routing Platform interfaces or IQ2 PICs with IEEE 802.1ad frame formats or EX Series switches, you can set the forwarding class and loss priority for traffic on the basis of the three IEEE 802.1p bits (three bits in either the inner virtual LAN (VLAN) tag or the outer VLAN tag) and the drop eligible indicator (DEI) bit. You can apply the default map or customize one or more of the default values.



You then apply the classifier to the interface on which you configure IEEE 802.1ad frame formats.

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

**1. Define the custom IEEE 802.1ad map:**

- a. Create the classifier by specifying a name for it and defining it as an IEEE-802.1ad (DEI) classifier.

```
[edit]
user@host# edit class-of-service classifiers ieee-802.1ad dot1p_dei_class
```

- b. Assign the forwarding class and loss priority to the code-point alias.

```
[edit class-of-service classifiers ieee-802.1ad dot1p_dei_class]
user@host# set forwarding-class best-effort loss-priority low code-points [0000 1101]
```

**2. Apply the classifier to the logical interface:**

- a. Specify the interface to which you want to apply the classifier.

```
[edit]
user@host# edit class-of-service interfaces ge-2/0/0 unit 0
```

- b. Specify the name of the classifier you want to apply to the interface.

```
[edit class-of-service interfaces ge-2/0/0 unit 0]
user@host# set classifiers ieee-802.1ad dot1p_dei_class
```

**3. Verify the custom IEEE 802.1ad map configuration:**

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

```
class-of-service {
  classifiers {
    ieee-802.1ad dot1p_dei_class {
      forwarding-class best-effort {
        loss-priority low code-points [ 0000 1101 ];
      }
    }
  }
}
```



```

    }
  }
}

```

```

class-of-service {
  interfaces {
    ge-2/0/0 {
      unit 0 {
        classifiers {
          ieee-802.1ad dot1p_dei_class;
        }
      }
    }
  }
}

```

## RELATED DOCUMENTATION

[Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 44](#)

[Applying Behavior Aggregate Classifiers to Logical Interfaces | 66](#)

## Configuring Rate Limits to Protect Lower Queues on IQ2 and Enhanced IQ2 PICs

You can rate-limit strict-high and high priority queues on IQ2 and IQ2E PICs. Without this limiting, traffic in higher priority queues can block the transmission of lower priority packets. Unless limited, higher priority traffic is always sent before lower priority traffic, causing the lower priority queues to “starve,” which in turn leads to timeouts and unnecessary resending of packets.

On the IQ2 and IQ2E PICs you can rate-limit queues before the packets are queued for output. All packets exceeding the configured rate limit are dropped, so care is required when establishing this limit.



**NOTE:** IQ2E PICs exclude the transmit rate of strict-high and high priority queues, thereby allowing low and medium priority queues to be configured up to 100 percent.



To rate-limit queues, include the `transmit-rate` statement with the `rate-limit` option at the `[edit class-of-service schedulers scheduler-name]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
transmit-rate rate rate-limit;
```

This example limits the transmit rate of a strict-high expedited-forwarding queue to 1 megabit per second (Mbps). The scheduler and scheduler map are defined and then applied to the traffic at the `[edit interfaces]` and `[edit class-of-service]` hierarchy levels:

```
[edit class-of-service]
schedulers {
  scheduler-1 {
    transmit-rate 1m rate-limit; # This establishes the limit
    priority strict-high;
  }
}
scheduler-maps {
  scheduler-map-1 {
    forwarding-class expedited-forwarding scheduler scheduler-1;
  }
}

[edit interfaces]
so-2/1/0 {
  per-unit-scheduler;
  encapsulation frame-relay;
  unit 0 {
    dlci 1;
  }
}

[edit class-of-service]
interfaces {
  so-2/1/0 {
    unit 0 {
      scheduler-map scheduler-map-1;
      shaping-rate 2m;
    }
  }
}
```



You can issue the following operational mode commands to verify your configuration (the first shows the rate limit in effect):

- `show class-of-service scheduler-map scheduler-map-name`
- `show class-of-service interface interface-name`

## RELATED DOCUMENTATION

[CoS on Enhanced IQ2 PICs Overview | 1005](#)

[Configuring Schedulers | 338](#)

## Simple Filters Overview

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

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

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



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



## RELATED DOCUMENTATION

[Configuring a Simple Filter](#) | 1058

## Configuring a Simple Filter

This simple filter sets the loss priority to low for TCP traffic with source address 10.1.1.1, sets the loss priority to high for HTTP (web) traffic with source addresses in the 203.0.113.0/24 range, and sets the loss priority to low for all traffic with destination address 10.6.6.6. The simple filter is applied as an input filter (arriving packets are checking for destination address 10.6.6.6, not queued output packets) on interface ge-0/0/1.0.

```
[edit]
firewall {
  family inet {
    simple-filter filter1 {
      term 1 {
        from {
          source-address {
            10.1.1.1/32;
          }
          protocol {
            tcp;
          }
        }
        then loss-priority low;
      }
      term 2 {
        from {
          source-address {
            203.0.113.0/24;
          }
          source-port {
            http;
          }
        }
        then loss-priority high;
      }
      term 3 {
        from {
```



```

        destination-address {
            10.6.6.6/32;
        }
    }
    then {
        loss-priority low;
        forwarding-class best-effort;
    }
}

}

}

}

interfaces {
    ge-0/0/1 {
        unit 0 {
            family inet {
                simple-filter {
                    input filter1;
                }
                address 10.1.2.3/30;
            }
        }
    }
}

}
```

## RELATED DOCUMENTATION

Simple Filters Overview | 1057



# Configuring Class of Service on 10-Gigabit Ethernet LAN/WAN PICs with SFP+

## IN THIS CHAPTER

- [CoS on 10-Gigabit Ethernet LAN/WAN PIC with SFP+ Overview | 1060](#)
- [BA and Fixed Classification on 10-Gigabit Ethernet LAN/WAN PIC with SFP+ Overview | 1061](#)
- [DSCP Rewrite for the 10-Gigabit Ethernet LAN/WAN PIC with SFP+ | 1062](#)
- [Configuring DSCP Rewrite for the 10-Gigabit Ethernet LAN/WAN PIC | 1065](#)
- [Queuing on 10-Gigabit Ethernet LAN/WAN PICs Properties | 1066](#)
- [Mapping Forwarding Classes to CoS Queues on 10-Gigabit Ethernet LAN/WAN PICs | 1067](#)
- [Scheduling and Shaping on 10-Gigabit Ethernet LAN/WAN PICs Overview | 1068](#)
- [Example: Configuring Shaping Overhead on 10-Gigabit Ethernet LAN/WAN PICs | 1069](#)

## CoS on 10-Gigabit Ethernet LAN/WAN PIC with SFP+ Overview

The 10-Gigabit Ethernet LAN/WAN PIC with SFP+ supports intelligent handling of oversubscribed traffic in applications, such as data centers and dense-core uplinks. The 10-Gigabit Ethernet LAN/WAN PIC with SFP+ supports line-rate operation for five 10-Gigabit Ethernet ports from each port group or a total WAN bandwidth of 100 Gbps with Packet Forwarding Engine bandwidth of 50 Gbps.



**NOTE:** This PIC has a front panel label with the designation “ETHERNET 10GBASE-SFP+ LAN-WAN” and can also be identified by its model number, PD-5-10XGE-SFPP. It is referred to hereafter as the 10-Gigabit Ethernet LAN/WAN PIC.

The class-of-service (CoS) configuration for the 10-Gigabit Ethernet LAN/WAN PICs are supported on standalone T640 and T1600 core routers, as well as T640 and T1600 routers in a routing matrix. The 10-Gigabit Ethernet LAN/WAN PICs support behavior aggregate (BA) and fixed classification, weighted round-robin scheduling with two queue priorities (low and strict-high), committed and peak information rate shaping on a per-queue basis, and excess information rate configuration for allocation of excess bandwidth.



To configure these features, include the corresponding class-of-service (CoS) statements at the [edit class-of-service] hierarchy level. The CoS statements supported on the 10-Gigabit Ethernet LAN/WAN PICs are shown in [Table 123 on page 1061](#).

**Table 123: CoS Statements Supported on the 10-Gigabit Ethernet LAN/WAN PICs**

| CoS Statements   | Supported |
|------------------|-----------|
| buffer-size      | No        |
| drop-profile-map | No        |
| excess-priority  | No        |
| excess-rate      | Yes       |
| priority         | Yes       |
| shaping-rate     | Yes       |
| transmit-rate    | Yes       |

#### RELATED DOCUMENTATION

[Junos OS Network Interfaces Library for Routing Devices](#)

## BA and Fixed Classification on 10-Gigabit Ethernet LAN/WAN PIC with SFP+ Overview

The 10-Gigabit Ethernet LAN/WAN PICs support the following behavior aggregate (BA) classifiers:

- DSCP, DSCP IPv6, or IP precedence—IP packet classification (Layer 3 headers)
- MPLS EXP—MPLS packet classification (Layer 2 headers)
- IEEE 802.1p—Packet classification (Layer 2 headers)



- IEEE 802.1ad—Packet classification for IEEE 802.1ad formats (including DEI bit)

Multiple classifiers can be configured to a single *logical interface*. However, there are some restrictions on which the classifiers can coexist. For example, the DSCP and IP precedence classifiers cannot be configured on the same logical interface. The DSCP and IP precedence classifiers can coexist with the DSCP IPv6 classifier on the same logical interface. An IEEE 802.1 classifier can coexist with other classifiers and is applicable only if a packet does not match any of the configured classifiers. For information about the supported combinations, see ["Applying Behavior Aggregate Classifiers to Logical Interfaces" on page 66](#).

If the classifiers are not defined explicitly, then the default classifiers are applied as follows:

- All MPLS packets are classified using the MPLS (EXP) classifier. If there is no explicit MPLS (EXP) classifier, then the default MPLS (EXP) classifier is applied.
- All IPv4 packets are classified using the IP precedence and DSCP classifiers. If there is no explicit IP precedence or DSCP classifier, then the default IP precedence classifier is applied.
- All IPv6 packets are classified using a DSCP IPv6 classifier. If there is no explicit DSCP IPv6 classifier, then the default DSCP IPv6 classifier is applied.
- If the IEEE 802.1p classifier is configured and a packet does not match any explicitly configured classifier, then the IEEE 802.1p classifier is applied.

The fixed classification matches the traffic on a logical interface level. The following example classifies all traffic on logical unit zero to the queue corresponding to assured forwarding.

```
[edit class-of-service interfaces xe-0/1/2 unit 0]
forwarding-class fc-af11;
```



**NOTE:** The 10-Gigabit Ethernet LAN/WAN PICs do not support multifield classification. However, the multifield classification can be done at the Packet Forwarding Engine using the firewall filters, which overrides the classification done at the PIC level. The multifield classification at the Packet Forwarding Engine occurs after the PIC handles the oversubscribed traffic.

## DSCP Rewrite for the 10-Gigabit Ethernet LAN/WAN PIC with SFP+

The 10-Gigabit Ethernet LAN/WAN PIC with SFP+ (Model Number: PD-5-10XGE-SFPP) in T640 and T1600 standalone routers and TX Matrix and TX Matrix Plus routing matrices supports 6-bit DSCP rewrite (IPv4 and IPv6) functionality. The following DSCP rewrite features are supported:



- Full 6-bit DSCP rewrite
- Independent rewrite for DSCPv4 and DSCPv6 simultaneously on the same *logical interface*
- Four tables per PIC for DSCPv4 and DSCPv6, respectively
- Rewrite based on queue number rather than forwarding class. Queues are mapped to a forwarding class by using the global forwarding-class configuration on the router.
- Ability to bind multiple (maximum of all) logical interfaces on the PIC to the same rewrite table.
- Ability of DSCP rewrite on the PIC to configure, by default, code-point 000000 if you do not specify a classifier in the `rewrite-rules` statement.



**NOTE:** The 10-Gigabit Ethernet LAN/WAN PIC with SFP+ (P/N: PD-5-10XGE-SFPP), when used in T640 and T1600 standalone routers, and T640 and T1600 routers in TX Matrix and TX Matrix Plus routing matrices, has the following known limitations:

- DSCP rewrite on the PIC does not support distinct DSCP code-point rewrites if multiple forwarding classes (FC) are configured to map to the same queue in the “forwarding-class” configuration.
- The PIC can perform DSCP rewrite based on three PLP values, unlike four PLP values by the Packet Forwarding Engine.
- The protocol option is not supported in the following DSCP rewrite rule configuration:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
rewrite-rules {
    dscp (rewrite-name | <default>) protocol <protocol-types>;
}
```

- The PIC has the ability to parse a packet with up to two VLAN tags. However, the following conditions apply when DSCP rewrite is enabled:
  - The PIC supports DSCP rewrite only for untagged and single VLAN tagged packets.
  - For DSCP rewrite in conjunction with VLAN rewrite push operations, the PIC can push only one tag if the packet is untagged.



- If the packet has more than one VLAN tag (either because it was double tagged or because additional tags were pushed as part of a VLAN rewrite), then DSCP rewrite is not executed.
- Configuration of DSCP *rewrite rules* on the PIC overwrites the DSCP value coming from the Routing Engine for host-generated traffic. The behavior is as follows:
  - If the packet's forwarding class and packet loss priority (PLP) match the DSCP rewrite rule on the PIC, then the DSCP code-point rewritten by the *host-outbound-traffic* statement is overwritten by the PIC's DSCP rewrite with the corresponding DSCP code-point configured in the rewrite rule.
  - If the packet's forwarding class and PLP do not match any DSCP rewrite rule on the PIC, then the DSCP code-point rewritten by the *host-outbound-traffic* statement is overwritten by the PIC's DSCP rewrite as 6b'000000.

This behavior is different from DSCP rewrites done in the Packet Forwarding Engine for other PICs. In those cases, the Packet Forwarding Engine processing is bypassed for host-generated packets and hence the DSCP set in the Routing Engine for host-generated packets is not overwritten in the Packet Forwarding Engine or PIC.

- If multiple forwarding classes map to the same queue, then the last forwarding class that maps to the same queue is picked and its code-point is used for DSCP rewrite.
- If both medium-high and medium-low PLP values are configured in the rewrite rule and if their rewrite code-points are different, then the code-point associated with medium-high is used for rewrite for both medium-high and medium-low packets on that logical interface. If only one of the PLP values (either medium-high or medium-low) is configured, then its corresponding code-point is used for rewrite for both medium-high and medium-low packets on that logical interface.



**NOTE:** A system error message can result if a configuration that conflicts with these limitations is committed or used .

## RELATED DOCUMENTATION

[Configuring DSCP Rewrite for the 10-Gigabit Ethernet LAN/WAN PIC | 1065](#)

*dscp*

*dscp-ipv6*

*forwarding-class*



*rewrite-rules*

[Understanding DSCP Classification for VPLS | 89](#)

[Default DSCP and DSCP IPv6 Classifiers | 49](#)

## Configuring DSCP Rewrite for the 10-Gigabit Ethernet LAN/WAN PIC

To configure DSCP rewrite, use the `rewrite-rules` statement at the class-of-service interfaces *interface-name* unit *logical-unit-number* hierarchy level, as shown in the following configuration example:

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
rewrite-rules {
  dscp (rewrite-name | <default>);
  dscp-ipv6 (rewrite-name | <default>);
  exp (rewrite-name | <default>) protocol <protocol-types>;
  exp-push-push-push <default>;
  exp-swap-push-push <default>;
  ieee-802.1 (rewrite-name | <default>) vlan-tag (outer | outer-and-inner);
  inet-precedence (rewrite-name | <default>) protocol <protocol-types>;
}
```

To configure DSCP rewrite rules, use the `rewrite-rules` statement's (<dscp> | <dscp-ipv6>) option's subordinate rewrite rules statements at the edit `class-of-service` hierarchy level, as shown in the following configuration example:

```
[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 level code-point (alias | bits);
    }
  }
}
```

### RELATED DOCUMENTATION

[DSCP Rewrite for the 10-Gigabit Ethernet LAN/WAN PIC with SFP+ | 1062](#)



[\*dscp\*](#)[\*dscp-ipv6\*](#)[\*forwarding-class\*](#)[\*rewrite-rules\*](#)[Understanding DSCP Classification for VPLS | 89](#)[Default DSCP and DSCP IPv6 Classifiers | 49](#)

## Queuing on 10-Gigabit Ethernet LAN/WAN PICs Properties

The 10-Gigabit Ethernet LAN/WAN PICs have the following features to support queuing:

- Committed and peak information rate shaping on a per-queue basis
- Excess information rate configuration for allocation of excess bandwidth
- Ingress queuing based on behavior aggregate (BA) classification
- Egress queuing at the Packet Forwarding Engine and at the PIC level

The Packet Forwarding Engine egress queues are shared by two physical interfaces in a port group.

- Weighted round-robin (WRR) scheduling with two queue priorities (low and strict-high)
- Two special queues available in ingress, one per physical interface, called *control queues*

Layer 2 and Layer 3 control protocol packets (OSPF, OSPF3, VRRP, IGMP, RSVP, PIM, BGP, BFD, LDP, ISIS, RIP, RIPv6, LACP, ARP, IPv6 NDP, CFM, and LFM) are mapped to the control queue. In the control queue, these packets are not dropped even if there is oversubscription or congestion on a port group.



**NOTE:** The control queue is rate-limited to 2 Mbps per physical interface. The packets in excess of 2 Mbps are dropped and accounted for.

## RELATED DOCUMENTATION

[Mapping Forwarding Classes to CoS Queues on 10-Gigabit Ethernet LAN/WAN PICs | 1067](#)



## Mapping Forwarding Classes to CoS Queues on 10-Gigabit Ethernet LAN/WAN PICs

The 10-Gigabit Ethernet LAN/WAN PICs support eight CoS queues per port in the egress direction. To map forwarding classes to the eight CoS queues in egress, include the following statements at the [edit class-of-service] hierarchy level:

```
[edit class-of-service forwarding-classes] {
  class fc-be queue-num 0;
  class fc-be1 queue-num 1;
  class fc-ef queue-num 2;
  class fc-ef1 queue-num 3;
  class fc-af11 queue-num 4;
  class fc-af12 queue-num 5;
  class fc-nc1 queue-num 6;
  class fc-nc2 queue-num 7;
}
```



**CAUTION:** 10-Gigabit Ethernet LAN/WAN PICs do not support more than eight forwarding classes. If you define more than eight forwarding classes, excess forwarding classes can get mapped to queues with undefined schedulers.

The 10-Gigabit Ethernet LAN/WAN PICs support four ingress queues per physical interface. The PICs use restricted-queues configuration to map multiple forwarding classes to the four queues. There are no queues at the logical interface level. In the following example, two forwarding classes are mapped to one queue.

```
[edit class-of-service restricted-queues] {
  forwarding-class fc-be queue-num 0;
  forwarding-class fc-be1 queue-num 0;
  forwarding-class fc-ef queue-num 1;
  forwarding-class fc-ef1 queue-num 1;
  forwarding-class fc-af11 queue-num 2;
  forwarding-class fc-af12 queue-num 2;
  forwarding-class fc-nc1 queue-num 3;
  forwarding-class fc-nc2 queue-num 3;
}
```



RELATED DOCUMENTATION

|                                                                                            |
|--------------------------------------------------------------------------------------------|
| <a href="#">Queuing on 10-Gigabit Ethernet LAN/WAN PICs Properties   1066</a>              |
| <a href="#">Understanding How Forwarding Classes Assign Classes to Output Queues   275</a> |
| <a href="#">Configuring a Custom Forwarding Class for Each Queue   281</a>                 |
| <i>forwarding-classes</i>                                                                  |

## Scheduling and Shaping on 10-Gigabit Ethernet LAN/WAN PICs Overview

The 10-Gigabit Ethernet LAN/WAN PIC has ten 10-Gigabit Ethernet ports providing 100 Gbps of WAN bandwidth and 50 Gbps of Packet Forwarding Engine bandwidth. On the ingress side of the 10-Gigabit Ethernet LAN/WAN PIC, two consecutive physical interfaces on the PICs are grouped together into a port group and are serviced by a single scheduler. The port groups are as shown in [Table 124 on page 1068](#):

**Table 124: Port Groups on 10-Gigabit Ethernet LAN/WAN PICs**

| Port Group | Mapped Ports                               |
|------------|--------------------------------------------|
| Group 1    | <i>xe-fpc/pic/0</i><br><i>xe-fpc/pic/1</i> |
| Group 2    | <i>xe-fpc/pic/2</i><br><i>xe-fpc/pic/3</i> |
| Group 3    | <i>xe-fpc/pic/4</i><br><i>xe-fpc/pic/5</i> |
| Group 4    | <i>xe-fpc/pic/6</i><br><i>xe-fpc/pic/7</i> |



**Table 124: Port Groups on 10-Gigabit Ethernet LAN/WAN PICs (Continued)**

| Port Group | Mapped Ports                               |
|------------|--------------------------------------------|
| Group 5    | <i>xe-fpc/pic/8</i><br><i>xe-fpc/pic/9</i> |

The two physical interfaces in a port group share 10 Gbps bandwidth towards the Packet Forwarding Engine. A scheduler has eight class-of-service (CoS) queues and two control queues. On the ingress side of the 10-Gigabit Ethernet LAN/WAN PIC, the eight CoS queues are split four plus four for the two physical interfaces. Thus, the 10-Gigabit Ethernet LAN/WAN PIC supports four ingress queues and eight egress queues per physical interface.

At the ingress side of the 10-Gigabit Ethernet LAN/WAN PIC, multiple forwarding classes can be mapped to one queue using the restricted-queue configuration. When creating a scheduler-map for the ingress queues, only one forwarding class should be chosen from the multiple forwarding classes that map to the same queue. Then, the scheduler-map can be specified using the `set class-of-service scheduler-maps map-name forwarding-class class-name scheduler scheduler` command.

The 10-Gigabit Ethernet LAN/WAN PICs manage packet buffering internally and no configuration is required.



**NOTE:** The delay-bandwidth buffering configuration is not supported on the 10-Gigabit Ethernet LAN/WAN PICs.

## Example: Configuring Shaping Overhead on 10-Gigabit Ethernet LAN/WAN PICs

By default, the 10-Gigabit Ethernet LAN/WAN PIC uses 20 bytes as the shaping overhead. This includes 8 bytes preamble and 12 bytes interpacket gap (IPG) in shaper operations. To exclude this overhead, it should be configured as -20 bytes. The shaping overhead value can be set between 0 and 31 bytes, as shown in the following example. This range translates to a CLI range of -20 to 11 bytes for the shaping overhead configuration.

```
show chassis
  fpc 6 {
    pic 0 {
      traffic-manager {
```



```
    ingress-shaping-overhead -20;  
    egress-shaping-overhead -20;  
  }  
}
```



**NOTE:** When the configuration for the overhead bytes on a PIC are changed, the PIC is taken offline and then brought back online. In addition, the configuration in the CLI is on a per-PIC basis, and thus, applies to all the ports on the PIC.

## RELATED DOCUMENTATION

| [Scheduling and Shaping on 10-Gigabit Ethernet LAN/WAN PICs Overview](#) | 1068



# Configuring Class of Service on Enhanced Queuing DPCs

## IN THIS CHAPTER

- [Enhanced Queuing DPC CoS Properties | 1071](#)
- [Configuring Rate Limits on Enhanced Queuing DPCs | 1074](#)
- [Configuring WRED on Enhanced Queuing DPCs | 1076](#)
- [Configuring MDRR on Enhanced Queuing DPCs | 1078](#)
- [Configuring Excess Bandwidth Sharing | 1081](#)
- [Configuring Customer VLAN \(Level 3\) Shaping on Enhanced Queuing DPCs | 1086](#)
- [Simple Filters Overview | 1089](#)
- [Configuring Simple Filters on Enhanced Queuing DPCs | 1090](#)
- [Configuring a Simple Filter | 1092](#)

## Enhanced Queuing DPC CoS Properties

On a Juniper Networks MX Series 5G Universal Routing Platform with Enhanced Queuing Dense Port Concentrators (DPCs), you can configure schedulers and queues. You can configure 15 VLAN sets per Gigabit Ethernet (1G) port and 255 VLAN sets per 10-Gigabit Ethernet (10G) port. The Enhanced Queuing DPC performs priority propagation from one hierarchy level to another and drop statistics are available on the Enhanced Queuing DPC per color per queue instead or just per queue.



**NOTE:** The Enhanced Queuing DPC (EQ DPC) does not support BA classification for packets received from a Layer 3 routing interface or a virtual routing and forwarding (VRF) interface and routed to an integrated routing and bridging interface (IRB) to reach the remote end of a pseudowire connection. The EQ DPC also does not support BA classification for Layer 2 frames received from a Virtual Private LAN Service (VPLS) pseudowire connection from a remote site and routed to a Layer 3 routing interface through an IRB interface.



Juniper Networks MX Series 5G Universal Routing Platforms with Enhanced Queuing DPCs have Packet Forwarding Engines that can support up to 515 MB of frame memory, and packets are stored in 512-byte frames. [Table 125 on page 1072](#) compares the major properties of the Intelligent Queuing 2 (IQ2) PIC and the Packet Forwarding Engine within the Enhanced Queuing DPC.

**Table 125: IQ2 PIC and Enhanced Queuing DPC Compared**

| Feature                             | IQ2 PIC                        | Packet Forwarding Engine Within Enhanced Queuing DPC   |
|-------------------------------------|--------------------------------|--------------------------------------------------------|
| Number of usable queues             | 8,000                          | 16,000                                                 |
| Number of shaped logical interfaces | 1,000 with 8 queues each.      | 2,000 with 8 queues each, or 4,000 with 4 queues each. |
| Number of hardware priorities       | 2                              | 4                                                      |
| Priority propagation                | No                             | Yes                                                    |
| Dynamic mapping                     | No: schedulers/port are fixed. | Yes: schedulers/port are not fixed.                    |
| Drop statistics                     | Per queues                     | Per queue per color (PLP high, low)                    |

In addition, the Enhanced Queuing DPC features support for hierarchical weighted random early detection (WRED) and enhanced queuing on aggregated Ethernet interfaces with link protection as well.

The Enhanced Queuing DPC supports the following hierarchical scheduler characteristics:

- Shaping at the physical interface level
- Shaping and scheduling at the service VLAN interface set level
- Shaping and scheduling at the customer VLAN *logical interface* level
- Scheduling at the queue level

VLAN (Level 3) shaping on a 10-Gigabit Ethernet MX Series Enhanced Queuing DPC differs from the VLAN (Level3) shaping on a 1-Gigabit Ethernet Enhanced Queuing DPC. To use the VLAN (Level 3) shaping on a 10-Gigabit Ethernet MX Series Enhanced Queuing DPC, configure an interface set at the [edit interfaces interface-set] hierarchy level. The interface set configuration is not required for configuring a 1-Gigabit Ethernet VLANs on the same Enhanced Queuing DPC.

The Enhanced Queuing DPC supports the following features for scalability:



- 16,000 queues per Packet Forwarding Engine
- 4 Packet Forwarding Engines per DPC
  - 4000 schedulers at logical interface level (Level 3) with 4 queues each
  - 2000 schedulers at logical interface level (Level 3) with 8 queues each
- 255 schedulers at the interface set level (Level 2) per 1-port Packet Forwarding Engine on a 10-Gigabit Ethernet DPC
- 15 schedulers at the interface set level (Level 2) per 10-port Packet Forwarding Engine on a 1-Gigabit Ethernet DPC
- About 400 milliseconds of buffer delay (this varies by packet size and if large buffers are enabled)
- 4 levels of priority (strict-high, high, medium, and low)



**NOTE:** Including the transmit-rate *rate* exact statement at the [edit class-of-service schedulers *scheduler-name*] hierarchy level is not supported on Enhanced Queuing DPCs on MX Series routers.

The way that the Enhanced Queuing DPC maps a queue to a scheduler depends on whether 8 queues or 4 queues are configured. By default, a scheduler at level 3 has 4 queues. Level 3 scheduler X controls queue  $X*4$  to  $X*4+3$ , so that scheduler 100 (for example) controls queues 400 to 403. However, when 8 queues per scheduler are enabled, the odd numbered schedulers are disabled, allowing twice the number of queues per subscriber as before. With 8 queues, level 3 scheduler X controls queue  $X*4$  to  $X*4+7$ , so that scheduler 100 (for example) now controls queues 400 to 407.

You configure the `max-queues-per-interface` statement to set the number of queues at 4 or 8 at the FPC level of the hierarchy. Changing this statement results in a restart of the DPC. For more information about the `max-queues-per-interface` statement, see the [Junos OS Network Interfaces Library for Routing Devices](#).

The Enhanced Queuing DPC maps level 3 (customer VLAN) schedulers in groups to level 2 (service VLAN) schedulers. Sixteen contiguous level 3 schedulers are mapped to level 2 when 4 queues are enabled, and 8 contiguous level 3 schedulers are mapped to level 2 when 8 queues are enabled. All of the schedulers in the group should use the same queue priority mapping. For example, if the queue priorities of one scheduler are high, medium, low, and low, then all members of this group should have the same queue priority.

Mapping of a group at level 3 to level 2 can be done at any time. However, a group at level 3 can only be unmapped from a level 2 scheduler only if all the schedulers in the group are free. Once unmapped, a level 3 group can be remapped to any level 2 scheduler. There is no restriction on the number of level 3 groups that can be mapped to a particular level 2 scheduler. There can be 256 level 3 groups, but



fragmentation of the scheduler space can reduce the number of schedulers available. In other words, there are scheduler allocation patterns that might fail even though there are free schedulers.

In contrast to level-3-to-level-2 mapping, the Enhanced Queuing DPC maps level 2 (service VLAN) schedulers in a fixed mode to level 1 (physical interface) schedulers. On 40-port Gigabit Ethernet DPCs, there are 16 level 1 schedulers, and 10 of these are used for the physical interfaces. There are 256 level 2 schedulers, or 16 per level 1 scheduler. A level 1 scheduler uses level 2 schedulers  $X \times 16$  through  $X \times 16 + 15$ . So level 1 scheduler 0 uses level 2 schedulers 0 through 15, level 1 scheduler 1 uses level 2 schedulers 16 through 31, and so on. On 4-port 10-Gigabit Ethernet PICs, there is one level 1 scheduler for the physical interface, and 256 level 2 schedulers are mapped to the single level 1 scheduler.

The maximum number of level 3 (customer VLAN) schedulers that can be used is 4076 (4 queues) or 2028 (8 queues) for the 10-port Gigabit Ethernet Packet Forwarding Engine and 4094 (4 queues) or 2046 (8 queues) for the 10-Gigabit Ethernet Packet Forwarding Engine.

Enhanced Queuing is supported on aggregated Ethernet (AE) interfaces with two links in link protection mode. However, only one link in the AE bundle can be active at a time. Traffic is shaped independently on the two links, but the member's links do not need to reside in the same Packet Forwarding Engine or the same DPC. Finally, shared schedulers are not supported on the Enhanced Queuing DPC (use hierarchical schedulers to group logical interfaces).

## RELATED DOCUMENTATION

[Configuring Customer VLAN \(Level 3\) Shaping on Enhanced Queuing DPCs](#) | 1086

## Configuring Rate Limits on Enhanced Queuing DPCs

You can rate-limit the strict-high and high queues on the Enhanced Queuing DPC. Without rate limits, traffic in higher-priority queues can block the transmission of lower-priority packets. Unless limited, higher-priority traffic is always sent before lower-priority traffic, causing the lower-priority queues to “starve” and cause timeouts and unnecessarily resent packets.

On the Enhanced Queuing DPC, you can rate-limit queues before the packets are queued for output. All packets exceeding the configured rate limit are dropped, so care is required when establishing this limit. This model is also supported on IQ2 PICs. For more information about configuring CoS on IQ2 PICs, see ["CoS on Enhanced IQ2 PICs Overview" on page 1005](#).



**NOTE:** Rate limiting is implemented differently on Enhanced Queuing DPCs and non-queuing Packet Forwarding Engines. On Enhanced Queuing DPCs, rate limiting is



implemented using a single-rate two-color policer. On nonqueuing Packet Forwarding Engines, rate limiting is achieved by shaping the queue to the transmit rate and keeping the queue delay buffers small to prevent too many packets from being queued after the shaping rate is reached.

To rate-limit queues, include the `transmit-rate` statement with the `rate-limit` option at the `[edit class-of-service schedulers scheduler-name]` hierarchy level:

```
[edit class-of-service schedulers scheduler-name]
transmit-rate rate rate-limit;
```

The following example limits the transmit rate of a strict-high expedited-forwarding queue to 1 Mbps. The scheduler and scheduler map are defined, and then applied to the traffic at the `[edit interfaces]` and `[edit class-of-service]` hierarchy levels:

```
[edit class-of-service]
schedulers {
    scheduler-1 {
        transmit-rate 1m rate-limit; # This establishes the limit
        priority strict-high;
    }
}
scheduler-maps {
    scheduler-map-1 {
        forwarding-class expedited-forwarding scheduler scheduler-1;
    }
}

[edit interfaces]
so-2/2/0 {
    per-unit-scheduler;
    encapsulation frame-relay;
    unit 0 {
        dlci 1;
    }
}

[edit class-of-service]
interfaces {
    so-2/2/0 {
```



```

    unit 0 {
        scheduler-map scheduler-map-1;
        shaping-rate 2m;
    }
}

```

You can issue the following operational mode commands to verify your configuration (the first shows the rate limit in effect):

- `show class-of-service scheduler-map scheduler-map-name`
- `show class-of-service interface interface-name`

You can issue the `show interfaces queue interface-name` command to view the number of packets dropped at an interface. The output of the `show interfaces queue interface-name` command always displays the rate-limit counter fields whether or not rate limiting is configured on the queue. Rate-limit counters are displayed in two columns. The first column is the consolidated count of the packets dropped and the second column is the real-time count of the packets dropped.

Rate-limit packet drop counters display the value 0 when rate limiting is not configured on the queue or when the queue does not have rate-limit packet drops even with rate limiting configured.

Rate-limit packet drop counters display meaningful values in both columns when the queue has rate-limit packet drops. However, when rate limiting is not happening in real time but has occurred earlier, the first column displays the consolidated count and the second column displays the value 0.

You can clear the packet drop statistics by using the `clear interface statistics interface-name` command.

## Configuring WRED on Enhanced Queuing DPCs

Shaping to drop out-of-profile traffic is done on the Enhanced Queuing DPC at all levels but the queue level. However, weighed random early detection (WRED) is done at the queue level with much the same result. With WRED, the decision to drop or send the packet is made before the packet is placed in the queue.

WRED shaping on the Enhanced Queuing DPC is similar to the IQ2 PIC, but involves only two levels, not 64. The probabilistic drop region establishes a minimum and a maximum queue depth. Below the minimum queue depth, the drop probability is 0 (send). Above the maximum level, the drop probability is 100 (certainty).



There are four drop profiles associated with each queue. These correspond to each of four loss priorities (low, medium-low, medium-high, and high). Sixty-four sets of four drop profiles are available (32 for ingress and 32 for egress). In addition, there are eight WRED scaling profiles in each direction.

To configure WRED, include the `drop-profiles` statement at the `[edit class-of-service]` hierarchy level:

```
[edit class-of-service]
drop-profiles {
  profile-name {
    fill-level percentage drop-probability percentage;
  }
}
```

The following example is an Enhanced Queuing DPC drop profile for expedited forwarding traffic:

```
[edit class-of-service drop-profiles]
drop-ef {
  fill-level 20 drop-probability 0; # Minimum Q depth
  fill-level 100 drop-probability 100; # Maximum Q depth
}
```

Note that only two fill levels can be specified for the Enhanced Queuing DPC. You can configure the `interpolate` statement, but only two fill levels are used. The `delay-buffer-rate` statement in the traffic control profile determines the maximum queue size. This delay buffer rate is converted to a packet delay buffers, where one buffer is equal to 512 bytes. For example, at 10 Mbps, the Enhanced Queuing DPC allocates 610 delay buffers when the delay-buffer rate is set to 250 milliseconds. The WRED threshold values are specified in terms of absolute buffer values.

The WRED scaling factor multiplies all WRED thresholds (both minimum and maximum) by the value specified. There are eight values in all: 1, 2, 4, 8, 16, 32, 64, and 128. The WRED scaling factor is chosen to best match the user-configured drop profiles. This is done because the hardware supports only certain values of thresholds (all values must be a multiple of 16). So if the configured value of a threshold is 500 (for example), the multiple of 16 is 256 and the scaling factor applied is 2, making the value 512, which allows the value of 500 to be used. If the configured value of a threshold is 1500, the multiple of 16 is 752 and the scaling factor applied is 2, making the value 1504, which allows the value of 1500 to be used.

Hierarchical RED is used to support the oversubscription of the delay buffers (WRED is only configured at the queue, physical interface, and PIC level). Hierarchical RED works with WRED as follows:

- If any level accepts the packet (the queue depth is less than the minimum buffer level), then this level accepts the packet.



- If any level probabilistically drops the packet, then this level drops the packet.

However, these rules might lead to the accepting of packets under loaded conditions which might otherwise have been dropped. In other words, the logical interface accepts packets if the physical interface is not congested.

## RELATED DOCUMENTATION

[Shaping Granularity Values for Enhanced Queuing Hardware | 1010](#)

[RED Drop Profiles for Congestion Management | 464](#)

## Configuring MDRR on Enhanced Queuing DPCs

The guaranteed rate (CIR) at the interface set level is implemented using modified deficit round-robin (MDRR). The Enhanced Queuing DPC hardware provides four levels of strict priority. There is no restriction on the number of queues for each priority. MDRR is used among queues of the same priority. Each queue has one priority when it is under the guaranteed rate and another priority when it is over the guaranteed rate but under the shaping rate (PIR). The Enhanced Queuing DPC hardware implements the priorities with 256 service profiles. Each service profile assigns eight priorities for eight queues. One set is for logical interfaces under the guaranteed rate and another set is for logical interfaces over the guaranteed rate but under the shaping rate. Each service profile is associated with a group of 16 level 3 schedulers, so there is a unique service profile available for all 256 groups at level 3, giving 4096 logical interfaces.

The Junos OS provides three priorities for traffic under the guaranteed rate and one reserved priority for traffic over the guaranteed rate that is not configurable. The Junos OS provides three priorities when there is no guaranteed rate configured on any logical interface.

The relationship between Junos OS priorities and the Enhanced Queuing DPC hardware priorities below and above the guaranteed rate (CIR) is shown in [Table 126 on page 1078](#).

**Table 126: Junos OS Priorities Mapped to Enhanced Queuing DPC Hardware Priorities**

| Junos OS Priority | Enhanced Queuing DPC Hardware Priority<br>Below Guaranteed Rate | Enhanced Queuing DPC Hardware<br>Priority Above Guaranteed Rate |
|-------------------|-----------------------------------------------------------------|-----------------------------------------------------------------|
| Strict-high       | High                                                            | High                                                            |



**Table 126: Junos OS Priorities Mapped to Enhanced Queuing DPC Hardware Priorities (*Continued*)**

| Junos OS Priority | Enhanced Queuing DPC Hardware Priority<br>Below Guaranteed Rate | Enhanced Queuing DPC Hardware<br>Priority Above Guaranteed Rate |
|-------------------|-----------------------------------------------------------------|-----------------------------------------------------------------|
| High              | High                                                            | Low                                                             |
| Medium-high       | Medium-high                                                     | Low                                                             |
| Medium-low        | Medium-high                                                     | Low                                                             |
| Low               | Medium-low                                                      | Low                                                             |

To configure MDRR, configure a scheduler at the [edit class-of-service schedulers] hierarchy level:

```
[edit class-of-service schedulers]
scheduler-name {
    buffer-size (seconds | percent percentage | remainder | temporal microseconds);
    priority priority-level;
    transmit-rate (percent percentage | rate | remainder) <exact | rate-limit>;
}
```

The following example creates two schedulers for MDRR:

```
[edit class-of-service schedulers]
best-effort-scheduler {
    transmit-rate percent 30; # if no shaping rate
    buffer-size percent 30;
    priority high;
}
expedited-forwarding-scheduler {
    transmit-rate percent 40; # if no shaping rate
    buffer-size percent 40;
    priority strict-high;
}
```





**NOTE:** The use of both shaping rate and a guaranteed rate at the interface set level (level 2) is not supported.

MDRR is provided at three levels of the scheduler hierarchy of the Enhanced Queuing DPC with a granularity of 1 through 255. There are 64 MDRR profiles at the queue level, 16 at the interface set level, and 32 at the physical interface level.

Queue transmit rates are used for queue level MDRR profile weight calculation. The queue MDRR weight is calculated differently based on the mode set for sharing excess bandwidth. If you configure the equal option for excess bandwidth, then the queue MDRR weight is calculated as:

$$\text{Queue weight} = (255 * \text{Transmit-rate-percentage}) / 100$$

If you configure the proportional option for excess bandwidth, which is the default, then the queue MDRR weight is calculated as:

Queue weight = Queue-transmit-rate / Queue-base-rate, where

Queue-transmit-rate = (Logical-interface-rate \* Transmit-rate-percentage) / 100, and

Queue-base-rate = Excess-bandwidth-proportional-rate / 255

To configure the way that the Enhanced Queuing DPC should handle excess bandwidth, configure the excess-bandwidth-share statement at the [edit interface-set *interface-set-name*] hierarchy level. By default, the excess bandwidth is set to proportional with a default value of 32.64 Mbps. In this mode, the excess bandwidth is shared in the ratio of the logical interface shaping rates. If set to equal, the excess bandwidth is shared equally among the logical interfaces.

This example sets the excess bandwidth sharing to proportional at a rate of 100 Mbps with a shaping rate of 80 Mbps.

```
[edit interface-set example-interface-set]
excess-bandwidth-share proportional 100m;
output-traffic-control-profile PIR-80Mbps;
```

Shaping rates established at the logical interface level are used to calculate the MDRR weights used at the interface set level. The 16 MDRR profiles are set to initial values, and the closest profile with rounded values is chosen. By default, the physical port MDRR weights are preset to the full bandwidth on the interface.



# Configuring Excess Bandwidth Sharing

## IN THIS SECTION

- [Excess Bandwidth Sharing and Minimum Logical Interface Shaping | 1081](#)
- [Selecting Excess Bandwidth Sharing Proportional Rates | 1082](#)
- [Mapping Calculated Weights to Hardware Weights | 1082](#)
- [Allocating Weight with Only Shaping Rates or Unshaped Logical Interfaces | 1083](#)
- [Sharing Bandwidth Among Logical Interfaces | 1084](#)

When using the Enhanced Queuing DPC on an MX Series router, there are circumstances when you should configure excess bandwidth sharing and minimum logical interface shaping. This section details some of the guidelines for configuring excess bandwidth sharing.

## Excess Bandwidth Sharing and Minimum Logical Interface Shaping

The default excess bandwidth sharing proportional rate is 32.65 Mbps (128 Kbps x 255). In order to have better weighed fair queuing (WFQ) accuracy among queues, the shaping rate configured should be larger than the excess bandwidth sharing proportional rate. Some examples are shown in [Table 127 on page 1081](#).

**Table 127: Shaping Rates and WFQ Weights**

| Shaping Rate | Configured Queue Transmit Rate | WFQ Weight        | Total Weights |
|--------------|--------------------------------|-------------------|---------------|
| 10 Mbps      | (30, 40, 25, 5)                | (22, 30, 20, 4)   | 76            |
| 33 Mbps      | (30, 40, 25, 5)                | (76, 104, 64, 13) | 257           |
| 40 Mbps      | (30, 40, 25, 5)                | (76, 104.64, 13)  | 257           |

With a 10-Mbps shaping rate, the total weights are 76. This is divided among the four queues according to the configured transmit rate. Note that when the shaping rate is larger than the excess bandwidth sharing proportional rate of 32.65 Mbps, the total weights on the logical interface are 257 and the WFQ accuracy is the same.



## Selecting Excess Bandwidth Sharing Proportional Rates

A good excess bandwidth sharing proportional rate to configure is to choose the largest CIR (guaranteed rate) among all the logical interfaces (units). If the logical units have PIRs (shaping rates) only, then choose the largest PIR rate. However, this is not ideal if a single logical interface has a large weighed round-robin (WRR) rate. This can skew the distribution of traffic across the queues of the other logical interfaces. To avoid this issue, set the excess bandwidth sharing proportional rate to a lower value on the logical interfaces where the WRR rates are concentrated. This improves the bandwidth sharing accuracy among the queues on the same logical interface. However, the excess bandwidth sharing for the logical interface with the larger WRR rate is no longer proportional.

As an example, consider five logical interfaces on the same physical port, each with four queues, all with only PIRs configured and no CIRs. The WRR rate is the same as the PIR for the logical interface. The excess bandwidth is shared proportionally with a rate of 40 Mbps. The traffic control profiles for the logical interfaces are shown in [Table 128 on page 1082](#).

**Table 128: Sample Shaping Rates and WFQ Weights**

| Shaping Rate      | Configured Queue Transmit Rate | WFQ Weight        | Total Weights |
|-------------------|--------------------------------|-------------------|---------------|
| (Unit 0) 10 Mbps  | (95, 0, 0, 5)                  | (60, 0, 0, 3)     | 63            |
| (Unit 1) 20 Mbps  | (25, 25, 25, 25)               | (32, 32, 32, 32)  | 128           |
| (Unit 2) 40 Mbps  | (40, 30, 20, 10)               | (102, 77, 51, 26) | 255           |
| (Unit 3) 200 Mbps | (70, 10, 10, 10)               | (179, 26, 26, 26) | 255           |
| (Unit 4) 2 Mbps   | (25, 25, 25, 25)               | (5, 5, 5, 5)      | 20            |

Even though the maximum transmit rate for the queue on logical interface unit 3 is 200 Mbps, the excess bandwidth sharing proportional rate is kept at a much lower value. Within a logical interface, this method provides a more accurate distribution of weights across queues. However, the excess bandwidth is now shared equally between unit 2 and unit 3 (total weight of each = 255).

## Mapping Calculated Weights to Hardware Weights

The calculated weight in a traffic control profile is mapped to hardware weight, but the hardware only supports a limited WFQ profile. The weights are rounded to the nearest hardware weight according to the values in [Table 129 on page 1083](#).



**Table 129: Rounding Configured Weights to Hardware Weights**

| Traffic Control Profile Number | Number of Traffic Control Profiles | Weights                  | Maximum Error |
|--------------------------------|------------------------------------|--------------------------|---------------|
| 1–16                           | 16                                 | 1–16 (interval of 1)     | 50.00%        |
| 17–29                          | 13                                 | 18–42 (interval of 2)    | 6.25%         |
| 30–35                          | 6                                  | 45–60 (interval of 3)    | 1.35%         |
| 36–43                          | 8                                  | 64–92 (interval of 4)    | 2.25%         |
| 44–49                          | 6                                  | 98–128 (interval of 6)   | 3.06%         |
| 50–56                          | 7                                  | 136–184 (interval of 8)  | 3.13%         |
| 57–62                          | 6                                  | 194–244 (interval of 10) | 2.71%         |
| 63–63                          | 1                                  | 255–255 (interval of 11) | 2.05%         |

From the table, as an example, the calculated weight of 18.9 is mapped to a hardware weight of 18, because 18 is closer to 18.9 than 20 (an interval of 2 applies in the range 18–42).

### Allocating Weight with Only Shaping Rates or Unshaped Logical Interfaces

Logical interfaces with only shaping rates (PIRs) or unshaped logical interfaces (units) are given a weight of 10. A logical interface with a small guaranteed rate (CIR) might get an overall weight less than 10. In order to allocate a higher share of the excess bandwidth to logical interfaces with a small guaranteed rate in comparison to the logical interfaces with only shaping rates configured, a minimum weight of 20 is given to the logical interfaces with guaranteed rates configured.

For example, consider a logical interface configuration with five units, as shown in [Table 130 on page 1084](#).



**Table 130: Allocating Weights with PIR and CIR on Logical Interfaces**

| Logical Interface (Unit) | Traffic Control Profile  | WRR Percentages | Weights         |
|--------------------------|--------------------------|-----------------|-----------------|
| Unit 1                   | PIR 100 Mbps             | 95, 0, 0, 5     | 10, 1, 1, 1     |
| Unit 2                   | CIR 20 Mbps              | 25, 25, 25, 25  | 64, 64, 64, 64  |
| Unit 3                   | PIR 40 Mbps, CIR 20 Mbps | 50, 30, 15, 5   | 128, 76, 38, 13 |
| Unit 4                   | Unshaped                 | 95, 0, 0, 5     | 10, 1, 1, 1     |
| Unit 5                   | CIR 1 Mbps               | 95, 0, 0, 5     | 10, 1, 1, 1     |

The weights for these units are calculated as follows:

- Select the excess bandwidth sharing proportional rate to be the maximum CIR among all the logical interfaces: 20 Mbps (unit 2).
- Unit 1 has a PIR and unit 4 is unshaped. The weight for these units is 10.
- The weight for unit 1 queue 0 is 9.5 ( $10 \times 95\%$ ), which translates to a hardware weight of 10.
- The weight for unit 1 queue 1 is 0 ( $0 \times 0\%$ ), but although the weight is zero, a weight of 1 is assigned to give minimal bandwidth to queues with zero WRR.
- Unit 5 has a very small CIR (1 Mbps), and a weight of 20 is assigned to units with a small CIR.
- The weight for unit 5 queue 0 is 19 ( $20 \times 95\%$ ), which translates to a hardware weight of 18.
- Unit 3 has a CIR of 20 Mbps, which is the same as the excess bandwidth sharing proportional rate, so it has a total weight of 255.
- The weight of unit 3 queue 0 is 127.5 ( $255 \times 50\%$ ), which translates to a hardware weight of 128.

### Sharing Bandwidth Among Logical Interfaces

As a simple example showing how bandwidth is shared among the logical interfaces, assume that all traffic is sent on queue 0. Assume also that there is a 40-Mbps load on all of the logical interfaces. Configuration details are shown in [Table 131 on page 1085](#).





**NOTE:** On the MX960 router, bandwidth sharing across high priority and strict-high priority schedulers configured on logical interfaces might not be as expected. This is a hardware limitation.

**Table 131: Sharing Bandwidth Among Logical Interfaces**

| Logical Interface (Unit) | Traffic Control Profile  | WRR Percentages | Weights         |
|--------------------------|--------------------------|-----------------|-----------------|
| Unit 1                   | PIR 100 Mbps             | 95, 0, 0, 5     | 10, 1, 1, 1     |
| Unit 2                   | CIR 20 Mbps              | 25, 25, 25, 25  | 64, 64, 64, 64  |
| Unit 3                   | PIR 40 Mbps, CIR 20 Mbps | 50, 30, 15, 5   | 128, 76, 38, 13 |
| Unit 4                   | Unshaped                 | 95, 0, 0, 5     | 10, 1, 1, 1     |

1. When the port is shaped at 40 Mbps, because units 2 and 3 have a guaranteed rate (CIR) configured, both units 2 and 3 get 20 Mbps of shared bandwidth.
2. When the port is shaped at 100 Mbps, because units 2 and 3 have a guaranteed rate (CIR) configured, each of them can transmit 20 Mbps. On units 1, 2, 3, and 4, the 60 Mbps of excess bandwidth is shaped according to the values shown in [Table 132 on page 1085](#).

**Table 132: First Example of Bandwidth Sharing**

| Logical Interface (Unit) | Calculation                                   | Bandwidth  |
|--------------------------|-----------------------------------------------|------------|
| Unit 1                   | $10 / (10+64+128+10) \times 60 \text{ Mbps}$  | 2.83 Mbps  |
| Unit 2                   | $64 / (10+64+128+10) \times 60 \text{ Mbps}$  | 18.11 Mbps |
| Unit 3                   | $128 / (10+64+128+10) \times 60 \text{ Mbps}$ | 36.22 Mbps |
| Unit 4                   | $10 / (10+64+128+10) \times 60 \text{ Mbps}$  | 2.83 Mbps  |



However, unit 3 only has 20 Mbps extra (PIR and CIR) configured. This means that the leftover bandwidth of 16.22 Mbps (36.22 Mbps – 20 Mbps) is shared among units 1, 2, and 4. This is shown in [Table 133 on page 1086](#).

**Table 133: Second Example of Bandwidth Sharing**

| Logical Interface (Unit) | Calculation                                     | Bandwidth  |
|--------------------------|-------------------------------------------------|------------|
| Unit 1                   | $10 / (10+64+128+10) \times 16.22 \text{ Mbps}$ | 1.93 Mbps  |
| Unit 2                   | $64 / (10+64+128+10) \times 16.22 \text{ Mbps}$ | 12.36 Mbps |
| Unit 4                   | $10 / (10+64+128+10) \times 16.22 \text{ Mbps}$ | 1.93 Mbps  |

Finally, [Table 134 on page 1086](#) shows the resulting allocation of bandwidth among the logical interfaces when the port is configured with a 100-Mbps shaping rate.

**Table 134: Final Example of Bandwidth Sharing**

| Logical Interface (Unit) | Calculation                       | Bandwidth  |
|--------------------------|-----------------------------------|------------|
| Unit 1                   | 2.83 Mbps + 1.93 Mbps             | 4.76 Mbps  |
| Unit 2                   | 20 Mbps + 18.11 Mbps + 12.36 Mbps | 50.47 Mbps |
| Unit 3                   | 20 Mbps + 20 Mbps                 | 40 Mbps    |
| Unit 4                   | 2.83 Mbps + 1.93 Mbps             | 4.76 Mbps  |

## Configuring Customer VLAN (Level 3) Shaping on Enhanced Queuing DPCs

Customer VLAN (level 3) shaping on an MX Series 10-Gigabit Ethernet Enhanced Queuing DPC differs from the customer VLAN (level 3) shaping on an MX Series 1-Gigabit Ethernet Enhanced Queuing DPC. To use the customer VLAN (level 3) shaping on an MX Series 10-Gigabit Ethernet Enhanced Queuing DPC, configure an interface set at the [edit interfaces interface-set] hierarchy level. You do not need to



configure the interface set while using customer VLAN (level 3) on an MX Series 1-Gigabit Ethernet Enhanced Queuing DPC.

To configure customer VLAN (level 3) shaping on an MX Series 10-Gigabit Ethernet Enhanced Queuing DPC:

1. Configure the interface set at the [edit interfaces] hierarchy level.

```
[edit interfaces]
user@host# set interface-set jnpr interface unit 100
user@host# set interface-set jnpr interface xe-1/0/0 unit 101
```

2. Configure the hierarchical scheduler and enable VLAN tagging.

```
[edit interfaces]
user@host# set xe-1/0/0 hierarchical-scheduler
user@host# set xe-1/0/0 vlan-tagging
```

3. Configure the logical interface properties.

```
[edit interfaces]
user@host# set xe-1/0/0 unit 100 vlan-id 100
user@host# set xe-1/0/0 unit 100 family inet address 10.1.0.1/24
user@host# set xe-1/0/0 unit 101 vlan-id 101
user@host# set xe-1/0/0 unit 101 family inet address 10.1.1.1/24
```

4. Configure the traffic control profiles at the [edit class-of-service] hierarchy level.

```
[edit class-of-service]
user@host# set traffic-control-profiles profile1 shaping-rate 10g burst-size 2k
user@host# set traffic-control-profiles profile1 guaranteed-rate 10g burst-size 2k
user@host# set traffic-control-profiles profile2 shaping-rate 50m burst-size 2k
user@host# set traffic-control-profiles profile2 guaranteed-rate 50m burst-size 2k
user@host# set traffic-control-profiles profile3 shaping-rate 80m burst-size 3k
user@host# set traffic-control-profiles profile3 guaranteed-rate 80m burst-size 3k
```

5. Configure the output traffic control profiles at the [edit class-of-service interfaces] hierarchy level.

```
[edit class-of-service interfaces]
user@host# set interface-set jnpr output-traffic-control-profiles profile1
```



```

user@host# set xe-1/0/0 unit 100 output-traffic-control-profiles profile2
user@host# set xe-1/0/0 unit 101 output-traffic-control-profiles profile3

```

To configure customer VLAN (level 3) shaping on an MX Series 1-Gigabit Ethernet Enhanced Queuing DPC:

1. Configure the interface set at the [edit interfaces] hierarchy level.

```

[edit interfaces]
user@host# set interface ge-1/0/0 unit 100
user@host# set interface ge-1/0/0 unit 101

```

2. Configure the hierarchical scheduler and enable the VLAN tagging.

```

[edit interfaces]
user@host# set ge-1/0/0 hierarchical-scheduler
user@host# set ge-1/0/0 vlan-tagging

```

3. Configure the logical interface properties.

```

[edit interfaces]
user@host# set ge-1/0/0 unit 100 vlan-id 100
user@host# set ge-1/0/0 unit 100 family inet address 10.1.0.1/24
user@host# set ge-1/0/0 unit 101 vlan-id 101
user@host# set ge-1/0/0 unit 101 family inet address 10.1.1.1/24

```

4. Configure the traffic control profiles at the [edit class-of-service] hierarchy level.

```

[edit class-of-service]
user@host# set traffic-control-profiles profile1 shaping-rate 10g burst-size 2k
user@host# set traffic-control-profiles profile1 guaranteed-rate 10g burst-size 2k
user@host# set traffic-control-profiles profile2 shaping-rate 50m burst-size 2k
user@host# set traffic-control-profiles profile2 guaranteed-rate 50m burst-size 2k
user@host# set traffic-control-profiles profile3 shaping-rate 80m burst-size 3k
user@host# set traffic-control-profiles profile3 guaranteed-rate 80m burst-size 3k

```



5. Configure the traffic control profiles at the `[edit class-of-service interfaces]` hierarchy level.

```
[edit class-of-service interfaces]
user@host# set ge-1/0/0 unit 100 output-traffic-control-profiles profile2
user@host# set ge-1/0/0 unit 101 output-traffic-control-profiles profile3
```

## RELATED DOCUMENTATION

[Enhanced Queuing DPC CoS Properties](#) | 1071

## Simple Filters Overview

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

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

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



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



## RELATED DOCUMENTATION

[Configuring a Simple Filter](#) | 1058

## Configuring Simple Filters on Enhanced Queuing DPCs

You can configure and apply a simple filter to perform multifield classification on the ingress interfaces of an MX Series router with Enhanced Queuing DPCs. These simple filters can be used to override default CoS classification parameters such as forwarding class or loss priority. Simple filters, in contrast to other firewall filters, only support a subset of the full firewall filter syntax.

To configure a simple filter, include the `simple-filter` statement at the `[edit firewall family inet]` hierarchy level:

```
[edit firewall family inet]
simple-filter filter-name {
  term term-name {
    from {
      ... match-conditions...
    }
    then {
      forwarding-class class-name;
      loss-priority priority;
    }
  }
}
```

The following example configures a simple filter to detect ingress packets from various source addresses (10.1.1.1/32, 10.10.10.10/32, and 10.4.0.0/8), destination addresses (10.6.6.6/32), protocols (tcp), and source ports (400-500, http). The filter then assigns various forwarding classes and loss priorities to the filtered traffic. Finally, the filter is applied to the input side of an Enhanced Queuing DPC interface (ge-2/3/3).

```
[edit]
firewall {
  family inet {
    simple-filter sf-for-eq-dpc {
      term 1 {
        from {
          source-address 10.1.1.1/32;
          protocol tcp;
        }
      }
    }
  }
}
```



```

    }
    then loss-priority low;
}
term 2 {
    from {
        source-address 10.4.0.0/8;
        source-port http;
    }
    then loss-priority high;
}
term 3 {
    from {
        destination-address 10.6.6.6/32;
        source-port 400-500;
    }
    then {
        loss-priority low;
        forwarding-class best-effort;
    }
}
term 4 {
    from {
        forwarding-class expedited-forwarding;
        source-address 10.10.10.10/32;
    }
    then loss-priority low;
}
term 5 {
    from {
        source-address 10.10.10.10/32;
    }
    then loss-priority low;
}
}
}

interfaces { # Apply the simple filter above to the input side of the interface.
ge-2/3/3 {
    unit 0 {
        family inet {
            simple-filter {
                input sf-for-eq-dpc;
            }
        }
    }
}
}

```



```

    }
  }
}

```

## RELATED DOCUMENTATION

*Simple Filter Overview*

*How Simple Filters Evaluate Packets*

*Guidelines for Configuring Simple Filters*

*Guidelines for Applying Simple Filters*

## Configuring a Simple Filter

This simple filter sets the loss priority to low for TCP traffic with source address 10.1.1.1, sets the loss priority to high for HTTP (web) traffic with source addresses in the 203.0.113.0/24 range, and sets the loss priority to low for all traffic with destination address 10.6.6.6. The simple filter is applied as an input filter (arriving packets are checking for destination address 10.6.6.6, not queued output packets) on interface ge-0/0/1.0.

```

[edit]
firewall {
  family inet {
    simple-filter filter1 {
      term 1 {
        from {
          source-address {
            10.1.1.1/32;
          }
          protocol {
            tcp;
          }
        }
        then loss-priority low;
      }
      term 2 {
        from {
          source-address {

```



## Simple Filters Overview | 1057



# Configuring Class of Service on MICs, MPCs, and MLCs

## IN THIS CHAPTER

- CoS Features and Limitations on MIC and MPC Interfaces | 1095
- Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview | 1098
- Verifying the Number of Dedicated Queues Configured on MIC and MPC Interfaces | 1101
- Scaling of Per-VLAN Queuing on Non-Queuing MPCs | 1102
- Increasing Available Bandwidth on Rich-Queuing MPCs by Bypassing the Queuing Chip | 1108
- Flexible Queuing Mode | 1110
- Multifield Classifier for Ingress Queuing on MX Series Routers with MPC | 1113
- Example: Configuring a Filter for Use as an Ingress Queuing Filter | 1114
- Ingress Queuing Filter with Policing Functionality | 1118
- Ingress Rate Limiting on MX Series Routers with MPCs | 1124
- Rate Shaping on MIC and MPC Interfaces | 1126
- Per-Priority Shaping on MIC and MPC Interfaces Overview | 1128
- Example: Configuring Per-Priority Shaping on MIC and MPC Interfaces | 1134
- Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates | 1142
- Example: Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates | 1143
- Traffic Burst Management on MIC and MPC Interfaces Overview | 1146
- Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 1150
- Configuring Ingress Hierarchical CoS on MIC and MPC Interfaces | 1152
- Configuring a CoS Scheduling Policy on Logical Tunnel Interfaces | 1154
- Per-Unit Scheduling and Hierarchical Scheduling for MPC Interfaces | 1157
- Managing Dedicated and Remaining Queues for Static CoS Configurations on MIC and MPC Interfaces | 1160
- Excess Bandwidth Distribution on MIC and MPC Interfaces Overview | 1162
- Bandwidth Management for Downstream Traffic in Edge Networks Overview | 1163



- [Scheduler Delay Buffering on MIC and MPC Interfaces | 1165](#)
- [Managing Excess Bandwidth Distribution on Static Interfaces on MICs and MPCs | 1166](#)
- [Drop Profiles on MIC and MPC Interfaces | 1169](#)
- [Intelligent Oversubscription on MIC and MPC Interfaces Overview | 1170](#)
- [Jitter Reduction in Hierarchical CoS Queues | 1172](#)
- [Example: Reducing Jitter in Hierarchical CoS Queues | 1175](#)
- [CoS on Ethernet Pseudowires in Universal Edge Networks Overview | 1183](#)
- [CoS Scheduling Policy on Logical Tunnel Interfaces Overview | 1184](#)
- [Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks | 1184](#)
- [CoS for L2TP LNS Inline Services Overview | 1186](#)
- [Configuring Static CoS for an L2TP LNS Inline Service | 1187](#)
- [CoS on Circuit Emulation ATM MICs Overview | 1190](#)
- [Configuring CoS on Circuit Emulation ATM MICs | 1192](#)
- [Understanding IEEE 802.1p Inheritance push and swap from a Transparent Tag | 1194](#)
- [Configuring IEEE 802.1p Inheritance push and swap from the Transparent Tag | 1195](#)
- [CoS on Application Services Modular Line Card Overview | 1198](#)

## CoS Features and Limitations on MIC and MPC Interfaces

MIC and MPC interfaces on MX Series 5G Universal Routing Platforms use the Trio chipset-based queuing model, which supports CoS characteristics that are optimized compared to CoS characteristics supported by the standard queuing model. However, some CoS features are not supported or are supported with limitations on MIC and MPC interfaces.

When configuring CoS features on MIC and MPC interfaces on MX Series routers, keep the following limitations in mind.

**Table 135: CoS Limitations on MIC and MPC Interfaces**

| CoS Feature | Limitation on MIC or MPC Interfaces                                      |
|-------------|--------------------------------------------------------------------------|
| Classifiers | Interfaces on MPCs support up to 32 classifiers of each type per module. |



Table 135: CoS Limitations on MIC and MPC Interfaces (*Continued*)

| CoS Feature                                       | Limitation on MIC or MPC Interfaces                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| BA classifier for MPLS packets                    | <p>When you configure a behavior aggregate (BA) classifier that does not include a specific rewrite rule for MPLS packets, we highly recommend that you enable the default MPLS EXP classifier. Doing so ensures that MPLS exp value is rewritten according to the BA classifier rules configured for forwarding or packet loss priority. For more information, see <a href="#">"Default MPLS EXP Classifier" on page 51</a>.</p> <p>To enable the default MPLS EXP classifier, include the default statement at the [edit class-of-service interfaces <i>interface-name</i> unit <i>logical-unit-number</i> rewrite-rules exp] hierarchy level.</p> |
| <i>Rewrite rules</i>                              | <p>For interfaces on MPCs or on MICs installed in MPCs, you can figure up to 32 rewrite rules:</p> <ul style="list-style-type: none"> <li>• DSCP rewrite rules</li> <li>• Internet rewrite rules</li> <li>• EXP rewrite rules</li> <li>• IEEE rewrite rules</li> </ul> <p>However, if you configure all 32 allowed rewrite rules, the class-of-service process can intermittently fail and generate syslog entries.</p>                                                                                                                                                                                                                              |
| Default rewrite rules for MPLS-enabled interfaces | <p>On interfaces other than MIC and MPC interfaces, the default EXP rewrite rule is automatically applied to MPLS-enabled interfaces, even if not configured. On MIC and MPC interfaces, you must explicitly configure EXP rewrite rules to MPLS-enabled interfaces.</p>                                                                                                                                                                                                                                                                                                                                                                             |
| Rewrite rules for service VLAN tag CoS bits       | <p>For MIC and MPC interfaces for VPLS or bridge domains, rewrite service VLAN tag CoS bits by configuring the rewrite rules on the <i>core-facing</i> interface.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |



Table 135: CoS Limitations on MIC and MPC Interfaces (*Continued*)

| CoS Feature                     | Limitation on MIC or MPC Interfaces                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Excess bandwidth sharing        | <p>Interfaces on MICs and MPCs do not support the <code>excess-bandwidth-share configuration statement</code>, which specifies how excess bandwidth at an interface set in a hierarchical scheduler environment is to be shared: proportionally or equally.</p> <p>Instead, you can include the <code>excess-rate</code> statement at one of the following hierarchy levels:</p> <ul style="list-style-type: none"> <li>• <code>[edit class-of-service schedulers scheduler-name]</code></li> <li>• <code>[edit class-of-service traffic-control-profiles traffic-control-profile-name]</code></li> </ul> |
| Layer 1 and Layer 2 overhead    | <p>MIC and MPC interfaces take all Layer 1 and Layer 2 overhead bytes into account for all levels of the hierarchy, including preamble, interpacket gaps, frame check sequence, and cyclical redundancy check.</p> <p>Queue statistics also take these overheads into account when displaying byte statistics.</p>                                                                                                                                                                                                                                                                                        |
| Pairing of load-balancing links | <p>When load balancing EQ MIC interfaces installed in Type 1 MPCs, you should configure odd- and even-numbered interfaces in the form <code>interface-fpc/odd   even/ports</code>. For example, if one link is <code>xe-1/0/0</code>, the other should be <code>xe-1/1/0</code>. If you do not configure odd and even load balancing, the system RED-drops packets when sending at line rate. This limitation does not apply to interfaces on EQ MICs installed in Type 2 MPCs.</p>                                                                                                                       |

## RELATED DOCUMENTATION

[Rate Shaping on MIC and MPC Interfaces | 1126](#)

[Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 1150](#)

[Scheduler Delay Buffering on MIC and MPC Interfaces | 1165](#)

[Drop Profiles on MIC and MPC Interfaces | 1169](#)



# Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview

IN THIS SECTION

- Queue Scaling for MPCs | 1098
- Managing Remaining Queues | 1100

Queuing Ethernet Modular Port Concentrators (MPCs) provide a set of dedicated queues for subscriber interfaces configured with hierarchical scheduling or per-unit scheduling.

The dedicated queues offered on these MPCs enable service providers to reduce costs through different scaling configurations. These queuing MPCs enable service providers to reduce the cost per subscriber by allowing many subscriber interfaces to be created with four or eight queues.

This topic describes the overall queue, scheduler node, and *logical interface* scaling for subscriber interfaces created on these MIC and MPC combinations.

## Queue Scaling for MPCs

Beginning with Junos OS Release 15.1, MPC2E-3D-NG-Q, MPC3E-3D-NG-Q, MPC5EQ-40G10G, and MPC5EQ-100G10G MPCs support up to five levels of hierarchical queuing. Beginning with Junos OS Release 16.1R1, MPC7 line cards also support five levels of hierarchical queuing. [Table 136 on page 1098](#) lists the number of dedicated queues and nodes supported per MPC.

Table 136: Dedicated Queues for MPCs

| MPC           | Dedicated Queues | Level 4 Nodes | Level 3 Nodes | Level 2 Nodes | Level 1 Nodes (Ports) |
|---------------|------------------|---------------|---------------|---------------|-----------------------|
| MPC2E-3D-NG-Q | 512,000          | 64,000        | 16,000        | 4000          | 384                   |
| MPC3E-3D-NG-Q |                  |               |               |               |                       |



Table 136: Dedicated Queues for MPCs (Continued)

| MPC                              | Dedicated Queues | Level 4 Nodes | Level 3 Nodes | Level 2 Nodes | Level 1 Nodes (Ports) |
|----------------------------------|------------------|---------------|---------------|---------------|-----------------------|
| MPC5EQ-40G10G                    | 1 million        | 128,000       | 32,000        | 4000          | 384                   |
| MPC5EQ-100G10G                   |                  |               |               |               |                       |
| MPC7                             | 512,000          | 64,000        | 16,000        | 8,000         | 252                   |
| MPC10E on MX10K series platforms | 256,000          | 32,000        | 8,000         | 4,000         | 128                   |



**CAUTION:** The maximum scaling targets provided in [Table 136 on page 1098](#) are based on system level design specifications. Actual realized subscriber or session scale is highly dependent upon the configuration and can be influenced by configuration variables including: the number of routes, the number of enabled services, the number of policy and firewall filters, policers, counters, statistics and access model type. Once you define a configuration, your Juniper account team can help characterize the expected system level scale or scale range for your live deployment.

MPCs vary in the number of Packet Forwarding Engines on board. MPC2E-3D-NG-Q and MPC3E-3D-NG-Q MPCs each have one Packet Forwarding Engine, allowing all 64,000 level 4 (subscriber) nodes to be allocated to a single MIC. MPC5EQ MPCs have two Packet Forwarding Engines, one for each possible MIC, each supporting 64,000 level 4 (subscriber) nodes. MPC7 MPCs also have two Packet Forwarding Engines, one for each possible MIC, each supporting 256,000 dedicated queues and 32,000 level 4 (subscriber) nodes.



**NOTE:** The nonqueueing MPCs MPC2E-3D-NG, MPC3E-3D-NG, MPC5E-40G10G, and MPC5E-100G10G provide up to eight queues per port in standard configuration. However, each of these MPCs can be configured to provide limited-scale hierarchical class of service (HCoS) and up to 32,000 queues.



## Managing Remaining Queues

In Junos OS releases earlier than Release 15.1R4, SNMP traps generate system log messages to notify you:

- When the number of available dedicated queues on the MPC drops below 10 percent. For example:

```
Mar 15 14:55:22.977 host cosd[1963]: COSD_OUT_OF_DEDICATED_QUEUES: Queue usage count for
interface xe-3/0/0 is at 90 percent
```

- When the maximum number of dedicated queues on the MPCs is reached. For example,

```
Mar 15 18:01:59.344 host cosd[3848]: COSD_OUT_OF_DEDICATED_QUEUES: Queue usage count for
interface xe-3/0/0 is at 100 percent.
```

When the maximum number of dedicated queues is allocated, the system does not provide subsequent subscriber interfaces with a dedicated set of queues. For per-unit scheduling configurations, there are no configurable queues remaining on the MPC.

For hierarchical scheduling configurations, remaining queues are available when the maximum number of dedicated queues is reached on the MPC. Traffic from these logical interfaces is considered unclassified and attached to a common set of queues that are shared by all subsequent logical interfaces. These common queues are the default port queues that are created for every port. You can configure a traffic-control profile and attach that to the interface to provide CoS parameters for the remaining queues. These subscriber interfaces remain with this traffic-control profile, even if dedicated queues become available.



**NOTE:** Starting in Junos OS Release 15.1R4, the COSD\_OUT\_OF\_DEDICATED\_QUEUES functionality is not available for QoS-enabled dynamic subscribers. Starting in Junos OS Release 17.4R1, CoS resource monitoring enables you to set a per-FPC queue threshold of up to 90 percent of resources bound to a scheduling hierarchy; subscriber logins are not allowed when the threshold is reached. However, this threshold applies to all queues, not dedicated queues alone. See [Resource Monitoring for Subscriber Management and Services Overview](#) for more information.

## Change History Table



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

| Release | Description                                                                                                                                                   |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 16.1R1  | Beginning with Junos OS Release 16.1R1, MPC7 line cards also support five levels of hierarchical queuing.                                                     |
| 15.1R1  | Beginning with Junos OS Release 15.1, MPC2E-3D-NG-Q, MPC3E-3D-NG-Q, MPC5EQ-40G10G, and MPC5EQ-100G10G MPCs support up to five levels of hierarchical queuing. |

### RELATED DOCUMENTATION

- [Hierarchical Class of Service User Guide](#)
- [Understanding Hierarchical Scheduling](#)
- [Managing Dedicated and Remaining Queues for Static CoS Configurations on MIC and MPC Interfaces](#)
- [Understanding Hierarchical Scheduling for MIC and MPC Interfaces](#)

## Verifying the Number of Dedicated Queues Configured on MIC and MPC Interfaces

### IN THIS SECTION

- [Purpose | 1101](#)
- [Action | 1102](#)

### Purpose

Display the number of dedicated queue resources that are configured for the logical interfaces on a port.



# Action

```

user@host#show class-of-service interface ge-1/1/0
Physical interface: ge-1/1/0, Index: 166
Queues supported: 4, Queues in use: 4
Total non-default queues created: 4
  Scheduler map: <default>, Index: 2
  Chassis scheduler map: <default-chassis>, Index: 4

Logical interface: ge-1/1/0.100, Index: 72, Dedicated Queues: no
  Shaping rate: 32000
  Object      Name              Type      Index
  Scheduler-map  <remaining>      0
  Classifier    ipprec-compatibility  ip      13

Logical interface: ge-1/1/0.101, Index: 73, Dedicated Queues: no
  Shaping rate: 32000
  Object      Name              Type      Index
  Scheduler-map  <remaining>      0
  Classifier    ipprec-compatibility  ip      13

Logical interface: ge-1/1/0.102, Index: 74, Dedicated Queues: yes
  Shaping rate: 32000
  Object      Name              Type      Index
  Traffic-control-profile  <control_tc_prof>  Output  45866

```

## RELATED DOCUMENTATION

[Managing Dedicated and Remaining Queues for Static CoS Configurations on MIC and MPC Interfaces](#)

---

*Managing Dedicated and Remaining Queues for Dynamic CoS Configurations on MIC and MPC Interfaces*

## Scaling of Per-VLAN Queuing on Non-Queuing MPCs

Per-VLAN (logical interface) queueing has been introduced on most MPCs supported on the MX platform. [Table 137 on page 1103](#) shows the details along with the supported JUNOS release.



**Table 137: MPC and MIC support for per-VLAN (logical interface) queuing**

| MPC                       | MICs Supported   | JUNOS Release |
|---------------------------|------------------|---------------|
| 16x10GE MPC               | N/A              | 13.2          |
| MPC3E                     | 2x10GE XFP       | 13.2          |
|                           | 10x10GE SFPP     | 13.2          |
|                           | 2x40G QSFPP      | 13.2          |
|                           | 1x100GE CXP      | 13.2          |
|                           | 1x100G CFP       | 13.2          |
| MPC4E-32x10GE SFPP        | N/A              | 13.3          |
| MPC4E-2x100GE+8x10GE SFPP | N/A              | 13.3          |
| MPC6E                     | 24x10GE SFPP     | 15.1          |
|                           | 24x10GE SFFP OTN | 15.1          |
|                           | 2x100GE CFP2 OTN | 15.1          |
|                           | 4x100GE CXP      | 15.1          |
| MPC5E-10G100G             | N/A              | 13.3R3        |
| MPC5E-10G40G              | N/A              | 13.3R3        |
| MPC2E-3D-NG/MPC3E-3D-NG   | 20x1GE SFP       | 15.1          |



Table 137: MPC and MIC support for per-VLAN (logical interface) queuing *(Continued)*

| MPC         | MICs Supported       | JUNOS Release |
|-------------|----------------------|---------------|
|             | 2xGE-XFP             | 15.1          |
|             | 40x1GE               | 15.1          |
|             | 4xGE-XFP             | 15.1          |
|             | 8OC3OC12-4OC48       | 15.1          |
|             | 4OC3OC12-1OC48       | 15.1          |
|             | 8CHOC3-4CHOC12       | 15.1          |
|             | 4CHOC3-1CHOC12       | 15.1          |
|             | 8DS3-E3              | 15.1          |
|             | 1xOC192-XFP          | 15.1          |
|             | 16xCHE1-T1-CE        | 15.1          |
|             | 8OC3-2OC12-ATM-CC-CE | 15.1          |
|             | 4COC3-1COC12-CE      | 15.1          |
|             | 20xGE-SFP-E          | 15.1          |
| MPC3E-3D-NG | 2x10GE XFP           | 15.1          |
|             | 10x10GE SFPP         | 15.1          |



**Table 137: MPC and MIC support for per-VLAN (logical interface) queuing (*Continued*)**

| MPC | MICs Supported | JUNOS Release |
|-----|----------------|---------------|
|     | 2x40G QSFP     | 15.1          |
|     | 1x100GE CXP    | 15.1          |

To enable logical interface scheduling, you include the `per-unit-scheduler` statement at the `[edit interfaces interface name]` hierarchy level. When per-unit schedulers are enabled, you can define dedicated schedulers for logical interfaces by including the `scheduler-map` statement at the `[edit class-of-service interfaces interface name unit logical unit number]` hierarchy level. Alternatively, you can include the `scheduler-map` statement at the `[edit class-of-service traffic-control-profiles traffic control profile name]` hierarchy level and then include the `output-traffic-control-profile` statement at the `[edit class-of-service interfaces interface name unit logical unit number]` hierarchy level.

[Table 138 on page 1105](#) shows the number of VLANs per port available in both 8-queue and 4-queue mode for 16x10GE, MPC3E, MPC4E and MPC6E.

**Table 138: Number of VLANs on 16x10G, MPC3E, MPC4E and MPC6E**

| MPC                    | MIC               | VLANs/Port – 8-Queue Mode | VLANs/Port – 4-Queue Mode |
|------------------------|-------------------|---------------------------|---------------------------|
| 16X10GE                | No                | 21                        | 44                        |
| MPC3E                  | 2x10GE with XFP   | 20                        | 42                        |
|                        | 10X10GE with SFP+ | 12 per group of 5 ports*  | 34 per group of 5 ports*  |
|                        | 2X40GE with QSFP+ | 20                        | 42                        |
|                        | 1X100GE with CXP  | 20                        | 42                        |
| 32x10GE MPC4E          | No                | 20 per group of 4 ports*  | 48 per group of 4 ports*  |
| 2x100GE + 8x10GE MPC4E | No                | 26                        | 54                        |



**Table 138: Number of VLANs on 16x10G, MPC3E, MPC4E and MPC6E (Continued)**

| MPC   | MIC                   | VLANs/Port – 8-Queue Mode | VLANs/Port – 4-Queue Mode |
|-------|-----------------------|---------------------------|---------------------------|
| MPC6E | 24X10GE               | 20 per group of 3 ports*  | 42 per group of 3 ports*  |
|       | 2X100GE with CFP2 OTN | 26                        | 54                        |
|       | 4X100GE MIC with CXP  | 21                        | 44                        |

\*The 10X10GE MIC for the MPC3E, the 32X10GE MPC4E, and the 24X10GE MICs for the MPC6E share VLANs across a port group. You can assign all of the available VLANs to one port within the port group or spread them across the ports in any combination.

Enabling and configuring per-unit schedulers on these interfaces adds additional output to the `show interfaces interface name [detail | extensive]` command. This additional output lists the maximum resources available and the number of configured resources for schedulers. Following is sample output showing the CoS scheduler resource information on a non-queuing line card:

```
root@R1# run show interfaces et-2/2/0 detail
```

```
Physical interface: et-2/2/0, Enabled, Physical link is Down
```

```
Interface index: 165, SNMP ifIndex: 550, Generation: 168
```

```
Link-level type: Ethernet, MTU: 1522, Speed: 100Gbps, BPDU Error: None, Loopback: Disabled,  
Source filtering: Disabled,
```

```
Flow control: Enabled
```

```
Device flags : Present Running Down
```

```
Interface flags: Hardware-Down SNMP-Traps Internal: 0x4000
```

```
Link flags : Scheduler
```

```
CoS queues : 8 supported, 8 maximum usable queues
```

```
Schedulers : 0
```

```
Hold-times : Up 0 ms, Down 0 ms
```

```
Current address: 80:71:1f:10:e6:b4, Hardware address: 80:71:1f:10:e6:b4
```

```
Last flapped : 2013-05-07 16:17:01 PDT (03:16:41 ago)
```

```
Statistics last cleared: Never
```

```
Traffic statistics:
```

```
Input bytes : 0 0 bps
```

```
Output bytes : 0 0 bps
```

```
Input packets: 0 0 pps
```



```

Output packets:          0          0 pps
IPv6 transit statistics:
  Input bytes :          0
  Output bytes :         0
  Input packets:         0
  Output packets:        0

```

Egress queues: 8 supported, 4 in use

**CoS scheduler resource information:**

**Maximum units supported per MIC/PIC: 20**

**Configured units per MIC/PIC: 1**

**Maximum units allowed per port: 20**

**Configured units on this port: 1**

| Queue counters: | Queued packets | Transmitted packets | Dropped packets |
|-----------------|----------------|---------------------|-----------------|
| 0 best-effort   | 0              | 0                   | 0               |
| 1 expedited-fo  | 0              | 0                   | 0               |
| 2 assured-forw  | 0              | 0                   | 0               |
| 3 network-cont  | 0              | 0                   | 0               |

| Queue number: | Mapped forwarding classes |
|---------------|---------------------------|
| 0             | best-effort               |

If you enable more VLANs than the previously mentioned MPC/MIC combinations support, VLANs up to the supported numbers receive dedicated queuing resources. The additional VLANs share port queues. Scheduling for port queues cannot be controlled. However, port queues are guaranteed 1 percent of the physical interface bandwidth to avoid queue starving and buffer holdup.

In the case of MPC2E-NG/3E-NG, MPC5E and MPC7E/8E/9E SKUs, the following command needs to be configured to enable “flexible queuing” on the MPC. Configuration of this knob results in a reboot of the MPC. The per-unit-scheduler, hierarchical scheduling and 2 level hierarchical scheduling are supported. There are 32K queues enabled and they can be used for either ingress queueing or egress queueing. The 32K queues are available when all 8 queues are used per IFL.

```

chassis {
  fpc 0 {
    flexible-queuing-mode;
  }
}

```

[Table 139 on page 1108](#) shows the number of VLANs per port available in both 8-queue and 4-queue mode for MPC3E-NG/MPC2E-NG, and MPC5E.



Table 139: Number of VLANs on MPC3E-NG/MPC2E-NG, MPC5E

| MPC               | MIC            | VLANs/Port – 8-Queue Mode | VLANs/Port – 4-Queue Mode |
|-------------------|----------------|---------------------------|---------------------------|
| MPC3E-NG/MPC2E-NG | Supported MICs | 32K                       | 32K                       |
| MPC5E             | Supported MICs | 32K                       | 32K                       |



**NOTE:** The number of logical interfaces with per-vlan queuing enabled should not exceed line card maximum. If the line card maximum is exceeded, then the queuing behavior is unpredictable. This could mean that some logical interfaces have queues assigned and some do not.

## RELATED DOCUMENTATION

*per-unit-scheduler*

[Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs | 391](#)

## Increasing Available Bandwidth on Rich-Queuing MPCs by Bypassing the Queuing Chip

Queuing MPCs contain a queuing chip that enables rich-queuing features such as hierarchical and per-vlan queuing. By default, all traffic passing through an interface on one of these MPCs also passes through the queuing chip, which decreases the available bandwidth of the interface. If you do not require hierarchical or per-vlan queuing on a particular interface of a queuing MPC, you can bypass the queuing chip to increase the available bandwidth.

To bypass the queuing chip on a queuing MPC, you must be running Junos OS Release 14.2 or later.

Starting with Junos OS 18.2R1, you can enable this option on vMX routers to save a vCPU when scheduling is not needed on an interface.

To bypass the queuing chip on an interface on a queuing MPC:

1. Ensure that neither `per-unit-scheduler` nor `hierarchical-scheduler` is configured on the interface.





**NOTE:** It is not possible to bypass the queuing chip on an interface if per-unit or hierarchical scheduling is configured on that interface.

2. Ensure that *flexi-queuing-mode* is enabled.
3. Enable `bypass-queuing-chip` on the interface.

For example:

```
[edit interfaces]
user@router# set interface- name bypass-queuing-chip
```

4. Commit your changes.

```
[edit interfaces]
user@router# show
interface-name {
    bypass-queueing-chip;
}
```

5. Verify your changes.

```
user@router> show interfaces interface-name
Physical interface: interface-name, Enabled, Physical link is Up
  Interface index: 147, SNMP ifIndex: 524
  Link-level type: Ethernet, MTU: 1514, MRU: 1522, LAN-PHY mode, Speed: 1000mbps,
  BPDU Error: None, MAC-REWRITE Error: None, Loopback: Disabled, Source filtering: Disabled,
  Flow control: Enabled, Auto-negotiation: Enabled, Remote fault: Online
  Pad to minimum frame size: Disabled
  Device flags   : Present Running
  Interface flags: SNMP-Traps Internal: 0x4000
  Link flags     : None
  CoS queues     : 8 supported, 4 maximum usable queues
  Schedulers     : 0, Queuing Chip Bypassed
  Current address: 00:21:59:0f:35:31, Hardware address: 00:21:59:0f:35:31
  Last flapped   : 2014-04-29 14:10:18 PDT (02:27:46 ago)
  Input rate     : 0 bps (0 pps)
  Output rate    : 0 bps (0 pps)
  Active alarms  : None
  Active defects : None
  Interface transmit statistics: Disabled
```



### Change History Table

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

| Release | Description                                                                                                                            |
|---------|----------------------------------------------------------------------------------------------------------------------------------------|
| 18.2R1  | Starting with Junos OS 18.2R1, you can enable this option on vMX routers to save a vCPU when scheduling is not needed on an interface. |
| 14.2    | To bypass the queuing chip on a queuing MPC, you must be running Junos OS Release 14.2 or later.                                       |

### RELATED DOCUMENTATION

| [bypass-queuing-chip](#)

## Flexible Queuing Mode

### IN THIS SECTION

- [Flexible Queuing Mode Overview | 1110](#)
- [Upgrading non-HQoS MPCs to Support Flexible Queuing | 1111](#)
- [Disabling Flexible Queuing for non-HQoS MPCs to Optimize Power Utilization | 1112](#)

You can configure the non-hierarchical quality-of-service (non-HQoS) MPCs to support port-based flexible queuing. By default, the non-HQoS MPCs do not support queuing. To enable queuing, you must upgrade these MPCs through an add-on license. After you upgrade these MPCs, they can support a flexible queuing capability of up to 32,000 queues per port and per card, including queues on both ingress and egress interfaces. Channelized MICs are supported on non-HQoS MPCs only when flexible queuing is configured.

### Flexible Queuing Mode Overview

The queuing component on non-HQoS MPCs is disabled by default to save power. When flexible queuing is enabled on a non-HQoS MPC, the MPC is restarted with the queuing component enabled. The MPC is powered on only if the PEM has sufficient power to bring up the MPC with the queuing component enabled. The MPC remains offline if the required power is not available.



You can enable flexible queuing on the non-HQoS MPCs by including the `flexible-queuing-mode` statement at the `[edit chassis fpc]` hierarchy level. When queuing is configured, the power consumed by the queuing components at the configured ambient temperature is considered when power is allocated for the MPC.



**NOTE:** The following MICs are supported on non-HQoS MPCs only when flexible queuing is enabled:

- MIC-3D-8CHOC3-4CHOC12
- MIC-3D-4CHOC3-2CHOC12
- MIC-4COC3-2COC12-G
- MIC-2COC3-1COC12-G

[Table 140 on page 1111](#) lists the MPCs that support flexible queuing and the supported Junos OS release for these MPCs.

**Table 140: MPCs and the Junos OS Release that Support Flexible Queuing**

| MPCs        | First Supported Junos OS Release |
|-------------|----------------------------------|
| MPC2E-3D-NG | 15.1R1                           |
| MPC3E-3D-NG | 15.1R1                           |
| MPC5E       | 14.1R1                           |
| MPC7E-MRATE | 15.1F4                           |
| MPC7E-10G   | 15.1F5                           |
| MPC8E       |                                  |
| MPC9E       |                                  |

### Upgrading non-HQoS MPCs to Support Flexible Queuing

You can enable flexible queuing on a non-HQoS MPC to support a maximum of up to 32,000 queues per port and per card, including queues on both ingress and egress interfaces.



This topic describes how to enable flexible queuing on a non-HQoS MPC.

To configure flexible queuing on non-HQoS MPCs:

1. Run the `set chassis fpc slot-number flexible-queuing-mode configuration mode` command.

For example, to configure flexible queuing on an MPC in slot 2:

```
[edit]
user@router# set chassis fpc 2 flexible-queuing-mode
```



**NOTE:** When flexible queuing is enabled, the MPC is restarted with the queuing component enabled. The MPC comes online only if the power entry module (PEM) has sufficient power to bring up the MPC with the queuing component enabled. The MPC remains offline if the required power is not available in the PEM.

2. Review your configuration and issue the `commit` command.

```
[edit]
user@router# commit
[edit]
'chassis fpc'
  WARNING: FPC configuration for flexible-queuing is changed. FPC would undergo reboot to
  enable flexible-queuing. FPC would come online only if power available is sufficient to
  enable queuing components.
commit complete
```

## Disabling Flexible Queuing for non-HQoS MPCs to Optimize Power Utilization

You can optimize power utilization by disabling flexible queuing on a non-HQoS MPC.

This topic describes how to disable flexible queuing on a non-HQoS MPC.

1. Run the `delete chassis fpc slot-number flexible-queuing-mode` command at the `[edit chassis]` hierarchy level.

For example, to disable flexible queuing on an MPC in slot 2:

```
[edit]
user@router# delete chassis fpc 2 flexible-queuing-mode
```



2. Review your configuration and issue the `commit` command.

```
[edit]
user@router# commit
commit complete

[edit]
user@router#
```

## RELATED DOCUMENTATION

| [flexible-queuing-mode](#)

## Multifield Classifier for Ingress Queuing on MX Series Routers with MPC

Beginning with Junos OS Release 16.1, the multifield classifier for ingress queuing is an implementation point for firewall filters configured with specific traffic shaping actions. These filters allow you to set the forwarding class and packet loss priority for packets, or drop the packets prior to ingress queue selection. The filters are applied as ingress queue filters. Class-of-service (CoS) commands can then be used to select ingress queue, set rate limiting and so forth.

Firewall filters configured at the protocol family level are able to distinguish specific types of traffic from other types by matching on multiple fields within the packet header. The number and types of matches available depend on which protocol family is used in the filter. Before the introduction of the ingress queuing filter, these firewall filters could only be applied to traffic after the ingress queue had been selected based solely on the behavior aggregate (BA). With the introduction of the ingress queuing filter, firewall filters can be used to set forwarding classification and packet loss priority based on multiple fields within the packet header prior to forwarding queue selection. CoS functions provide traffic classification options and the ability to assign that classified traffic to specific forwarding queues.



**NOTE:** Ingress queuing filters are only available when the traffic manager mode is set to ingress-and-egress at the `[edit chassis fpc fpc-id pic pic-id traffic-manager mode]` hierarchy level.

The `ingress-queuing-filter` configuration statement is used at the `[edit interfaces interface-name unit unit-number family family-name]` hierarchy level to designate a previously configured firewall filter to be used as an ingress queuing filter. The following list shows which protocol families are compatible with the `ingress-queuing-filter` statement:



- bridge
- ccc
- inet
- inet6
- mpls
- vpls

The named firewall filter is a normal firewall filter that must be configured with at least one of the following actions: accept, discard, forwarding-class, and loss-priority.

Change History Table

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

| Release | Description                                                                                                                                                                           |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 16.1    | Beginning with Junos OS Release 16.1, the multifield classifier for ingress queuing is an implementation point for firewall filters configured with specific traffic shaping actions. |

RELATED DOCUMENTATION

|                                                                                                  |
|--------------------------------------------------------------------------------------------------|
| <a href="#">Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic   44</a> |
| <i>ingress-queuing-filter</i>                                                                    |
| <i>Example: Configuring a Filter for Use as an Ingress Queuing Filter</i>                        |

Example: Configuring a Filter for Use as an Ingress Queuing Filter

IN THIS SECTION

- [Requirements | 1115](#)
- [Overview | 1115](#)
- [Configuration | 1115](#)



This example shows how to configure a firewall filter for use as an ingress queuing filter. The ingress queuing filter assists in traffic shaping operations by enabling you to set the forwarding class and packet loss priority, or drop the packet before ingress queue selection. The firewall filter must be configured within one of the following protocol families: bridge, cc, inet, inet6, mpls, or vpls and have one or more of the following actions: accept, discard, forwarding-class, and loss-priority.



**NOTE:** Although the ingress queuing filter can be used with EX9200 switches and T-Series routers as well as MX-Series routers, it is used only on those MX Series routers that have MPCs. An error is generated at commit if the ingress queuing filter is applied to an interface on any other type of port concentrator.

## Requirements

This example uses the following hardware and software components:

- An MX Series router with MPC

In order for ingress queuing filters to function, ingress-and-egress must be configured as the traffic-manager mode at the `[edit chassis fpc slot pic slot traffic-manager mode]` hierarchy level.

## Overview

In this example, you create a firewall filter named `iqfilter1` in the `inet` protocol family that sets the loss priority and forwarding class of packets coming from the `192.168.2.0/24` network. You then apply the `iqfilter1` filter to the `ge-0/0/0.0` logical interface as an ingress queuing filter.

To configure a firewall filter and apply it for use as an ingress queuing filter involves:

- Creating a firewall filter named `iqfilter1` in the `inet` protocol family with the following two actions: forwarding-class and loss-priority.
- Applying the firewall filter to the `ge-0/0/0.0` interface as an ingress queuing filter.

## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 1116](#)
- [Configuring the Firewall Filter and Applying It to an Interface as an Input Queuing Filter | 1116](#)
- [Results | 1117](#)



## CLI Quick Configuration

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

```
set firewall family inet filter iqfilter1 term t1 from address 192.168.2.0/24
set firewall family inet filter iqfilter1 term t1 then loss-priority low
set firewall family inet filter iqfilter1 term t1 then forwarding-class expedited-forwarding
set interfaces ge-0/0/0 unit 0 family inet ingress-queuing-filter iqfilter1
```

## Configuring the Firewall Filter and Applying It to an Interface as an Input Queuing Filter

### Step-by-Step Procedure

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

To configure the firewall filter, iqfilter1, and apply it to logical interface ge-0/0/0 unit 0:

1. Create a firewall filter named iqfilter1.

```
[edit firewall family inet]
user@router# set filter iqfilter1 term t1 from address 192.168.2.0/24
user@router# set filter iqfilter1 term t1 then loss-priority low
user@router# set filter iqfilter1 term t1 then forwarding-class expedited-forwarding
```

2. Apply the firewall filter to the logical interface.

```
[edit]
user@router# set interfaces ge-0/0/0 unit 0 family inet ingress-queuing-filter iqfilter1
```



## Results

From configuration mode, confirm your configuration by entering the `show firewall` and the `show interfaces ge-0/0/0.0` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```
user@router# show firewall
family inet {
  filter iqfilter1 {
    term t1 {
      from {
        address {
          192.168.0.0/24;
        }
      }
      then {
        loss-priority low;
        forwarding-class expedited-forwarding;
      }
    }
  }
}
user@router# show interfaces ge-0/0/0.0
family inet {
  ingress-queuing-filter iqfilter1;
}
```

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

```
user@router# commit
```

## RELATED DOCUMENTATION

*Multifield Classifier for Ingress Queuing on MX Series Routers with MPC*

*ingress-queuing-filter*



## Ingress Queuing Filter with Policing Functionality

### IN THIS SECTION

- [Understanding the Ingress Queuing Policing Filter | 1118](#)
- [Example: Configuring a Filter for Use as an Ingress Queuing Policing Filter | 1119](#)

Starting with Junos OS Release 18.1R1, on MPCs that support ingress queuing, you can implement policer actions, along with other filter actions, on traffic before the traffic is assigned to ingress queues. Ingress queuing policing filters allow you to rate limit traffic as well as count and set the forwarding class and packet loss priority for packets prior to ingress queue selection. Class-of-service (CoS) commands can then be used to select ingress queuing parameters.

### Understanding the Ingress Queuing Policing Filter

The ingress queuing policing filter (`iq-policing-filter`) function similarly and at the same point as the ingress policing filter (`ingress-queuing-filter`), which was introduced in Junos OS Release 16.1, but provides the added benefit of accepting almost all filter actions, including policing and counting actions. The ingress queuing policing filter is also more efficient, requiring fewer system resources.



**NOTE:** Ingress queuing filters are only available when the traffic manager mode is set to `ingress-and-egress` at the `[edit chassis fpc fpc-id pic pic-id traffic-manager mode]` hierarchy level.

The `iq-policing-filter` configuration statement is used at the `[edit interfaces interface-name unit unit-number family family-name]` hierarchy level to designate a previously configured firewall filter to be used as an ingress queuing policing filter. The following list shows which protocol families are compatible with the `iq-policing-filter` statement:

- bridge
- inet
- vpls

### SEE ALSO

| [\*iq-policing-filter\*](#)



## Example: Configuring a Filter for Use as an Ingress Queuing Policing Filter

### IN THIS SECTION

- [Requirements | 1119](#)
- [Overview | 1119](#)
- [Configuration | 1120](#)

This example shows how to configure a firewall filter for use as an ingress queuing policing filter. The ingress queuing filter assists in ingress traffic policing operations by allowing you to rate limit traffic prior to ingress queue selection. The firewall filter must be configured within one of the following protocol families: bridge, inet, or vpls.

The ingress queuing policing filter can only be used on MX Series routers with MPCs that support ingress queuing. An error is generated at commit if the ingress queuing filter is applied to an interface on any other type of port concentrator.

### Requirements

This example uses the following hardware and software components:

- An MX Series router with an MPC that supports ingress queuing

In order for ingress queuing filters to function, ingress-and-egress must be configured as the traffic-manager mode at the `[edit chassis fpc slot pic slot traffic-manager mode]` hierarchy level.

### Overview

In this example, you create a firewall filter named `vpls_iqp_filter` in the `vpls` protocol family that counts and polices voice and best effort traffic. You then apply the `vpls_iqp_filter` filter to the `xe-0/0/0.0` logical interface as an ingress queuing policing filter.

To configure a firewall filter and apply it for use as an ingress queuing filter involves:

- Creating a firewall filter named `vpls_iqp_filter` in the `vpls` protocol family with the following actions: `count`, `forwarding-class` and `policer`.
- Applying the firewall filter to the `xe-0/0/0.0` interface as an ingress queuing policing filter.



## Configuration

### IN THIS SECTION

- [CLI Quick Configuration | 1120](#)
- [Configuring the Firewall Filter and Applying It to an Interface as an Input Queuing Policing Filter | 1121](#)
- [Results | 1122](#)

### *CLI Quick Configuration*

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

```
set firewall family vpls filter vpls_iqp_filter interface-specific
set firewall family vpls filter vpls_iqp_filter term VoiceSum from learn-vlan-1p-priority 5
set firewall family vpls filter vpls_iqp_filter term VoiceSum then count VoiceSum
set firewall family vpls filter vpls_iqp_filter term VoiceSum then forwarding-class Voice
set firewall family vpls filter vpls_iqp_filter term VoiceSum then next term
set firewall family vpls filter vpls_iqp_filter term Voice from learn-vlan-1p-priority 5
set firewall family vpls filter vpls_iqp_filter term Voice then policer Voice-IN
set firewall family vpls filter vpls_iqp_filter term Voice then count Voice
set firewall family vpls filter vpls_iqp_filter term Voice then accept
set firewall family vpls filter vpls_iqp_filter term BestEffortSum then count BestEffortSum
set firewall family vpls filter vpls_iqp_filter term BestEffortSum then next term
set firewall family vpls filter vpls_iqp_filter term BestEffort then policer BestEffort-IN
set firewall family vpls filter vpls_iqp_filter term BestEffort then count BestEffort
set firewall family vpls filter vpls_iqp_filter term BestEffort then accept
set firewall family vpls filter vpls_iqp_filter policer pol-vpls if-exceeding bandwidth-limit
400m
set firewall family vpls filter vpls_iqp_filter policer pol-vpls if-exceeding burst-size-limit
40m
set firewall family vpls filter vpls_iqp_filter policer pol-vpls then discard
set firewall family vpls filter vpls_iqp_filter policer Voice-IN if-exceeding bandwidth-limit
100m
set firewall family vpls filter vpls_iqp_filter policer Voice-IN if-exceeding burst-size-limit
10m
```



```

set firewall family vpls filter vpls_iqp_filter policer Voice-IN then loss-priority high
set firewall family vpls filter vpls_iqp_filter policer BestEffort-IN if-exceeding bandwidth-
limit 350m
set firewall family vpls filter vpls_iqp_filter policer BestEffort-IN if-exceeding burst-size-
limit 30m
set firewall family vpls filter vpls_iqp_filter policer BestEffort-IN then loss-priority high
set interfaces xe-0/0/0 unit 0 family vpls iq-policing-filter vpls_iqp_filter

```

### *Configuring the Firewall Filter and Applying It to an Interface as an Input Queuing Policing Filter*

#### **Step-by-Step Procedure**

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

To configure the firewall filter, vpls\_iqp\_filter, and apply it to logical interface xe-0/0/0 unit 0:

1. Create a firewall filter named vpls\_iqp\_filter.

```

[edit firewall family vpls filter vpls_iqp_filter]
user@router# set interface-specific
user@router# set term VoiceSum from learn-vlan-1p-priority 5
user@router# set term VoiceSum then count VoiceSum
user@router# set term VoiceSum then forwarding-class Voice
user@router# set term VoiceSum then next term
user@router# set term Voice from learn-vlan-1p-priority 5
user@router# set term Voice then policer Voice-IN
user@router# set term Voice then count Voice
user@router# set term Voice then accept
user@router# set term BestEffortSum then count BestEffortSum
user@router# set term BestEffortSum then next term
user@router# set term BestEffort then policer BestEffort-IN
user@router# set term BestEffort then count BestEffort
user@router# set term BestEffort then accept
user@router# set policer pol-vpls if-exceeding bandwidth-limit 400m
user@router# set policer pol-vpls if-exceeding burst-size-limit 40m
user@router# set policer pol-vpls then discard
user@router# set policer Voice-IN if-exceeding bandwidth-limit 100m
user@router# set policer Voice-IN if-exceeding burst-size-limit 10m
user@router# set policer Voice-IN then loss-priority high
user@router# set policer BestEffort-IN if-exceeding bandwidth-limit 350m

```



```

user@router# set policer BestEffort-IN if-exceeding burst-size-limit 30m
user@router# set policer BestEffort-IN then loss-priority high

```

2. Apply the firewall filter to the logical interface.

```

[edit interfaces xe-0/0/0]
user@router# set unit 0 family vpls iq-policing-filter vpls_iqp_filter

```

### Results

From configuration mode, confirm your configuration by entering the `show firewall` and the `show interfaces xe-0/0/0.0` commands. If the output does not display the intended configuration, repeat the instructions in this example to correct the configuration.

```

user@router# show firewall family vpls filter vpls_iqp_filter

```

```

interface-specific;
term VoiceSum {
  from {
    learn-vlan-1p-priority 5;
  }
  then {
    count VoiceSum;
    forwarding-class Voice;
    next term;
  }
}
term Voice {
  from {
    learn-vlan-1p-priority 5;
  }
  then {
    policer Voice-IN;
    count Voice;
    accept;
  }
}
term BestEffortSum {
  then {

```



```

        count BestEffortSum;
        next term;
    }
}
term BestEffort {
    then {
        policer BestEffort-IN;
        count BestEffort;
        accept;
    }
}
policer pol_vpls {
    if-exceeding {
        bandwidth-limit 400m;
        burst-size-limit 40m;
    }
    then discard;
}
policer Voice-IN {
    if-exceeding {
        bandwidth-limit 100m;
        burst-size-limit 10m;
    }
    then loss-priority high;
}
policer BestEffort-IN {
    if-exceeding {
        bandwidth-limit 350m;
        burst-size-limit 30m;
    }
    then loss-priority high;
}

```

```

user@router# show interfaces xe-0/0/0 unit 0
family vpls {
    iq-policing-filter vpls_iqp_filter;
}

```



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

```
user@router# commit
```

SEE ALSO

| *iq-policing-filter*

Change History Table

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

| Release | Description                                                                                                                                                                                                   |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 18.1    | Starting with Junos OS Release 18.1R1, on MPCs that support ingress queuing, you can implement policer actions, along with other filter actions, on traffic before the traffic is assigned to ingress queues. |

RELATED DOCUMENTATION

| *iq-policing-filter*

Ingress Rate Limiting on MX Series Routers with MPCs

Beginning with Junos OS Release 16.2R1, on MPCs that support ingress queueing, you can perform rate limiting on incoming packets based on the forwarding class and packet loss priority (PLP) defined for each packet at ingress. You can define the ingress forwarding class either through behavior aggregate (BA) classification or through multifield (MF) ingress-queueing-filter classification.

A packet entering an interface that has ingress queueing and ingress rate-limiting enabled has the following path through the device:

```
Packet in -> BA classification -> MF classification -> Ingress rate-limiting -> Ingress queueing -> Loopback through the interface -> BA classification -> MF classification -> Firewall filters -> Routing -> Egress pipeline
```

As the packet enters the interface, BA and MF classification are used to determine the forwarding class and PLP for the packet. Ingress rate-limiting is applied to the packet based on the forwarding class and



PLP just determined. If no BA classifier is defined for the packet, the default BA classifier is used. Use of MF classification is optional and overrides any BA classification, including default BA classification.

Ingress rate limiting can be applied to physical and logical interfaces as well as interface sets.

To configure ingress rate-limiting:

1. Configure the traffic-manager statement with ingress-and-egress mode:

```
[edit chassis fpc slot-number pic pic-number]
user@router# set traffic-manager mode ingress-and-egress;
```

2. Configure a rate-limited scheduler. For example:

```
[edit class-of-service]
user@router# set schedulers sched_RL transmit-rate percent 70;
user@router# set schedulers sched_RL transmit-rate rate-limit;
```

3. Apply the scheduler to a scheduler map. For example:

```
[edit class-of-service]
user@router# set scheduler-maps SMap_RL forwarding-class expedited-forwarding scheduler
sched_RL;
```

4. Apply the scheduler map as an input-scheduler-map to a physical or logical interface or interface set. For example:

```
[edit class-of-service]
user@router# set interfaces interface-set my-set input-scheduler-map SMap_RL;
user@router# set interfaces ge-8/0/2 input-scheduler-map SMap_RL;
user@router# set interfaces ge-8/0/3 unit 0 input-scheduler-map SMap_RL;
```



**NOTE:** Alternatively, apply the scheduler map to a traffic control profile, then apply the traffic control profile as an input-traffic-control-profile to a physical or logical interface or interface set.



RELATED DOCUMENTATION

|                                                                                |
|--------------------------------------------------------------------------------|
| <i>Multifield Classifier for Ingress Queuing on MX Series Routers with MPC</i> |
| <a href="#">Configuring Ingress Hierarchical CoS   461</a>                     |

Rate Shaping on MIC and MPC Interfaces

IN THIS SECTION

- [Granularity of Rate Shaping on MIC and MPC Interfaces | 1126](#)
- [Accounting for Layer 1 and Layer 2 Overhead in Egress Rate-Shaping Statistics | 1127](#)

This topic covers the following information:

Granularity of Rate Shaping on MIC and MPC Interfaces

Interfaces hosted on MIC and MPC line cards have a certain granularity in the application of configured shaping rates. In other words, the observed hardware value might not exactly match the user-configured value. Nevertheless, the derived values are as close to the configured values as allowed.

[Table 141 on page 1126](#) lists the shaping granularity for each MPC port type. The derived shaping rate granularity ranges from 250 Kbps for coarse-grained queuing on the basic hardware up to 1.5 Kbps for fine-grained queuing on the enhanced queuing hardware.

Table 141: Shaping Rate Granularity for MPC Ports

| MIC or MPC Port in an MX Series Router |                  | Shaping Rate Granularity |                                              |
|----------------------------------------|------------------|--------------------------|----------------------------------------------|
| Line Card Type                         | Port Speed       | Port Level, Queue Level  | Logical Interface Level, Interface Set Level |
| Non-Queuing MPC                        | 1 Gbps / 10 Gbps | 250 Kbps                 | n/a                                          |
| Queuing MIC or MPC                     | 1 Gbps           | 2.4 Kbps                 | 9.6 Kbps                                     |



**Table 141: Shaping Rate Granularity for MPC Ports (Continued)**

| MIC or MPC Port in an MX Series Router |            | Shaping Rate Granularity |                                              |
|----------------------------------------|------------|--------------------------|----------------------------------------------|
| Line Card Type                         | Port Speed | Port Level, Queue Level  | Logical Interface Level, Interface Set Level |
|                                        | 10 Gbps    | 9.6 Kbps                 | 38.4 Kbps                                    |
| Enhanced Queuing MPCs                  | 1 Gbps     | 1.5 Kbps                 | 6 Kbps                                       |
|                                        | 10 Gbps    | 6 Kbps                   | 24 Kbps                                      |



**NOTE:** The shaping rate granularity for MX Series routers with the MPC3E and MPC4E is approximately 293-300 Kbps. For routers with other MPCs (Trio-based FPCs), the shaping rate granularity is 250 Kbps. The predefined shaping rates for these MPCs are the next multiple of these shaping rate granularity values. The expected deviation from the predefined shaping rates is 5 to 10 percent.

## Accounting for Layer 1 and Layer 2 Overhead in Egress Rate-Shaping Statistics

In calculating egress rate-shaping statistics for shaped-session packets on the egress side of MIC and MPC interfaces, the system adds 20 bytes per packet by default.

To configure an explicit overhead value to use for calculating egress rate-shaping statistics, include the `egress-shaping-overhead` statement at the `[edit chassis fpc slot-number pic pic-number traffic-manager]` hierarchy level. You can specify an offset value from -63 bytes through 192 bytes per egress packet.

## RELATED DOCUMENTATION

[CoS Features and Limitations on MIC and MPC Interfaces | 1095](#)

[Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 1150](#)

[Scheduler Delay Buffering on MIC and MPC Interfaces | 1165](#)

[Drop Profiles on MIC and MPC Interfaces | 1169](#)



## Per-Priority Shaping on MIC and MPC Interfaces Overview

Per-priority shaping enables you to configure a separate shaping rate for each of the five priority levels supported by MIC and MPC interfaces. The main use of per-priority shaping rates is to ensure that higher priority services such as voice and video do not starve lower priority services such as data.

There are five scheduler priorities:

- Guaranteed high (GH)
- Guaranteed medium (GM)
- Guaranteed low (GL)
- Excess high (EH)
- Excess low (EL)

The five scheduler priorities support a shaping rate for each priority:

- Shaping rate priority high (GH)
- Shaping rate priority medium (GM)
- Shaping rate priority low (GL)
- Shaping rate excess high (EH)
- Shaping rate excess low (EL)

On MPC7E (MPC7E-MRATE and MPC7E-10G), MPC8E (MX2K-MPC8E), MPC9E(MX2K-MPC9E), and PTX series, when you enable the enhanced priority mode feature, additional scheduler priorities and shaping rates are supported. For more information on the enhanced priority mode feature, see *enhanced-priority-mode*.

The additional scheduler priorities supported when you enable the enhanced priority mode feature:

- Shaping rate priority strict high (GHL)
- Shaping rate priority medium low (GML)
- Shaping rate excess medium high (EMH)
- Shaping rate excess medium low (EML)

When you enable the enhanced priority mode feature, the queue priorities are mapped to the priorities of the MPCs :



**Table 142: Shaping Priority and Default Excess Shaping Priority**

| Configured Priority | Priority Supported on the MPC | Default Excess Priority |
|---------------------|-------------------------------|-------------------------|
| Strict-High         | GH                            | EH                      |
| High                | GHL                           | EH                      |
| Medium-High         | GM                            | EL                      |
| Medium-Low          | GML                           | EL                      |
| Low                 | GL                            | EM                      |

If each service is represented by a forwarding class queued at a separate priority, then assigning a per-priority shaping rate to higher priority services accomplishes the goal of preventing the starvation of lower priority services.

To configure per-priority shaping rates, include the `shaping-rate-excess-high rate <burst-size burst>`, `shaping-rate-excess-low rate <burst-size burst>`, `shaping-rate-priority-high rate <burst-size burst>`, `shaping-rate-priority-low rate <burst-size burst>`, or `shaping-rate-priority-medium rate <burst-size burst>` at the `[edit class-of-service traffic-control-profiles tcp-name]` hierarchy level and apply the traffic control profile at the `[edit interfaces]` hierarchy level. You can specify the rate in absolute values, or by using k (kilo-), m (mega-) or g (giga-) units.

You can include one or more of the per-priority shaping statements in a traffic control profile:

```
[edit class-of-service]
traffic-control-profiles {
  tcp-ge-port {
    shaping-rate-excess-high rate <burst-size bytes>;
    shaping-rate-excess-low rate <burst-size bytes>;
    shaping-rate-priority-high rate <burst-size bytes>;
    shaping-rate-priority-low rate <burst-size bytes>;
    shaping-rate-priority-medium rate <burst-size bytes>;
  }
}
```





**NOTE:** To use per-priority shaping on a physical interface on the MX104 router, you must enable hierarchical scheduling on the interface with the **set hierarchical-scheduler** statement at the [edit interface *interface-name*] hierarchy level.



**BEST PRACTICE:** When planning your implementation, consider the following behavior. You can configure independent burst-size values for each rate, but the system uses the maximum burst-size value configured in each rate family. For example, the system uses the highest configured value for the guaranteed rates (GH and GM) or the highest value of the excess rates (EH and EM).

There are several important points about per-priority shaping rates:

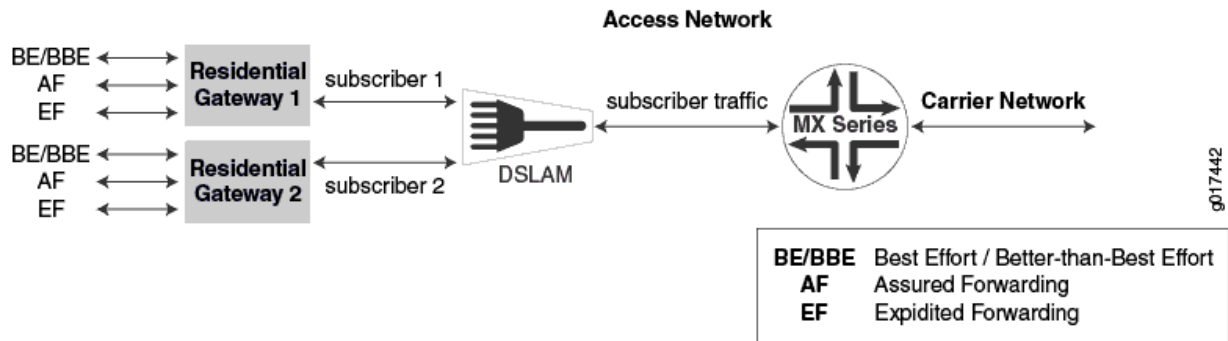
- Per-priority shaping rates are only supported on MIC and MPC interfaces (with the exception of the 10-Gigabit Ethernet MPC with SFP+).
- Per-priority shaping is only available for level 1 and level 2 scheduler nodes. (For more information on hierarchical schedulers, see ["Configuring Hierarchical Schedulers for CoS" on page 448.](#))
- Per-priority shaping rates are supported when level 1 or level 2 scheduler nodes have static or dynamic interfaces above them.
- Per-priority shaping rates are supported on aggregated Ethernet (AE) interfaces.
- Per-priority shaping rates are only supported in traffic control profiles.

Per-priority shaping rates can be helpful when the MX Series 5G Universal Routing Platform is in a position between subscriber traffic on an access network and the carrier network, playing the role of a broadband services router. In that case, the MX Series router provides quality-of-service parameters on the subscriber access network so that each subscriber receives a minimum bandwidth (determined by the guaranteed rate) and a maximum bandwidth (determined by the shaping rate). This allows the devices closer to the carrier network to operate more efficiently and more simply and reduces operational network expenses because it allows more centralized network management.

One architecture for using per-priority shaping on the MX Series router is shown in [Figure 64 on page 1131](#). In the figure, subscribers use residential gateways with various traffic classes to support voice, video, and data services. The MX Series router sends this traffic from the carrier network to the digital subscriber line access multiplexer (DSLAM) and from the DSLAM on to the residential gateway devices.



Figure 64: Architecture for MIC and MPC Interface Per-Priority Shaping



One way that the MX Series router can provide service classes for this physical network topology is shown in [Figure 65 on page 1132](#). In the figure, services such as voice and video are placed in separate forwarding classes and the services at different priority levels. For example:

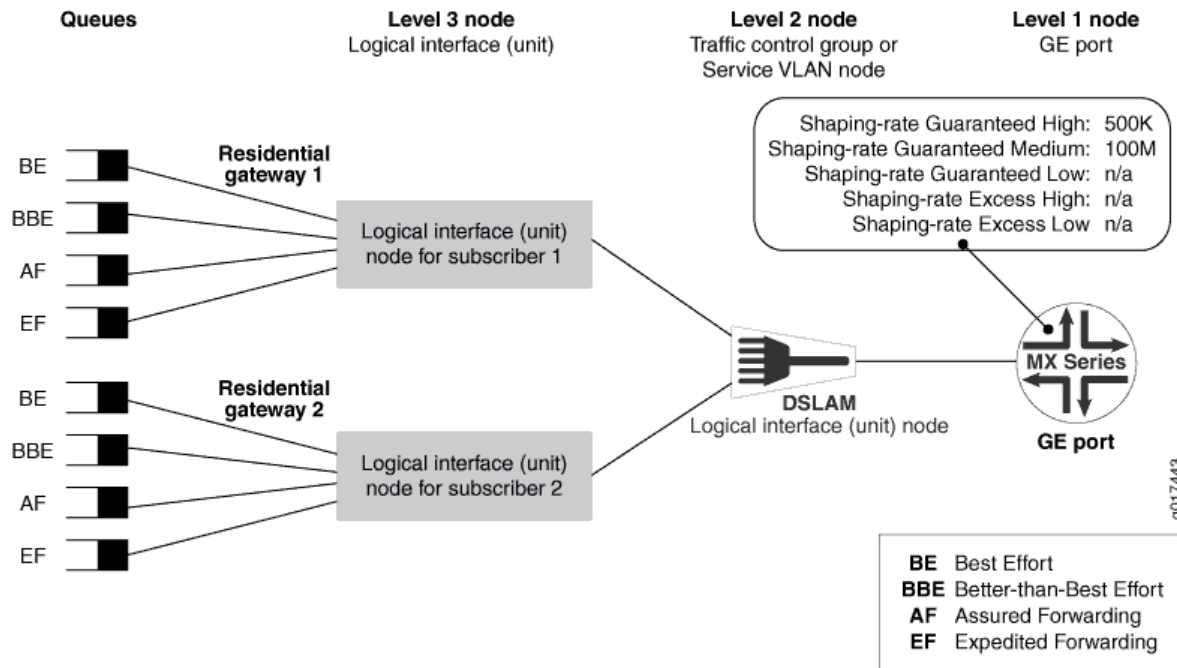
- All expedited-forwarding queues are voice services at a priority level of guaranteed high.
- All assured-forwarding queues are video services at a priority level of guaranteed medium.
- All better-than-best-effort queues are services at a priority level of excess high.
- All best-effort queues are services at a priority level of excess low.



**NOTE:** This list covers only one possible configuration. Others are possible and reasonable, depending on the service provider's goals. For example, best-effort and better-than-best-effort traffic can have the same priority level, with the better-than-best-effort forwarding class having a higher scheduler weight than the best-effort forwarding class. For more information on forwarding classes, see ["Configuring a Custom Forwarding Class for Each Queue" on page 281](#).



Figure 65: Scheduling Hierarchy for Per-Priority Shaping



Aggregated voice traffic in this topology is shaped by applying a high-priority shaper to the port. Aggregated video traffic is shaped in the same way by applying a medium-priority shaper to the port. As long as the sum of the high- and medium-priority shapers is less than the port speed, some bandwidth is reserved for best-effort and better-than-best-effort traffic. So assured-forwarding and expedited-forwarding voice and video cannot starve best-effort and better-than-best-effort data services. One possible set of values for high-priority (guaranteed high) and medium-priority (guaranteed medium) traffic is shown in [Figure 65 on page 1132](#).



**BEST PRACTICE:** We recommend that you do not shape delay-sensitive traffic such as voice traffic because it adds delay (latency). Service providers often use connection admission control (CAC) techniques to limit aggregated voice traffic. However, establishing a shaping rate for other traffic guards against CAC failures and can be useful in pacing extreme traffic bursts.

Per-priority shaping statements:

```
[edit class-of-service]
traffic-control-profile {
  tcp-for-ge-port {
    shaping-rate-priority-high 500k;
    shaping-rate-priority-medium 100m;
```



```

    }
}

```

Apply (attach) the traffic control profile to the physical interface (port) at the [edit class-of-services interfaces] hierarchy level:

```

[edit class-of-service]
interfaces {
    ge-1/0/0 {
        output-traffic-control-profile tcp-for-ge-port;
    }
}

```

Traffic control profiles with per-priority shaping rates can only be attached to interfaces that support per-priority shaping.

You can apply per-priority shaping to levels other than the level 1 physical interface (port) of the scheduler hierarchy. Per-priority shaping can also be applied at level 2, the interface set level, which would typically represent the digital subscriber link access multiplexer (DSLAM). At this level you could use per-priority shaping to limit to total amount of video traffic reaching a DSLAM, for example.

You apply (attach) the traffic control profile to an interface set at the [edit class-of-services interfaces] hierarchy level:

```

[edit class-of-service]
interfaces {
    interface-set svlan-1 {
        output-traffic-control-profile tcp-for-ge-port;
    }
}

```



**NOTE:** Although you can configure both input and output traffic control profiles, only output traffic control profiles are supported for per-priority shaping.

You can configure per-priority shaping for the traffic remaining with the output-traffic-control-profile-remaining statement on a physical port (a level 2 node) but not for an interface set (a level 3 node).

## Change History Table



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

| Release | Description                                                                                                                                                                                                       |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 16.1R1  | On MPC7E (MPC7E-MRATE and MPC7E-10G), MPC8E (MX2K-MPC8E), MPC9E(MX2K-MPC9E), and PTX series, when you enable the enhanced priority mode feature, additional scheduler priorities and shaping rates are supported. |

RELATED DOCUMENTATION

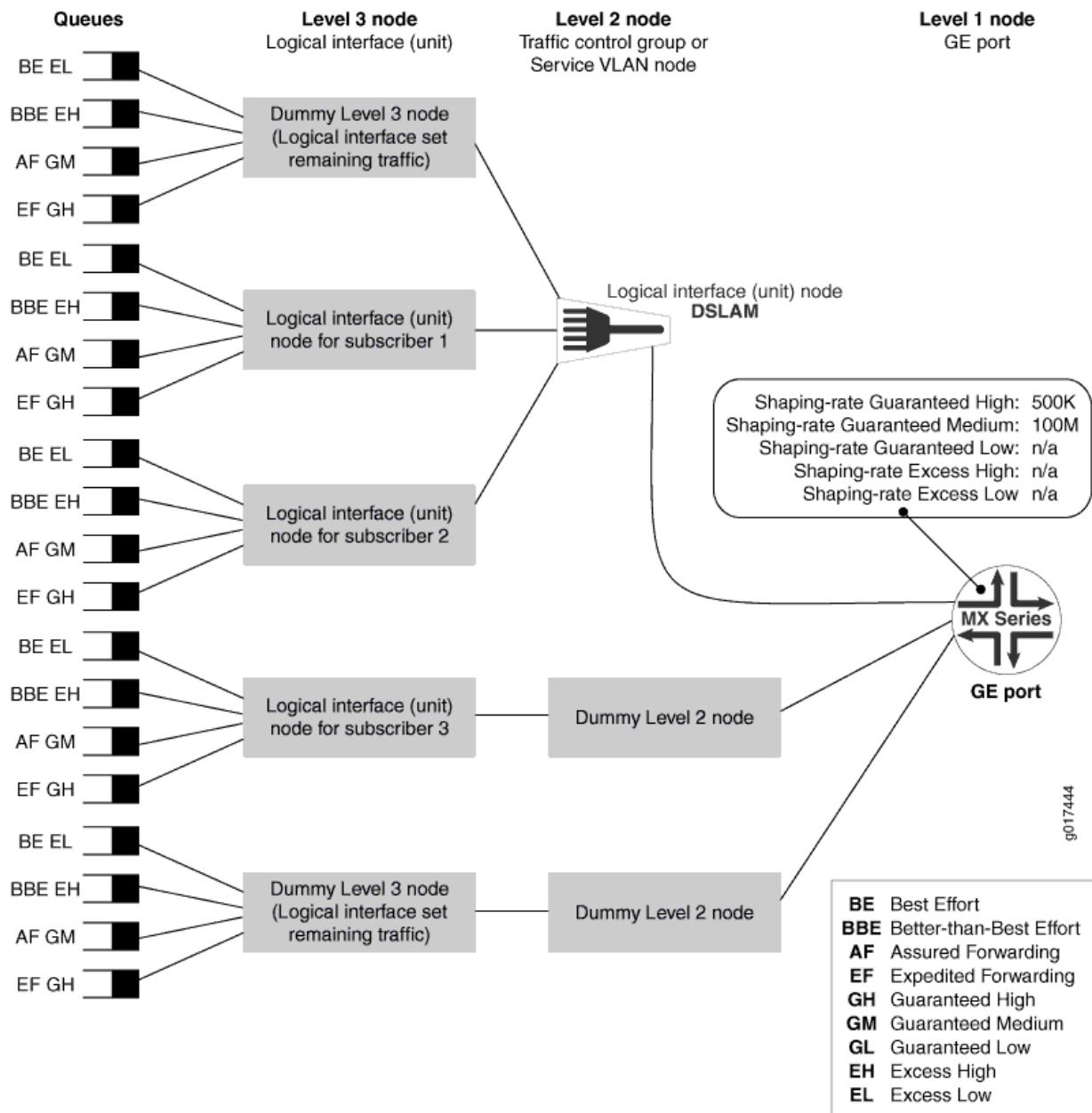
|                                                                         |
|-------------------------------------------------------------------------|
| <i>Excess Bandwidth Distribution on MIC and MPC Interfaces Overview</i> |
| <i>enhanced-priority-mode</i>                                           |

Example: Configuring Per-Priority Shaping on MIC and MPC Interfaces

In practice, per-priority shaping is used with other traffic control profiles to control traffic as a whole. Consider the traffic control profile applied to the physical interface (port), as shown in [Figure 66 on page 1135](#).



Figure 66: Example of MIC and MPC Interface Scheduling Hierarchy



This example is more complex than those used before. In addition to a pair of subscribers in an interface set (DSLAM), the figure now adds the following:

- A dummy level 3 scheduler node (interface-set-remaining-traffic) that provides scheduling for interface set members that do not have explicit class-of-service parameters configured.
- A subscriber (Subscriber 3) that is not a member of an interface set. A dummy level 2 node connects Subscriber 3's level 3 node to level 1, making it appear to be at level 2.



- A dummy level 3 scheduler node (port-remaining-traffic) in order to provide queues for traffic that does not have explicit class-of-service parameters configured.
- A dummy level 2 scheduler node to connect level 1 and level 3 scheduler nodes. This dummy level 2 scheduler node is internal only.

This example uses a gigabit Ethernet interface with five logical interface units, each one representing one of the level 3 nodes in [Figure 66 on page 1135](#).

From the top of the figure to the bottom, the level 3 nodes are:

- Unit 3 is scheduled as a “dummy” level 3 node because unit 3 is a member of an interface set (ifset-1) but there is no explicit CoS configuration.
- Unit 1 is scheduled as a logical interface node for subscriber 1 because unit 1 is a member of an interface set (ifset-1) and has an explicit CoS configuration under the [edit class-of-service interfaces] hierarchy.
- Unit 2 is scheduled as a logical interface node for subscriber 2 because unit 2 is a member of an interface set (ifset-1) and has an explicit CoS configuration under the [edit class-of-service interfaces] hierarchy.
- Unit 4 is scheduled as a logical interface node for subscriber 3 because unit 4 is not a member of an interface set but has an explicit CoS configuration under the [edit class-of-service interfaces] hierarchy level.
- Unit 5 is scheduled by another “dummy” level 3 node, this one for remaining traffic at the port level, because unit 5 is not a member of an interface set and has no explicit CoS configuration.

In this example, per-priority shaping is applied at the physical port level. The example uses three priorities, but other parameters are possible. The example does not use shaping rates, transmit rates, excess priorities, or other options for reasons of simplicity. The example uses five forwarding classes and leaves out a network control forwarding class that would typically be included in real configurations.

The example configuration is presented in several parts:

- Interfaces configuration
- Class-of-service forwarding classes and traffic control profiles configuration
- Class-of-service interfaces configuration
- Class-of-service schedulers and scheduler map configuration



Interfaces configuration:

```
[edit]
interfaces {
  # A three member interface-set.
  interface-set ifset-1 {
    interface ge-1/1/0 {
      unit 1;
      unit 2;
      unit 3;
    }
  }
  # A ge port configured for "hierarchical-scheduling" and
  # vlans. 5 vlans are configured for the 5 level-3 scheduler
  # nodes
  #
  ge-1/1/0 {
    hierarchical-scheduler;
    vlan-tagging;
    unit 1 {
      vlan-id 1;
    }
    unit 2 {
      vlan-id 2;
    }
    unit 3 {
      vlan-id 3;
    }
    unit 4 {
      vlan-id 4;
    }
    unit 5 {
      vlan-id 5;
    }
  }
}
```

Class-of-service forwarding classes and traffic control profiles configuration:

```
[edit class-of-service]
forwarding-classes {
```



```

queue 0 BE priority low;
queue 1 BBE priority low;
queue 2 AF priority low;
queue 3 EF priority high;
}
traffic-control-profiles {
    tcp-if-portd {
        shaping-rate-priority-high 500k;
        shaping-rate-priority-medium 100m;
    }
    tcp-if-port-rem {
        scheduler-map smap-1;
    }
    tcp-ifset-rem {
        scheduler-map smap-1;
    }
    tcp-if-unit {
        scheduler-map smap-1;
        shaping-rate 10m;
    }
}

```

Class-of-service interfaces configuration:

```

[edit class-of-service]
interfaces {
    interface-set ifset-1 {
        output-traffic-control-profile-remaining tcp-ifset-rem;
    }
    ge-1/1/0 {
        output-traffic-control-profile tcp-if-port;
        output-traffic-control-profile-remaining tcp-if-port-rem;
        unit 1 {
            output-traffic-control-profile tcp-if-unit;
        }
        unit 2 {
            output-traffic-control-profile tcp-if-unit;
        }

        # Unit 3 present in the interface config and interface-set
        # config, but is absent in this CoS config so that we can
        # show traffic that uses the interface-set
    }
}

```



```

    # remaining-traffic path.

    unit 4 {
        output-traffic-control-profile tcp-if-unit;
    }

    # Unit 5 is present in the interface config, but is absent
    # in this CoS config so that we can show traffic that
    # uses the if-port remaining-traffic path.
}
}

```

Class-of-service schedulers and scheduler map configuration:

```

[edit class-of-service]
scheduler-maps {
    smap-1 {
        forwarding-class BE scheduler sched-be;
        forwarding-class BBE scheduler sched-bbe;
        forwarding-class AF scheduler sched-af;
        forwarding-class EF scheduler sched-ef;
    }
    schedulers {
        sched-be {
            priority low;
        }
        sched-bbe {
            priority low;
        }
        sched-af {
            priority medium-high;
        }
        sched-ef {
            priority high;
        }
    }
}

```

You can configure both a shaping rate and a per-priority shaping rate. In this case, the legacy `shaping-rate` statement specifies the maximum rate for all traffic scheduled through the scheduler. Therefore, the per-priority shaping rates must be less than or equal to the overall shaping rate. So if there is a `shaping-rate 400m` statement configured in a traffic control profile, you cannot configure a higher value for a per-priority shaping rate (such as `shaping-rate-priority-high 500m`). However, the sum of the per-priority shaping



rates can exceed the overall shaping rate: for `shaping-rate 400m` you can configure both `shaping-rate-priority-high 300m` and `shaping-rate-priority-low 200m` statements.

Generally, you cannot configure a shaping rate that is smaller than the guaranteed rate (which is why it is guaranteed). However, no such restriction is placed on per-priority shaping rates unless all shaping rates are for priority high or low or medium traffic.

This configuration is allowed (per-priority rates smaller than guaranteed rate):

```
[edit class-of-service]
traffic-control-profile {
  tcp-for-ge-port {
    guaranteed-rate 500m;
    shaping-rate-priority-high 400m;
    shaping-rate-priority-medium 300m;
    shaping-rate-excess-high 100m;
  }
}
```

However, this configuration generates an error (no excess per-priority rate, so the node can never achieve its guaranteed rate):

```
[edit class-of-service]
traffic-control-profile {
  tcp-for-ge-port {
    guaranteed-rate 301m;
    shaping-rate-priority-high 100m;
    shaping-rate-priority-medium 100m;
    shaping-rate-priority-low 100m;
  }
}
```

You verify configuration of per-priority shaping with the `show class-of-service traffic-control-profile` command. This example shows shaping rates established for the high and medium priorities for a traffic control profile named `tcp-ge-port`.

```
user@host# show class-of-service traffic-control-profile
Traffic control profile: tcp-ae, Index: 22093
  Shaping rate: 3000000000
  Scheduler map: <default>
```



```

Traffic control profile: tcp-ge-port, Index: 22093
  Shaping rate priority high: 1000000000
  Shaping rate priority medium: 9000000000
  Scheduler map: <default>

```

There are no restrictions on or interactions between per-priority shaping rates and the excess rate. An excess rate (a weight) is specified as a percentage or proportion of excess bandwidth.

[Table 143 on page 1141](#) shows where traffic control profiles containing per-priority shaping rates can be attached for both per-unit schedulers and hierarchical schedulers.

**Table 143: Applying Traffic Control Profiles**

| Type of Traffic Control Profile                                                                                  | Per-unit Allowed? | Hierarchical Allowed? |
|------------------------------------------------------------------------------------------------------------------|-------------------|-----------------------|
| Port level output-traffic-control-profile with per-priority shaping                                              | Yes               | Yes                   |
| Port level output-traffic-control-profile-remaining with per-priority shaping                                    | No                | Yes                   |
| Port level output-traffic-control-profile and output-traffic-control-profile-remaining with per-priority shaping | No                | Yes                   |
| Port level input-traffic-control-profile with per-priority shaping                                               | No                | No                    |
| Port level input-traffic-control-profile-remaining with per-priority shaping                                     | No                | No                    |
| Interface set output-traffic-control-profile with per-priority shaping                                           | No                | Yes                   |
| Interface set output-traffic-control-profile-remaining with per-priority shaping                                 | No                | No                    |
| Interface set input-traffic-control-profile with per-priority shaping                                            | No                | No                    |
| Interface set input-traffic-control-profile-remaining with per-priority shaping                                  | No                | No                    |
| Logical interface level output-traffic-control-profile with per-priority shaping                                 | No                | No                    |



Table 143: Applying Traffic Control Profiles (*Continued*)

| Type of Traffic Control Profile                                                 | Per-unit Allowed? | Hierarchical Allowed? |
|---------------------------------------------------------------------------------|-------------------|-----------------------|
| Logical interface level input-traffic-control-profile with per-priority shaping | No                | No                    |

## RELATED DOCUMENTATION

[Per-Priority Shaping on MIC and MPC Interfaces Overview](#) | 1128

## Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates

The overhead accounting feature enables you to account for downstream traffic that has different encapsulations or downstream traffic from cell-based equipment, such as ATM switches.

You can configure the overhead accounting feature to shape downstream traffic based on frames or cell shaping mode.

You can also account for the different byte sizes per encapsulation by configuring a byte adjustment value for the shaping mode.

To configure the shaping mode and byte adjustment value for static CoS configurations:

### 1. Specify the shaping mode.

Frame shaping mode is enabled by default.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set overhead-accounting (frame-mode | cell-mode)
```

### 2. (Optional) Specify a byte adjustment value.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set overhead-accounting bytes byte-value
```





**BEST PRACTICE:** We recommend that you specify a byte adjustment value that represents the difference between the customer premise equipment (CPE) protocol overhead and the B-RAS protocol overhead.

The available range is -120 through 124 bytes. The system rounds up the byte adjustment value to the nearest multiple of 4. For example, a value of 6 is rounded to 8, and a value of -10 is rounded to -8.

## RELATED DOCUMENTATION

*Bandwidth Management for Downstream Traffic in Edge Networks Overview*

## Example: Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates

### IN THIS SECTION

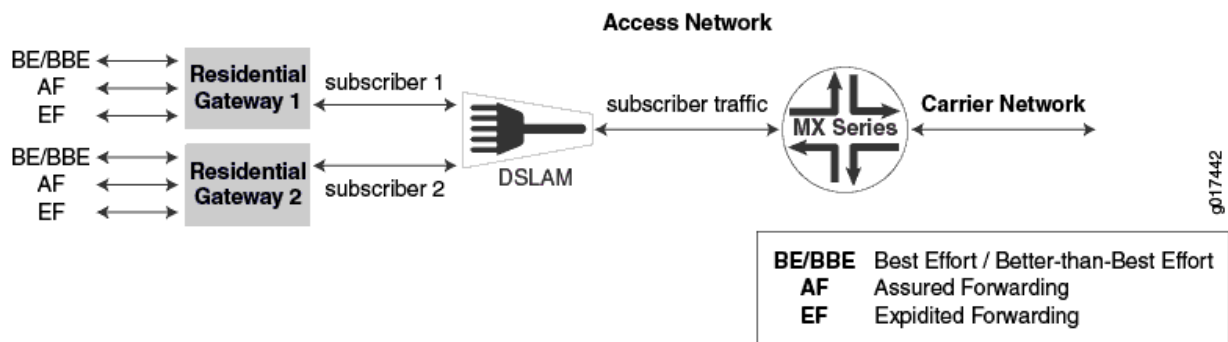
- [Managing Traffic with Different Encapsulations | 1144](#)
- [Managing Downstream Cell-Based Traffic | 1145](#)

This topic describes two scenarios for which you can configure static shaping parameters to account for packet overhead in a downstream network.

[Figure 67 on page 1144](#) shows the sample network that the examples reference.



Figure 67: Sample Network Topology for Downstream Traffic



## Managing Traffic with Different Encapsulations

In this example, the MX Series router shown in [Figure 67 on page 1144](#) sends stacked VLAN frames to the DSLAM, and the DSLAM sends single-tagged VLAN frames to the residential gateway.

To accurately shape traffic at the residential gateway, the MX Series router must account for the different frame sizes. The difference between the stacked VLAN (S-VLAN) frames sent by the router and the single-tagged VLAN frames received at the residential gateway is a 4-byte VLAN tag. The residential gateway receives frames that are 4 bytes less.

To account for the different frame sizes, the network administrator configures the frame shaping mode with -4 byte adjustment:

1. The network administrator configure the traffic shaping parameters and attaches them to the interface.

Enabling the overhead accounting feature affects the resulting shaping rate, guaranteed rate, and excess rate parameters, if they are configured.

```
[edit]
class-of-service {
  traffic-control-profiles {
    tcp-example-overhead-accounting-frame-mode {
      shaping-rate 10m;
      shaping-rate-priority-high 4m;
      guaranteed-rate 2m;
      excess-rate percent 50;
      overhead-accounting frame-mode bytes -4;
    }
  }
  interfaces {
```



```

        ge-1/0/0 {
            output-traffic-control-profile tcp-example-overhead-accounting-frame-mode;
        }
    }
}

```

2. The network administrator verifies the adjusted rates.

```

user@host#show class-of-service traffic-control-profile
Traffic control profile: tcp-example-overhead-accounting-frame-mode, Index: 61785
Shaping rate: 10000000
Shaping rate priority high: 4000000
Excess rate 50
Guaranteed rate: 2000000
Overhead accounting mode: Frame Mode
Overhead bytes: -4

```

## Managing Downstream Cell-Based Traffic

In this example, the DSLAM and residential gateway shown in [Figure 67 on page 1144](#) are connected through an ATM cell-based network. The MX Series router sends Ethernet frames to the DSLAM, and the DSLAM sends ATM cells to the residential gateway.

To accurately shape traffic at the residential gateway, the MX Series router must account for the different physical network characteristics.

To account for the different frame sizes, the network administrator configures the cell shaping mode with -4 byte adjustment:

1. Configure the traffic shaping parameters and attach them to the interface.

Enabling the overhead accounting feature affects the resulting shaping rate, guaranteed rate, and excess rate parameters, if they are configured.

```

[edit]
class-of-service {
    traffic-control-profiles {
        tcp-example-overhead-accounting-cell-mode {
            shaping-rate 10m;
            shaping-rate-priority-high 4m;
            guaranteed-rate 2m;
        }
    }
}

```



```

        excess-rate percent 50;
        overhead-accounting cell-mode;
    }
}
interfaces {
    ge-1/0/0 {
        output-traffic-control-profile tcp-example-overhead-accounting-cell-mode;
    }
}
}
}

```

## 2. Verify the adjusted rates.

```

user@host#show class-of-service traffic-control-profile
Traffic control profile: tcp-example-overhead-accounting-cell-mode, Index: 61785
Shaping rate: 10000000
Shaping rate priority high: 4000000
Excess rate 50
Guaranteed rate: 2000000
Overhead accounting mode: Cell Mode
Overhead bytes: 0

```

To account for ATM segmentation, the MX Series router adjusts all of the rates by 48/53 to account for ATM AAL5 encapsulation. In addition, the router accounts for cell padding, and internally adjusts each frame by 8 bytes to account for the ATM trailer.

## RELATED DOCUMENTATION

| *Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates*

## Traffic Burst Management on MIC and MPC Interfaces Overview

### IN THIS SECTION

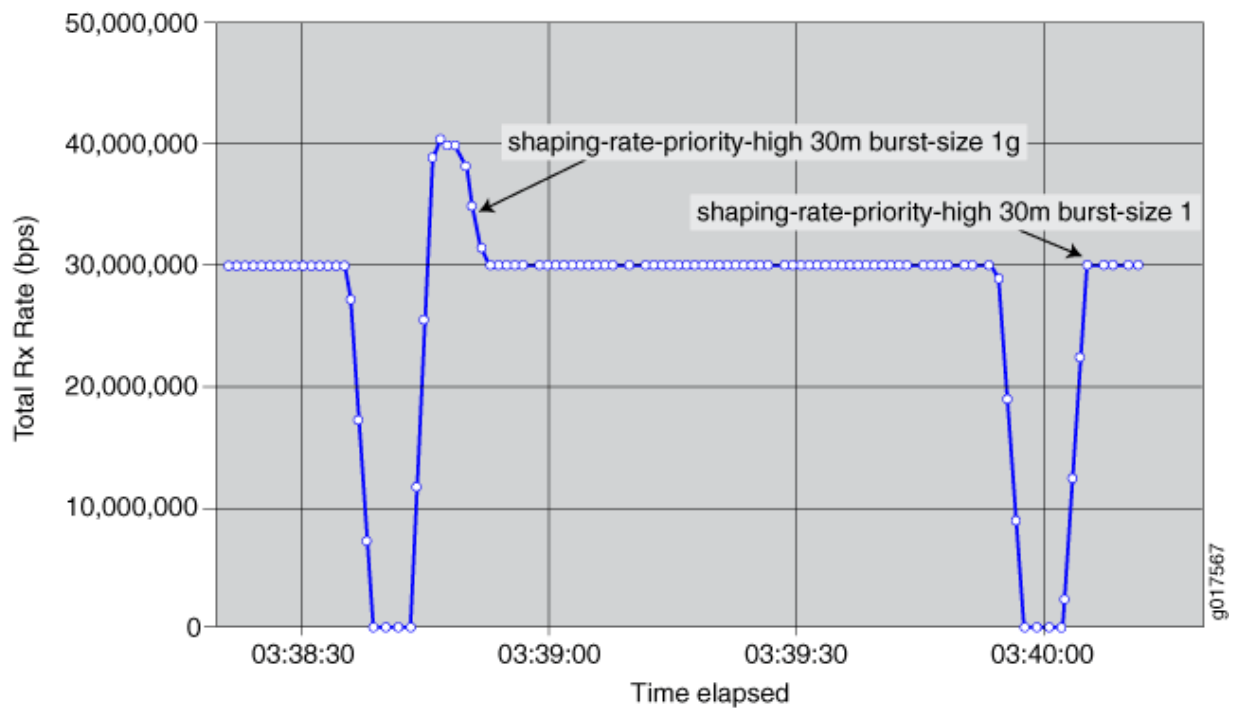
● [Guidelines for Configuring the Burst Size | 1148](#)



● How the System Calculates the Burst Size | 1149

You can manage the impact of bursts of traffic on your network by configuring a burst-size value with the shaping rate or the guaranteed rate. The value is the maximum bytes of rate credit that can accrue for an idle queue or scheduler node. When a queue or node becomes active, the accrued rate credits enable the queue or node to catch up to the configured rate.

**Figure 68: Sample Burst Shaping Rates**



In [Figure 68 on page 1147](#), the network administrator configures a large burst-size value for the shaping rate, then configures a small burst-size value. The larger burst size is subject to a maximum value. The smaller burst size is subject to a minimum value that enables the system to achieve the configured rates.

In both configurations, the scheduler node can burst beyond its shaping rate for a brief interval. The burst of traffic beyond the shaping rate is more noticeable with the larger burst size than the smaller burst size.



## Guidelines for Configuring the Burst Size

Typically, the default burst-size (100 ms) for both scheduler nodes and queues on MIC and MPC interfaces is adequate for most networks. However, if you have intermediate equipment in your network that has very limited buffering and is intolerant of bursts of traffic, you might want to configure a lower value for the burst size.

Use caution when selecting a different burst size for your network. A burst size that is too high can overwhelm downstream networking equipment, causing dropped packets and inefficient network operation. Similarly, a burst size that is too low can prevent the network from achieving your configured rate.

When configuring a burst size, keep the following considerations in mind:

- The system uses an algorithm to determine the actual burst size that is implemented for a node or queue. For example, to reach a shaping rate of 8 Mbps, you must allocate 1MB of rate credits every second. A shaping rate of 8 Mbps with a burst size of 500,000 bytes of rate-credit per seconds enables the system to transmit at most 500,000 bytes, or 4 Mbps. The system cannot implement a burst size that prevents the rate from being achieved.

For more information, see ["How the System Calculates the Burst Size" on page 1149](#).

- There are minimum and maximum burst sizes for each platform, and different nodes and queue types have different scaling factors. For example, the system ensures the burst cannot be set lower than 1 Mbps for a shaping rate of 8 Mbps. To smoothly shape traffic, rate credits are sent much faster than once per second. The interval at which rate credits are sent varies depending on the platform, the type of rate, and the scheduler level.
- When you have configured adjustments for the shaping rate (either by percentage or through an application such as ANCP or Multicast OIF), the system bases the default and minimum burst-size calculations on the adjusted shaping rate.
- When you have configured cell shaping mode to account for ATM cell tax, the system bases the default and minimum burst-size calculations on the post-tax shaping rate.
- The guaranteed rate and shaping rate share the value specified for the burst size. If the guaranteed rate has a burst size specified, that burst size is used for the shaping rate; if the shaping rate has a burst size specified, that bursts size is used for the guaranteed rate. If you have specified a burst size for both rates, the system uses the lesser of the two values.
- The burst size configured for the guaranteed rate cannot exceed the burst-size configured for the shaping rate. Starting in Junos OS Release 15.1, the CLI no longer generates a commit error when the guaranteed-rate burst size is statically configured to be more than the shaping-rate burst size. This behavior changed with the advent of enhanced subscriber management. The system logs an error when the guaranteed-burst rate is higher, whether it is configured statically, dynamically with predefined variables, or by means of a change of authorization request.



- If you have not configured a guaranteed rate, logical interfaces and interface sets receive a default guaranteed rate from the port speed. Queues receive a default guaranteed rate from the parent *logical interface* or interface set.
- Burst-size is not supported with per-priority-shaping.

## How the System Calculates the Burst Size

When calculating the burst size, the system uses an exponent of a power of two. For example:

Shaping-rate in bps \* 100 ms / (8 bits/byte \* 1000 ms/s) = 1,875,000 bytes

The system then rounds this value up. For example, the system uses the following calculation to determine the burst size for a scheduler node with a shaping rate of 150 Mbps:

Max (Shaping rate, Guaranteed rate) bps \* 100 ms / (8 bits/byte \* 1000 ms/s) = 1,875,000 bytes

Rounded up to the next higher power of two = 2,097,150 (which is 2\*\*21, or 0x200000)

The system assigns a single burst size to each of the following rate pairs:

- Shaping rate and guaranteed rate
- Guaranteed high (GH) and guaranteed medium (GM)
- Excess high (EH) and excess low (EL)
- Guaranteed low (GL)

To calculate the burst size for each pair, the system:

- Uses the configured burst-size if only one of the pair is configured.
- Uses the lesser of the two burst sizes if both values are configured.
- Uses the next lower power of two.
- To calculate the minimum burst size, the system uses the greater of the two rates.

### Change History Table

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

| Release | Description                                                                                                                                                                             |
|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15.1    | Starting in Junos OS Release 15.1, the CLI no longer generates a commit error when the guaranteed-rate burst size is statically configured to be more than the shaping-rate burst size. |



## RELATED DOCUMENTATION

[Per-Priority Shaping on MIC and MPC Interfaces Overview](#)

[Managing Excess Bandwidth Distribution on Static Interfaces on MICs and MPCs](#)

## Understanding Hierarchical Scheduling for MIC and MPC Interfaces

### IN THIS SECTION

- [Scheduler Node Scaling for MIC and MPC Interfaces | 1150](#)
- [Hierarchical Scheduling Priority Levels for MIC and MPC Interfaces | 1151](#)
- [Guaranteed Bandwidth and Weight of an Interface Node on MIC and MPC Interfaces | 1151](#)
- [Hierarchical Scheduling for MIC and MPC Interfaces in Oversubscribed PIR Mode | 1151](#)

### Scheduler Node Scaling for MIC and MPC Interfaces

In per-unit scheduling, the logical interfaces share a common level 2 node (one per port). In hierarchical-scheduling, each logical interface has its own level 2 node. Thus, scaling is limited by the number of level 2 nodes.

To better control system resources in hierarchical-scheduling mode, you can limit the number of scheduler node levels to two. In this case, all logical interfaces and interface sets with CoS scheduling policy share a single level 2 node. Consequently, the maximum number of logical interfaces with CoS scheduling policies is increased (the interface sets must be at level 3).

To configure scheduler node scaling, include the `hierarchical-scheduler` statement and set the `maximum-hierarchy-levels` option to 2 at the `[edit interfaces xe-fpc/pic/port]` hierarchy level.

```
[edit interfaces]
xe-2/0/0 {
    hierarchical-scheduler {
        maximum-hierarchy-levels 2;
    }
}
```





**NOTE:** The `maximum-hierarchy-levels` option supports level 3 interface sets but not level 2 interface sets. If you configure level 2 interface sets with the `maximum-hierarchy-levels` option, you generate Packet Forwarding Engine errors.

## Hierarchical Scheduling Priority Levels for MIC and MPC Interfaces

The queuing model used by MIC and MPC interfaces supports three priority levels for guaranteed scheduling priority and two lower priority levels for excess scheduling priority. You can configure a queue with one guaranteed priority and one excess priority. For example, you can configure a queue for guaranteed low (GL) as the guaranteed priority and configure excess high (EH) as the excess priority.

You can associate a guaranteed level with only one excess level. You can associate an excess level with any number of guaranteed priority levels, including none.

Interface nodes maintain their guaranteed priority level (for example, guaranteed high, GH) as long as they do not exceed their guaranteed bandwidth. If the queue bandwidth exceeds the guaranteed rate, then the priority drops to the excess priority (for example, excess high, EH). Because excess level priorities are lower than their guaranteed counterparts, the bandwidth guarantees for each of the other levels can be maintained.

## Guaranteed Bandwidth and Weight of an Interface Node on MIC and MPC Interfaces

The queuing model used by MIC and MPC interfaces separates the concepts of *guaranteed bandwidth* and *weight* of an interface node, although the two terms are often used interchangeably. The guaranteed bandwidth for an interface node is the bandwidth the node can use, independent of what is happening at the other nodes of the scheduling hierarchy. The weight of an interface node, on the other hand, is a value that determines how *excess bandwidth* is used. The weight of a node comes into play when other nodes at the same hierarchical scheduling level use less than the sum of their guaranteed bandwidths

For some application traffic types (such as constant bit rate voice, where there is little concern about excess bandwidth), the guaranteed bandwidth dominates the node. For other types of application traffic (such as bursty data, where a well-defined bandwidth is not always possible), the concept of weight dominates the node.

## Hierarchical Scheduling for MIC and MPC Interfaces in Oversubscribed PIR Mode

In contrast to the Intelligent Queuing Enhanced (IQE) and Intelligent Queuing 2 Enhanced (IQ2E) PICs, the interfaces on MICs and MPCs set the guaranteed rate to zero in oversubscribed peak information rate (PIR) mode for the per-unit scheduler. Also, the configured rate is scaled down to fit the oversubscribed value. For example, if there are two logical interface units with a shaping rate of 1 Gbps



each on a 1-Gbps port (which is, therefore, oversubscribed 2 to 1), then the guaranteed rate on each unit is scaled down to 500 Mbps (scaled down by 2).

With hierarchical schedulers in oversubscribed PIR mode, the guaranteed rate for every logical interface unit is set to zero. This means that the queue transmit rates are always oversubscribed.

Because in oversubscribed PIR mode the queue transmit rates are always oversubscribed, the following are true:

- If the queue transmit rate is set as a percentage, then the guaranteed rate of the queue is set to zero; but the excess rate (weight) of the queue is set correctly.
- If the queue transmit rate is set as an absolute value and if the queue has guaranteed high or medium priority, then traffic up to the queue's transmit rate is sent at that priority level. However, for guaranteed low traffic, that traffic is demoted to the excess low region. This means that best-effort traffic well within the queue's transmit rate gets a lower priority than out-of-profile excess high traffic. This differs from the IQE and IQ2E PICs.

## RELATED DOCUMENTATION

[Per-Unit Scheduling and Hierarchical Scheduling for MPC Interfaces | 1157](#)

[CoS Features and Limitations on MIC and MPC Interfaces | 1095](#)

[Jitter Reduction in Hierarchical CoS Queues | 1172](#)

[Scheduling and Shaping in Hierarchical CoS Queues for Traffic Routed to GRE Tunnels | 760](#)

*CoS Three-Level Hierarchical Scheduling on MPLS Pseudowire Subscriber Interfaces*

## Configuring Ingress Hierarchical CoS on MIC and MPC Interfaces

Use [Feature Explorer](#) to confirm platform and release support for specific features.

You can configure ingress CoS parameters, including hierarchical schedulers, on supported interfaces. In general, the supported configuration statements apply to per-unit schedulers or to hierarchical schedulers.



**NOTE:** Junos OS does not support ingress queuing and ingress hierarchical CoS on AE interfaces. You can, however, configure standard CoS classification and rewrite rules on AE interfaces.



To configure ingress CoS for per-unit schedulers, include the following statements at the [edit class-of-service interfaces *interface-name*] hierarchy level:

```
[edit class-of-service interfaces interface-name]
input-excess-bandwidth-share (proportional value | equal);
input-scheduler-map map-name;
input-shaping-rate rate;
input-traffic-control-profile profile-name;
unit logical-unit-number;
    input-scheduler-map map-name;
    input-shaping-rate (percent percentage | rate);
    input-traffic-control-profile profile-name;
```

To configure ingress CoS for hierarchical schedulers, include the interface-set *interface-set-name* statement at the [edit class-of-service interfaces] hierarchy level:

```
[edit class-of-service interfaces]
interface-set interface-set-name {
    input-traffic-control-profile profile-name;
    input-traffic-control-profile-remaining profile-name;
    interface interface-name {
        input-excess-bandwidth-share (proportional value | equal);
        input-traffic-control-profile profile-name;
        input-traffic-control-profile-remaining profile-name;
        unit logical-unit-number;
    }
}
```


By default, ingress CoS features are disabled interfaces. To enable ingress CoS on an interface, configure the traffic-manager statement with ingress-and-egress mode as shown in the following example:

```
chassis {
    fpc 7 {
        pic 0 {
            traffic-manager {
                mode ingress-and-egress;
            }
        }
    }
}
```



```
}  
}
```

Configured CoS features on the ingress are independent of CoS features on the egress.



**NOTE:** HQoS MPC cards installed in MX240, MX480, MX960, MX2008, MX2010, and MX2020 routers have a hardware limitation with an ingress queuing CoS "ingress-and-egress" configuration.

Ingress queuing can be enabled for a maximum of 10 ports per MIC Slot, resulting in 20 ports per MPC2E-3D-NG HQoS and MPC3E-3D-NG HQoS line card with 10 ports per MIC slot. In the XM chip there are 16 loopback streams allocated per port group for PG0 and PG1, where PG0 is mapped to MIC slot 0 and PG1 is mapped to MIC slot 1. On enabling ingress queuing on a PIC slot, one loopback stream from the XM chip is allocated per interface from the respective port group. Because there are only 16 loopback streams, out of which 2 are used by default and 4 are used for tunnel interfaces and inline services, 10 streams are left for ingress CoS.

Change History Table

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

| Release | Description                                                                                                                                                               |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17.4R2  | Starting with Junos 17.4R2 on supported devices, you can have precise control over which ports have ingress CoS enabled by configuring traffic-manager at the port level. |

RELATED DOCUMENTATION

| *mode (Layer 2 Tunneling Protocol Shaping)*

Configuring a CoS Scheduling Policy on Logical Tunnel Interfaces

You can configure a CoS scheduling policy on a logical tunnel interface (LT ifl). Logical tunnel interfaces can be used to terminate a pseudowire into a virtual routing and forwarding (VRF) instance. If an It device is used to terminate a pseudowire, CoS scheduling policies can be applied on the It interface to manage traffic entering the pseudowire. You accomplish this by configuring the hierarchical-scheduler attribute on the physical interface.





**NOTE:** It is important to first commit the hierarchical-scheduler configuration under the logical tunnel physical interface (LT ifd), and subsequently add and commit the class-of-service configuration.



**NOTE:** The output-traffic-control statement applies only to the LT ifl that is part of an L3 VRF instance.



**NOTE:** On QX chip-based MICs and MPCs only, LT ifls do not support shaping of Layer 2 pseudowires.

The following example shows two pseudowires (pw1 and pw2) over lt-1/0/10. pw1 carries data, voice, and video traffic, and pw2 carries only data and voice traffic. All pseudowire traffic is restricted to 800m bps. The shaping rate for traffic over pw1 is 400m bps and the shaping rate for traffic over pw2 is 400m bps.

```
[edit interfaces]
lt-1/0/10 {
    hierarchical-scheduler;
}

[edit class-of-service schedulers]
data_sch {
    buffer-size remainder;
    priority low;
}
voice_sch {
    transmit-rate 6k;
    priority strict-high;
}
video_sch {
    shaping-rate 1m;
    priority medium-low;
}

[edit class-of-service scheduler-maps]
pw1-smap {
    forwarding-class be scheduler data_sch;
    forwarding-class ef scheduler voice_sch;
    forwarding-class af scheduler video_sch;
```



```

}
pw2-smap {
    forwarding-class be scheduler data_sch;
    forwarding-class ef scheduler voice_sch;
}
[edit class-of-service traffic-control-profiles]
pw1-tcp {
    scheduler-map pw1-smap;
    shaping-rate 400m;
}
pw2-tcp {
    scheduler-map pw2-smap;
    shaping-rate 400m;
}
all-pw-tcp {
    shaping-rate 800m;
}
lt-ifd-remain {
    shaping-rate 10m;
}

[edit class-of-service interfaces]
lt-1/0/10 {
    output-traffic-control-profile all-pw-tcp;
    output-traffic-control-profile-remaining lt-ifd-remain;
unit 1 {
    output-traffic-control-profile pw1-tcp;
}
unit 3 {
    output-traffic-control-profile pw2-tcp;
}
}

```

## RELATED DOCUMENTATION

[CoS Scheduling Policy on Logical Tunnel Interfaces Overview | 1184](#)

[Hierarchical CoS for Metro Ethernet Environments | 448](#)

*Configuring Logical Tunnel Interfaces*

[CoS on Ethernet Pseudowires in Universal Edge Networks Overview | 1183](#)

[Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks | 1184](#)



## Per-Unit Scheduling and Hierarchical Scheduling for MPC Interfaces

### IN THIS SECTION

- [Scheduling Models Supported for MPC Interfaces | 1157](#)
- [Scheduler Node Levels for MPC Interfaces | 1158](#)

### Scheduling Models Supported for MPC Interfaces

Interfaces hosted on Modular Port Concentrator (MPC) line cards in MX Series 5G Universal Routing Platforms support the following models of class-of-service (CoS) scheduling, depending on MPC type:

#### Limited Scale Per-Unit Scheduling MPCs

Per-unit scheduling features on a limited scale are supported for interfaces hosted on some MPCs that do not have a dedicated queuing chip.

On MPCs that support per-unit scheduling, the following scheduling capabilities are available:

- Four or eight egress queues per unit.
- Delay buffer capacities of 100 ms by default, and up to 200 ms maximum delay.
- Rate shaping of the ports and their queues.
- Guaranteed rate enforced at the queues.

The per-unit scheduling features also support pre-classification of incoming packets to protect high priority packets in the event of congestion. Such features include ingress DSCP rewrite and per-VLAN classification, ingress and egress policing, and rewrites.

#### Hierarchical Scheduling MPCs

MPCs that provide a dedicated queuing chip support hierarchical scheduling.

Hierarchical scheduling MPCs support all per-unit scheduling functionality plus fine-grained queuing abilities over four or five levels of hierarchical scheduling:

- Hierarchical scheduling with ports, interface sets, and logical interfaces.



- Shaping—Committed Information Rate (CIR) and a peak information rate (PIR)—at all scheduling levels, including queues.
- Three normal- priority levels and two excess- priority levels configurable at all scheduling levels, including queues.
- Per-priority shaping of traffic at Level 1 or Level 2.
- Shaping for unconfigured customer VLANs (C-VLANs) and for service VLANs (S-VLANs).

## Scheduler Node Levels for MPC Interfaces

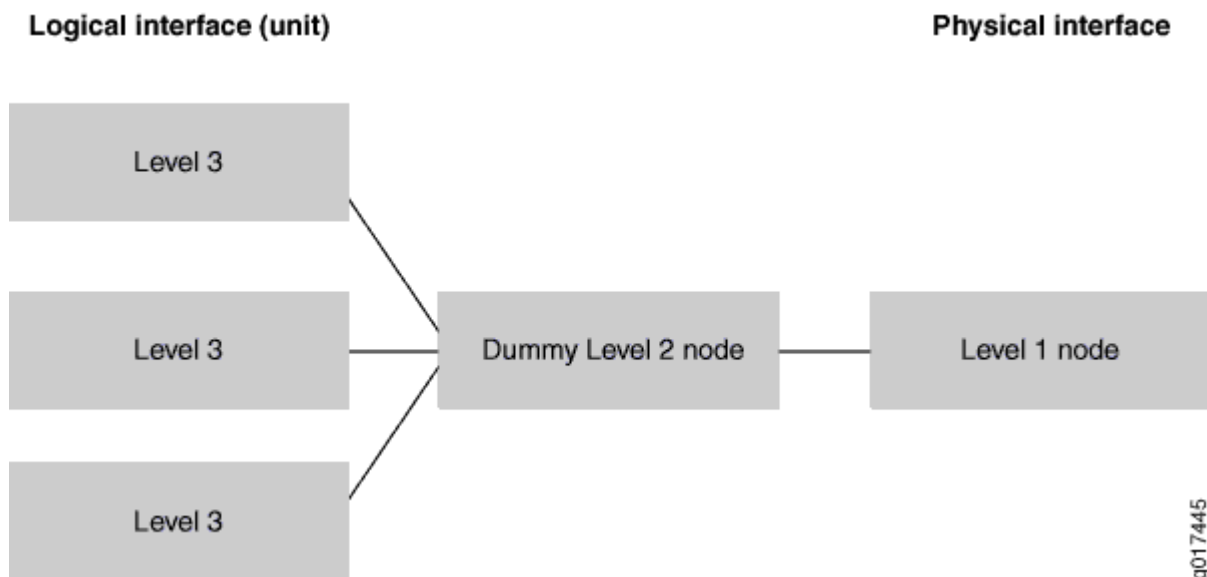
Interfaces hosted on MPCs support different scheduler node levels, depending on the MPC type:

### Scheduler Node Levels for Per-Unit Scheduling MPCs

For an interface hosted on a per-unit scheduling MPC, each logical interface has its own dedicated level 3 node, and all logical interfaces share a common level 2 node (one per port).

[Figure 69 on page 1158](#) illustrates scheduler node levels for an interface hosted on a per-unit queuing MPC.

**Figure 69: Scheduler Node Levels for Per-Unit Scheduling MPCs**



For interfaces hosted on per-unit scheduling MPCs, the level 2 node is always a dummy node.

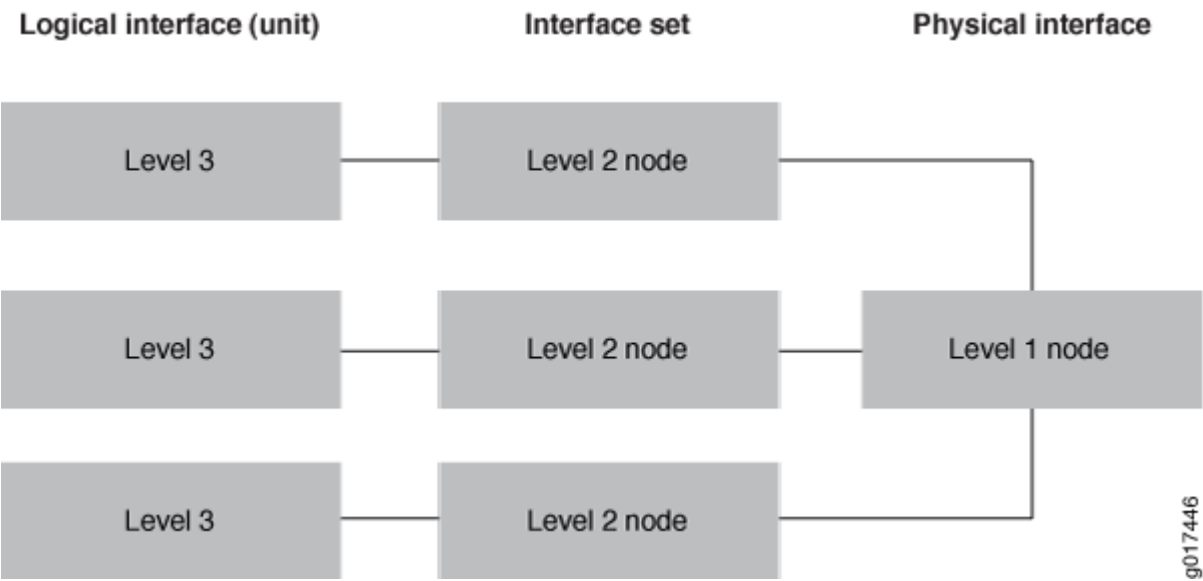


Scheduler Node Levels for Hierarchical Scheduling MPCs

The queuing model used by interfaces hosted on hierarchical scheduling MPCs supports up to five levels of scheduler nodes: the queue itself (level 5), session logical interface (ppp or dhcp) (level 4), customer VLAN (C-VLAN) (level 3), the interface set or service VLAN (S-VLAN) collection (level 2), and the physical interface or port (level 1).

Figure 70 on page 1159 illustrates the scheduler node levels for an interface hosted on a hierarchical scheduling MIC or MPC.

Figure 70: Scheduler Node Levels for Interfaces on Hierarchical Scheduling MPCs



The figure depicts scheduler nodes for an interface that does not include interface sets and for which traffic control profiles are applied to the logical interfaces only.

**NOTE:** If an interface set has a CoS scheduling policy but none of its child logical interfaces has a CoS scheduling policy, then the interface set is considered to be a leaf node and has one level 2 and one level 3 node.

RELATED DOCUMENTATION

|                                                                                                  |
|--------------------------------------------------------------------------------------------------|
| <a href="#">Understanding Hierarchical Scheduling for MIC and MPC Interfaces   1150</a>          |
| <a href="#">CoS Three-Level Hierarchical Scheduling on MPLS Pseudowire Subscriber Interfaces</a> |
| <a href="#">MX Series MPC Overview</a>                                                           |



[MPCs Supported by MX Series Routers](#)
[MX Series MIC Overview](#)
[MICs Supported by MX Series Routers](#)
[MX5, MX10, MX40, and MX80 Modular Interface Card Description](#)

## Managing Dedicated and Remaining Queues for Static CoS Configurations on MIC and MPC Interfaces

### IN THIS SECTION

- [Configuring the Maximum Number of Queues for MIC and MPC Interfaces | 1160](#)
- [Configuring Remaining Common Queues on MIC and MPC Interfaces | 1161](#)

This topic describes how to manage dedicated and remaining queues for static subscriber interfaces configured at the [edit class-of-service] hierarchy.

### Configuring the Maximum Number of Queues for MIC and MPC Interfaces

30-Gigabit Ethernet Queuing MPCs and 60-Gigabit Ethernet Queuing and Enhanced Queuing MPCs support a dedicated number of queues when configured for hierarchical scheduling and per-unit scheduling configurations.

To scale the number of subscriber interfaces per queue, you can modify the number of queues supported on the MIC.

To configure the number of queues:

1. Specify that you want to configure the MIC.

```
user@host# edit chassis fpc slot-number pic pic-number
```

2. Configure the number of queues.

```
[edit chassis fpc slot-number pic pic-number]  
user@host# set max-queues-per-interface (8 | 4)
```



## SEE ALSO

[Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview](#) | 1098

## Configuring Remaining Common Queues on MIC and MPC Interfaces

30-Gigabit Ethernet Queuing MPCs and 60-Gigabit Ethernet Queuing and Enhanced Queuing MPCs support a dedicated set of queues when configured with hierarchical scheduling.

When the number of dedicated queues is reached on the module, there can be queues remaining. Traffic from these logical interfaces are considered unclassified and attached to a common set of queues that are shared by all subsequent logical interfaces.

You can configure traffic shaping and scheduling resources for the remaining queues by attaching a special traffic-control profile to the interface. This feature enables you to provide the same shaping and scheduling to remaining queues as the dedicated queues.

To configure the remaining queues on a MIC or MPC interface:

1. Configure CoS parameters in a traffic-control profile.

```
[edit class-of-service]
user@host# edit traffic-control-profiles profile-name
```

2. Enable hierarchical scheduling for the interface.

```
[edit interfaces interface-name]
user@host# set hierarchical-scheduler
```

3. Attach the traffic control profiles for the dedicated and remaining queues to the port on which you enabled hierarchical scheduling.

To provide the same shaping and scheduling parameters to dedicated and remaining queues, reference the same traffic-control profile.

- a. Attach the traffic-control profile for the dedicated queues on the interface.

```
[edit class-of-service interfaces interface-name]
user@host# set output-traffic-control-profile profile-name
```



- b. Attach the traffic-control profile for the remaining queues on the interface.

```
[edit class-of-service interfaces interface-name]
user@host# set output-traffic-control-profile-remaining profile-name
```

## SEE ALSO

[Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview](#) | 1098

## RELATED DOCUMENTATION

*Dedicated Queue Scaling for CoS Configurations on MIC and MPC Interfaces Overview*

*Verifying the Number of Dedicated Queues Configured on MIC and MPC Interfaces*

[Hierarchical CoS for Metro Ethernet Environments](#) | 448

[Configuring Interface Sets](#) | 345

## Excess Bandwidth Distribution on MIC and MPC Interfaces Overview

Service providers often used tiered services to provide bandwidth for excess traffic as traffic patterns vary. By default, excess bandwidth between a configured guaranteed rate and shaping rate is shared equally among all queues on MIC and MPC interfaces, which might not be optimal for all subscribers to a service.

You can adjust this distribution by configuring the rates and priorities for the excess bandwidth.

By default, when traffic exceeds the shaping or guaranteed rates, the system demotes traffic with guaranteed high (GH) priority and guaranteed medium (GM) priority. You can disable this priority demotion for the MIC and MPC interfaces in your router.

## RELATED DOCUMENTATION

[Managing Excess Bandwidth Distribution on Static Interfaces on MICs and MPCs](#)

*Managing Excess Bandwidth Distribution for Dynamic CoS on MIC and MPC Interfaces*

[Per-Priority Shaping on MIC and MPC Interfaces Overview](#)

*Traffic Burst Management on MIC and MPC Interfaces Overview*



## Bandwidth Management for Downstream Traffic in Edge Networks

### Overview

#### IN THIS SECTION

- [Effective Shaping Rate | 1163](#)
- [Shaping Modes | 1164](#)
- [Byte Adjustments | 1164](#)
- [Relationship with Other CoS Features | 1165](#)

In a subscriber access network, traffic with different encapsulations can be passed downstream to other customer premise equipment (CPE) through the MX Series router. Managing the bandwidth of downstream ATM traffic to Ethernet interfaces can be especially difficult because of the different Layer 2 encapsulations.

The downstream network is not necessarily the directly attached network. In typical broadband network gateway (BNG) configurations, the directly attached network is an Ethernet access network, which provides access to either another frame-based network, or a cell-based network.

The *overhead accounting* feature enables you to shape traffic based on whether the downstream network is a frame-based network, like Ethernet, or a cell-based network, like ATM. It assigns a byte adjustment value to account for different encapsulations.

This feature is available on MIC and MPC interfaces.

### Effective Shaping Rate

The shaping-rate, also known as peak information rate (PIR), is the maximum rate for a scheduler node or queue.

The true rate of a subscriber at the access-loop/CPE is a function of:

- The shaping-rate in effect for the subscriber's household, in bits per second.
- Whether the subscriber is connected to a frame-based or cell-based network.
- Number of bytes in each frame that are accounted for by the shaper.





**NOTE:** Chassis `egress-shaping-overhead` is not included in the effective rate. Egress-shaping-overhead accounts for the physical interface overhead (ISO OSI Layer 1). Effective shaping-rate is a Layer 2 (ISO OSI) rate.

## Shaping Modes

There are two modes used for adjusting downstream traffic:

- *Frame shaping mode* is useful for adjusting downstream traffic with different encapsulations. Shaping is based on the number of bytes in the frame, without regard to cell encapsulation or padding overhead. Frame is the default shaping mode on the router.
- *Cell shaping mode* is useful for adjusting downstream cell-based traffic. In cell shaping mode, shaping is based on the number of bytes in cells, and accounts for the cell encapsulation and padding overhead.

When you specify cell mode, the resulting traffic stream conforms to the policing rates configured in downstream ATM switches, reducing the number of packet drops in the Ethernet network.

To account for ATM segmentation, the router adjusts all of the rates by 48/53 to account for 5-byte ATM AAL5 encapsulation. In addition, the router accounts for cell padding, and internally adjusts each frame by 8 bytes to account for the ATM trailer.

## Byte Adjustments

When the downstream traffic has different byte sizes per encapsulation, it is useful to configure a *byte adjustment* value to adjust the number of bytes per packet to be included in or excluded from the shaping mechanism. This value represents the number of bytes that are encapsulated and decapsulated by the downstream equipment. For example, to properly account for a 4-byte header stripped by the downstream network, set the overhead-accounting bytes to -4. To properly account for a 12-byte header added by the downstream network, set the overhead-accounting bytes to 12.

We recommend that you specify a byte adjustment value that represents the difference between the CPE protocol overhead and B-RAS protocol overhead.

The system rounds up the byte adjustment value to the nearest multiple of 4. For example, a value of 6 is rounded to 8, and a value of -10 is rounded to -8.

You do not need to configure a byte adjustment value to account for the downstream ATM network. However, you can specify the byte value to account for additional encapsulations or decapsulations in the downstream network.



## Relationship with Other CoS Features

Enabling the overhead accounting feature affects the resulting shaping rates, guaranteed rate, and excess rate parameters, if they are configured.

The overhead accounting feature also affects the egress shaping overhead feature that you can configure at the chassis level. We recommend that you use the egress shaping-overhead feature to account for the Layer 2 overhead of the outgoing interface, and use the overhead-accounting feature to account for downstream traffic with different encapsulations and cell-based networks.

When both features are configured, the total byte adjustment value is equal to the adjusted value of the overhead-accounting feature plus the value of the egress-shaping-overhead feature. For example, if the configured byte adjustment value is 40, and the router internally adjusts the size of each frame by 8, the adjusted overhead accounting value is 48. That value is added to the egress shaping overhead of 24 for a total byte adjustment value of 72.

### RELATED DOCUMENTATION

*Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates*

*Configuring Dynamic Shaping Parameters to Account for Overhead in Downstream Traffic Rates*

*Setting Shaping Rate and Overhead Accounting Based on PPPoE Vendor-Specific Tags*

## Scheduler Delay Buffering on MIC and MPC Interfaces

To control congestion at the output stage, you can configure the delay-buffer bandwidth. Scheduler delay-buffer bandwidth provides packet buffer space to absorb burst traffic up to the specified duration of delay. After the specified delay buffer becomes full, packets with 100 percent drop probability are dropped from the head of the buffer.

The buffer size is the amount of time in milliseconds of port bandwidth that a queue can use to continue to transmit packets during periods of congestion, before the buffer runs out and packets begin to drop.

For MIC and MPC interfaces the default maximum queue buffer size is:

- 500 ms for traffic rates below 1 Gbps.
- 100 ms for traffic rates of 1 Gbps and faster.
- 100 ms for all tunnel interfaces configured on MIC and MPC interfaces



You can configure an explicit buffer size ranging from 4 KB to 256 MB, depending on the MIC or MPC model. However, MIC and MPC interfaces do not support the large delay buffer size configuration statement `q-pic-large-buffer`.

Interfaces hosted on MIC and MPC line cards have a certain granularity in the application of configured delay buffer parameters. In other words, the observed hardware value might not exactly match the user-configured value. Nevertheless, the derived values are as close to the configured values as allowed.

When you configure an explicit buffer size, there are 256 points available and the closest point is chosen. High-priority and medium-priority queues use 64 points, and the low-priority queues uses 128.

## RELATED DOCUMENTATION

[CoS Features and Limitations on MIC and MPC Interfaces | 1095](#)

[Rate Shaping on MIC and MPC Interfaces | 1126](#)

[Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 1150](#)

[Drop Profiles on MIC and MPC Interfaces | 1169](#)

## Managing Excess Bandwidth Distribution on Static Interfaces on MICs and MPCs

Service providers often used tiered services that must provide bandwidth for excess traffic as traffic patterns vary. By default, excess bandwidth between a configured guaranteed rate and shaping rate is shared equally among all queues, which might not be optimal for all subscribers to a service.

To manage excess bandwidth:

1. Configure the parameters for the interface.
  - a. Configure the shaping rate.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set shaping-rate (percent percentage / rate) <burst-size bytes>
```



**TIP:** On MIC and MPC interfaces, the guaranteed rate and the shaping rate share the value specified for the burst size. If the guaranteed rate has a burst size specified, it is used for the shaping rate; if the shaping rate has a burst size specified, it is used



for the guaranteed rate. If you have specified a burst for both rates, the system uses the lesser of the two values.

b. Configure the excess rate.

You can configure an excess rate for all priorities of traffic.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set excess-rate (percent percentage | proportion value)
```

Optionally, you can configure an excess rate specifically for high- and low-priority traffic. When you configure the `excess-rate` statement for an interface, you cannot also configure the `excess-rate-low` and `excess-rate-high` statements.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set excess-rate-high (percent percentage | proportion value)
user@host# set excess-rate-low (percent percentage | proportion value)
```



**BEST PRACTICE:** We recommend that you configure either a percentage or a proportion of the excess bandwidth for all schedulers with the same parent in the hierarchy. For example, if you configure interface 1.1 with twenty percent of the excess bandwidth, configure interface 1.2 with eighty percent of the excess bandwidth.

2. (Optional) Configure parameters for the queue.

a. Configure the shaping rate.

```
[edit class-of-service scheduler scheduler-name]
user@host# set shaping-rate (rate | $junos-cos-scheduler-shaping-rate) <burst-size bytes>
```

b. Configure the excess rate.

```
[edit class-of-service scheduler scheduler-name]
user@host# set excess-rate (percent percentage | proportion value)
```



- c. (Optional) Configure the priority of excess bandwidth for the queue.

```
[edit class-of-service scheduler scheduler-name]
user@host#set excess-priority (low | medium-low | medium-high | high | none)
```



**TIP:** For queues, you cannot configure the excess rate in these cases:

- When the `transmit-rate exact` statement is configured. In this case, the shaping rate is equal to the transmit rate and the queue does not operate in the excess region.
- When the scheduling priority is configured as `strict-high`. In this case, the queue gets all available bandwidth and never operates in the excess region.

By default, when traffic exceeds the shaping or guaranteed rates, the system demotes traffic configured with guaranteed high (GH) priority and guaranteed medium (GM) priority. To disable priority demotion, specify the `none` option. You cannot configure this option for queues configured with `transmit-rate` expressed as a percent and when the parent's guaranteed rate is set to zero.

For example, the following statements establish a traffic control profile with a shaping rate of 80 Mbps and an excess rate of 100 percent.

```
[edit class-of-service traffic-control-profiles]
tcp-example-excess {
    shaping-rate 80m;
    excess-rate percent 100;
}
```

The following statements establish a scheduler with an excess rate of 5 percent and a low priority for excess traffic.

```
[edit class-of-service scheduler]
example-scheduler {
    excess-priority low;
    excess-rate percent 5;
}
```



## RELATED DOCUMENTATION

*Excess Bandwidth Distribution on MIC and MPC Interfaces Overview*

[Hierarchical CoS for Metro Ethernet Environments](#) | 448

## Drop Profiles on MIC and MPC Interfaces

### IN THIS SECTION

- [Drop Profiles on Enhanced Queuing MIC and MPC Interfaces](#) | 1169
- [Implicit Scaling of WRED Profiles](#) | 1169

This topic covers the following information

### Drop Profiles on Enhanced Queuing MIC and MPC Interfaces

Enhanced queuing (EQ) interfaces on MICs and MPCs support drop profiles as follows:

- Up to 255 drop profiles
- Up to 128 tail-drop priorities for guaranteed low (GL) priorities
- Up to 64 tail-drop priorities for guaranteed high and medium priorities

### Implicit Scaling of WRED Profiles

You can oversubscribe scheduler delay buffers by configuring more delay-buffer memory than the system can support. If you oversubscribe the scheduler delay buffers for MIC and MPC interfaces, the system implicitly scales down the configured weighted random early detection (WRED) profiles so that packets are dropped more aggressively from the relatively full queues. This automatic adjustment creates buffer space for packets in the relatively empty queues and provides a sense of fairness among the delay buffers.

## RELATED DOCUMENTATION

[CoS Features and Limitations on MIC and MPC Interfaces](#) | 1095



---

[Rate Shaping on MIC and MPC Interfaces | 1126](#)


---

[Understanding Hierarchical Scheduling for MIC and MPC Interfaces | 1150](#)


---

[Scheduler Delay Buffering on MIC and MPC Interfaces | 1165](#)


---

[RED Drop Profiles for Congestion Management | 464](#)


---

[Managing Transient Traffic Bursts by Configuring Weighted RED Buffer Occupancy | 496](#)


---

## Intelligent Oversubscription on MIC and MPC Interfaces Overview

On the MIC and MPC interfaces on MX Series routers, as on other types of interface hardware, arriving packets are assigned to one of two preconfigured traffic classes (network control and best effort) based on their header types and destination media access control (MAC) address. Oversubscription, the situation when the incoming packet rate is much higher than the Packet Forwarding Engine and system can handle, can cause key packets to be dropped and result in a flurry of resends, making the problem worse. However, MIC and MPC interfaces handle oversubscription more intelligently and drops lower priority packets when oversubscription occurs. Protocols such as routing protocols are classified as network control. Protocols such as telnet, FTP, and SSH are classified as best effort. No configuration is necessary.

The following frames and packets are assigned to the network control traffic class:

- ARPs: Ethertype 0x0806 for ARP and 0x8035 for dynamic RARP
- IEEE 802.3ad Link Aggregation Control Protocol (LACP): Ethertype 0x8809 and 0x01 or 0x02 (subtype) in first data byte
- IEEE 802.1ah: Ethertype 0x8809 and subtype 0x03
- IEEE 802.1g: Destination MAC address 0x01-80-C2-00-00-02 with Logical Link Control (LLC) 0xAAAA03 and Ethertype 0x08902
- PVST: Destination MAC address 0x01-00-0C-CC-CC-CD with LLC 0xAAAA03 and Ethertype 0x010B
- xSTP: Destination MAC address 0x01-80-C2-00-00-00 with LLC 0x424203
- GVRP: Destination MAC address 0x01-80-C2-00-00-21 with LLC 0x424203
- GMRP: Destination MAC address 0x01-80-C2-00-00-20 with LLC 0x424203
- IEEE 802.1x: Destination MAC address 0x01-80-C2-00-00-03 with LLC 0x424203
- Any per-port `my-mac` destination MAC address
- Any configured global Integrated Bridging and Routing (IRB) `my-mac` destination MAC address



In addition, the following Layer 3 control protocols are assigned to the network control traffic class:

- IGMP query and report: Ethertype 0x0800 and carrying an IPv4 protocol or IPv6 next header field set to 2 (IGMP)
- IGMP DVMRP: IGMP field version = 1 and type = 3
- IPv4 ICMP: Ethertype 0x0800 and IPv4 protocols = 1 (ICMP)
- IPv6 ICMP: Ethertype 0x86DD and IPv6 next header field = 0x3A (ICMP)
- IPv4 or IPv6 OSPF: Ethertype 0x0800 and IPv4 protocol field or IPv6 next header field = 89 (OSPF)
- IPv4 or IPv6 VRRP: IPv4 Ethertype 0x0800 or IPv6 Ethertype 0x86DD and IPv4 protocol field or IPv6 next header field = 112 (VRRP)
- IPv4 or IPv6 RSVP: IPv4 Ethertype 0x0800 or IPv6 Ethertype 0x86DD and IPv4 protocol field or IPv6 next header field = 46 or 134
- IPv4 or IPv6 PIM: IPv4 Ethertype 0x0800 or IPv6 Ethertype 0x86DD and IPv4 protocol field or IPv6 next header field = 103
- IPv4 or IPv6 IS-IS: IPv4 Ethertype 0x0800 or IPv6 Ethertype 0x86DD and IPv4 protocol field or IPv6 next header field = 124
- IPv4 router alert: IPv4 Ethertype 0x0800 and IPv4 option field = 0x94 (router alert)

Also, the following Layer 4 control protocols are assigned to the network control traffic class:

- IPv4 and IPv6 BGP: IPv4 Ethertype 0x0800 or IPv6 Ethertype 0x86DD, TCP port = 179, and carrying an IPv4 protocol or IPv6 next header field set to 6 (TCP)
- IPv4 and IPv6 LDP: IPv4 Ethertype 0x0800 or IPv6 Ethertype 0x86DD, TCP or UDP port = 646, and carrying an IPv4 protocol or IPv6 next header field set to 6 (TCP) or 17 (UDP)
- IPv4 UDP/L2TP control frames: IPv4 Ethertype 0x0800, UDP port = 1701, and carrying an IPv4 protocol field set to 17 (UDP)
- DHCP: Ethertype 0x0800, IPv4 protocol field set to 17 (UDP), and UDP destination port = 0x43 (DHCP service) or 0x44 (DHCP host)
- IPv4 or IPv6 UDP/BFD: Ethertype 0x0800, UDP port = 3784, and IPv4 protocol field or IPv6 next header field set to 17 (UDP)

Finally, any PPP encapsulation (Ethertype 0x8863 (PPPoE Discovery) or 0x8864 (PPPoE Session Control)) is assigned to the network control traffic class (queue 3).





**NOTE:** These classifications are preconfigured.

## Jitter Reduction in Hierarchical CoS Queues

### IN THIS SECTION

- [Queue Jitter as a Function of the Maximum Number of Queues | 1172](#)
- [Default Maximum Queues for Hierarchical Queuing MICs and MPCs | 1172](#)
- [Shaping Rate Granularity as a Function of the Rate Wheel Update Period | 1174](#)

### Queue Jitter as a Function of the Maximum Number of Queues

Each queuing chip on a Modular Interface Card (MIC) or Modular Port Concentrator (MPC) internally hosts a *rate wheel thread* that updates the *shaper credits* into the *shapers* available at each level of scheduling hierarchy. At each hierarchy level, the length of this update period determines two key characteristics of scheduling:

- The minimum buffer needed for the queue to pass packets without dropping.
- The degree of *jitter* encountered in the queue.

At each hierarchy level, the length of the rate wheel update period is dependent upon the number of entities enabled for that node level. Because traffic is queued at Level 5 (queues) and scheduled upwards to Level 1 (the port), the number of entities (queues) enabled at Level 5 determines the number of entities (logical interfaces, interface-sets, or ports) enabled at the other levels of the scheduling hierarchy. By extension, the number of queues enabled for a given scheduler node hierarchy determines the length of the update period at all hierarchy levels. Consequently, limiting the maximum number of queues supported by a hierarchical queuing MIC or MPC can reduce jitter in the queues. To configure the maximum number of queues allowed per hierarchical queuing MIC or MPC, include the `max-queues` statement at the `[edit chassis fpc slot-number]` hierarchy level.

### Default Maximum Queues for Hierarchical Queuing MICs and MPCs

The QX chip on a MIC or MPC consists of two symmetrical halves, and each half supports a maximum of 64 K queues (128 K queues per QX chip). The 2-port and 4-port 10-Gigabit Ethernet MICs with XFP and



the MPC1\_Q line cards have one chipset and can support a maximum of 128 K queues, distributed across the two partitions of the single QX chip. The MPC2 Q and MPC2 EQ line cards have two chipsets and can support a maximum of 256 K queues, distributed across the four partitions of the two QX chips.

[Table 144 on page 1173](#) lists the maximum number of queues supported by default and the corresponding rate wheel update period for each hierarchical queuing MIC or MPC.

**Table 144: Default Maximum Queues and Corresponding Rate Wheel Update Periods**

| Router Model                                        | Hierarchical Queuing MIC or MPC                                                                                                                                                                                                                                      | Maximum Queues | Rate Wheel Update Period |
|-----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------|--------------------------|
| MX5,<br>MX10,<br>MX40, and<br>MX80 modular          | <b>2-port 10-Gigabit Ethernet MIC with XFP</b><br>The chassis base board hosts one chipset-based Packet Forwarding Engine process that operates in standalone mode. The single QX chip is composed of two partitions that each support 64 K queues for egress ports. | 128 K          | 1.6 ms                   |
| MX240,<br>MX480,<br>MX960,<br>MX2010, and<br>MX2020 | <b>MPC1 Q</b><br>The MPC1 Q line card hosts one chipset-based Packet Forwarding Engine process that operates in fabric mode. The single QX chip is composed of two partitions that each support 64 K queues for egress ports.                                        | 128 K          | 1.6 ms                   |
|                                                     | <b>MPC2 Q</b><br>The MPC2 Q line card hosts two chipset-based Packet Forwarding Engine processes that operate in fabric mode. The two QX chips are composed of four partitions that each support 64 K queues for egress ports.                                       | 256 K          | 1.6 ms                   |
|                                                     | <b>MPC2 EQ</b><br>The MPC2 EQ line card hosts two chipset-based Packet Forwarding Engine processes that operate in fabric mode. The two QX chips are composed of four partitions that each support 64 K queues for egress ports.                                     | 256 K          | 2.6 ms                   |

You can configure hierarchical queuing MICs and MPCs to support a reduced maximum number of queues. Doing so reduces the rate wheel update period used by the QX chip, which in turn reduces jitter in the queues for the egress interfaces hosted on the line card.



## Shaping Rate Granularity as a Function of the Rate Wheel Update Period

Reducing the length of the QX chip rate wheel update period, in addition to reducing jitter in the hierarchical scheduling queues, also indirectly increases the shaping granularity.

For a given port line rate and scheduling hierarchy level, the shaping granularity is a function of the minimum shaper credit size and the rate wheel update period in effect as a result of the number of queues supported by the line card.

$$\text{shaping granularity} = \text{minimum shaper credit size} / \text{rate wheel update period}$$

Table 145 on page 1174 shows how shaping granularity is calculated for non-enhanced hierarchical queuing MIC and MPC line cards with default values for minimum shaper credit size and for rate wheel update period.

**Table 145: Default Shaping Granularities on Non-Enhanced Queuing MICs and MPCs**

| Port Type       | Hierarchy Level                     | Non-Enhanced Queuing MIC or MPC Defaults |                           | Calculation of Shaping Granularity           |
|-----------------|-------------------------------------|------------------------------------------|---------------------------|----------------------------------------------|
|                 |                                     | Minimum Credit                           | Update Period             |                                              |
| 1 Gbps Queuing  | Level 1 (port),<br>Level 4 (queues) | 4 bytes =<br>32 bits                     | 13.33 ms =<br>0.01333 sec | 32 bits / 0.01333 sec =<br><b>2.4 Kbps</b>   |
|                 | Level 2, Level 3                    | 16 bytes =<br>128 bits                   | 1.66 ms =<br>0.00166 sec  | 128 bits / 0.01333 sec =<br><b>9.6 Kbps</b>  |
| 10 Gbps Queuing | Level 1 (port),<br>Level 4 (queues) | 16 bytes =<br>128 bits                   | 13.33 ms =<br>0.01333 sec | 128 bits / 0.01333 sec =<br><b>9.6 Kbps</b>  |
|                 | Level 2, Level 3                    | 64 bytes =<br>512 bits                   | 1.66 ms =<br>0.00166 sec  | 512 bits / 0.01333 sec =<br><b>38.4 Kbps</b> |

## RELATED DOCUMENTATION

[Example: Reducing Jitter in Hierarchical CoS Queues | 1175](#)

[Per-Unit Scheduling and Hierarchical Scheduling for MPC Interfaces | 1157](#)



## Example: Reducing Jitter in Hierarchical CoS Queues

### IN THIS SECTION

- [Requirements | 1175](#)
- [Overview | 1175](#)
- [Configuration | 1176](#)

This example shows how to reduce jitter in the output queues for VLAN ports hosted on a hierarchical queuing MPC.

### Requirements

This example uses the following Juniper Networks hardware and Junos OS software:

- MX960 router in an IPv4 network and running Junos OS Release 13.2 or later.
- Available Gigabit Ethernet port hosted on FPC slot 2, PIC slot 0, port 0.
- Available Gigabit Ethernet port hosted on port 0 of a Gigabit Ethernet Modular Interface Card (MIC) in PIC slot 0 of an MPC2 Q Modular Port Concentrator (MPC) in FPC slot 5.

Before you begin configuring this example, make sure that the maximum number of queues allowed for the hierarchical queuing MPC in slot 5 has not yet been configured. When you enter the `show chassis fpc 5` command from configuration mode, the `max-queues` statement should not display.

### Overview

In this example you configure hierarchical scheduling on a VLAN port hosted on a hierarchical queuing MPC. To reduce jitter in the queues for all egress ports hosted on the MPC, reduce the maximum number of queues allowed for MPC.



## Configuration

### IN THIS SECTION

- [Verification](#) | 1181

### CLI Quick Configuration

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

```
set interfaces xe-2/0/0 per-unit-scheduler
set interfaces xe-2/0/0 flexible-vlan-tagging
set interfaces xe-2/0/0 unit 0 vlan-id 1
set interfaces xe-2/0/0 unit 0 family inet address 10.1.1.1/24
set interfaces xe-2/0/0 unit * classifiers ieee-802.1 ieee_jitter
set interfaces xe-5/0/0 per-unit-scheduler
set interfaces xe-5/0/0 flexible-vlan-tagging
set interfaces xe-5/0/0 unit 0 vlan-id 1
set interfaces xe-5/0/0 unit 0 family inet address 10.2.1.1/24
set class-of-service-interfaces xe-5/0/0 unit * output-traffic-control-profile tcp
set class-of-service forwarding-classes queue 0 be
set class-of-service forwarding-classes queue 1 ef
set class-of-service forwarding-classes queue 2 af
set class-of-service forwarding-classes queue 3 nc
set class-of-service schedulers be_sch priority low
set class-of-service schedulers ef_sch priority low
set class-of-service schedulers af_sch priority strict-high
set class-of-service schedulers nc_sch priority low
set class-of-service classifiers ieee_jitter forwarding-class be loss-priority low code-points
000
set class-of-service classifiers ieee_jitter forwarding-class ef loss-priority low code-points
001
set class-of-service classifiers ieee_jitter forwarding-class af loss-priority low code-points
010
set class-of-service classifiers ieee_jitter forwarding-class nc loss-priority low code-points
011
set class-of-service scheduler-maps smap_jitter forwarding-class be scheduler be_sch
```



```

set class-of-service scheduler-maps smap_jitter forwarding-class ef scheduler ef_sch
set class-of-service scheduler-maps smap_jitter forwarding-class af scheduler af_sch
set class-of-service scheduler-maps smap_jitter forwarding-class nc scheduler nc_sch
set class-of-service traffic-control-profiles tcp scheduler-map smap_jitter
set class-of-service traffic-control-profiles tcp shaping-rate 6g

```

## Baseline Configuration

### Step-by-Step Procedure

Configure hierarchical scheduling at xe-5.0.0.

1. To configure the VLAN 1 input and output at xe-2/0/0.0 and xe-5/0/0.0:

```

[edit]
user@host# set interfaces xe-2/0/0 per-unit-scheduler
user@host# set interfaces xe-2/0/0 flexible-vlan-tagging
user@host# set interfaces xe-2/0/0 unit 0 vlan-id 1
user@host# set interfaces xe-2/0/0 unit 0 family inet address 10.1.1.1/24

user@host# set interfaces xe-5/0/0 per-unit-scheduler
user@host# set interfaces xe-5/0/0 flexible-vlan-tagging
user@host# set interfaces xe-5/0/0 unit 0 vlan-id 1
user@host# set interfaces xe-5/0/0 unit 0 family inet address 10.2.1.1/24

```

2. Map each of four queues to a forwarding class.

```

[edit]
user@host# set class-of-service forwarding-classes queue 0 be
user@host# set class-of-service forwarding-classes queue 1 ef
user@host# set class-of-service forwarding-classes queue 2 af
user@host# set class-of-service forwarding-classes queue 3 nc

```

3. Assign a packet-scheduling priority value to each forwarding class.

```

[edit]
user@host# set class-of-service schedulers be_sch priority low
user@host# set class-of-service schedulers ef_sch priority low

```



```
user@host# set class-of-service schedulers af_sch priority strict-high
user@host# set class-of-service schedulers ef_sch priority low
```

4. Customize the default IEEE 802.1p classifier (BA classifier based on Layer 2 header) by defining different values for IEEE 802.1p code points.

```
[edit]
user@host# set class-of-service classifiers ieee_jitter forwarding-class be loss-priority low
code-points 000
user@host# set class-of-service classifiers ieee_jitter forwarding-class ef loss-priority low
code-points 001
user@host# set class-of-service classifiers ieee_jitter forwarding-class af loss-priority low
code-points 010
user@host# set class-of-service classifiers ieee_jitter forwarding-class nc loss-priority low
code-points 011
```

5. Apply the BA classifier to the input of the logical units on xe-2/0/0.

```
[edit]
user@host# set interfaces xe-2/0/0 unit * classifiers ieee-802.1 ieee_jitter
```

6. Configure the scheduler map smap\_jitter to map the forwarding classes to the schedulers.

```
[edit]
user@host# set class-of-service scheduler-maps smap_jitter forwarding-class be scheduler
be_sch
user@host# set class-of-service scheduler-maps smap_jitter forwarding-class ef scheduler
ef_sch
user@host# set class-of-service scheduler-maps smap_jitter forwarding-class af scheduler
af_sch
user@host# set class-of-service scheduler-maps smap_jitter forwarding-class nc scheduler
nc_sch
```



7. Configure the traffic control profile `tcp` to combine the scheduler map `smap_jitter` (that maps the forwarding classes to the schedulers for port-based scheduling) with a shaping rate (for hierarchical scheduling).

```
[edit]
user@host# set class-of-service traffic-control-profiles tcp scheduler-map smap_jitter
user@host# set class-of-service traffic-control-profiles tcp shaping-rate 6g
```

8. Apply the traffic control profile to the router output at `xe-5/0/0`.

```
[edit]
user@host# set class-of-service-interfaces xe-5/0/0 unit * output-traffic-control-profile tcp
```

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

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

## Results

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

```
[edit]
user@host# show interfaces
xe-2/0/0 {
  per-unit-scheduler;
  flexible-vlan-tagging;
  unit 0 {
    vlan-id 1;
    family inet {
      address 10.1.1.1/24;
    }
  }
}
xe-5/0/0 {
  per-unit-scheduler;
```



```
flexible-vlan-tagging;
unit 0 {
    vlan-id 1;
    family inet {
        address 10.2.1.1/24;
    }
}
}
```

```
[edit]
user@host# show class-of-service
classifiers {
    ieee-802.1 ieee_jitter {
        forwarding-class be {
            loss-priority low code-points 000;
        }
        forwarding-class ef {
            loss-priority low code-points 001;
        }
        forwarding-class af {
            loss-priority low code-points 010;
        }
        forwarding-class nc {
            loss-priority low code-points 011;
        }
    }
}
forwarding-classes {
    queue 0 be;
    queue 1 ef;
    queue 2 af;
    queue 3 nc;
}
traffic-control-profiles {
    tcp {
        scheduler-map smap_jitter;
        shaping-rate 6g;
    }
}
interfaces {
    xe-2/0/0 {
```



```

        unit * {
            classifiers {
                ieee-802.1 ieee_jitter;
            }
        }
    }
    xe-5/0/0 {
        unit * {
            output-traffic-control-profile tcp;
        }
    }
}
scheduler-maps {
    smap_jitter {
        forwarding-class be scheduler be_sch;
        forwarding-class ef scheduler ef_sch;
        forwarding-class af scheduler af_sch;
        forwarding-class nc scheduler nc_sch;
    }
}
schedulers {
    be_sch {
        priority low;
    }
    ef_sch {
        priority low;
    }
    af_sch {
        priority strict-high;
    }
    nc_sch {
        priority low;
    }
}
}

```

## Verification

### IN THIS SECTION

● [Measuring End-to-End Jitter to Establish the Baseline | 1182](#)



- [Configuring Jitter Reduction | 1182](#)
- [Measuring End-to-End Jitter to Verify Jitter Reduction | 1183](#)

Confirm that the configuration is working properly

### *Measuring End-to-End Jitter to Establish the Baseline*

#### **Purpose**

Establish a baseline measurement by noting the amount of jitter that occurs when the hierarchical queuing line card hosting the egress port is configured with the default maximum number of queues.

#### **Action**

To measure jitter:

1. Pass traffic through the VLAN.
2. Measure the variation in packet delay for selected packets in the data flow.

### *Configuring Jitter Reduction*

#### **Purpose**

Reduce jitter in the VLAN port output queues.

#### **Action**

1. Configure a reduced maximum number of queues for egress ports on the hierarchical queuing MPC in slot 5, thereby reducing the jitter in the port queues.

```
[edit]  
user@host# set chassis fpc 5 max-queue 64k
```



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

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

### *Measuring End-to-End Jitter to Verify Jitter Reduction*

#### **Purpose**

Measure the amount of jitter that occurs when the hierarchical queuing line card hosting the egress port is configured with a reduced maximum number of queues.

#### **Action**

To measure jitter:

1. Pass traffic through the VLAN.
2. Measure the variation in packet delay for selected packets in the data flow.

#### **RELATED DOCUMENTATION**

[Jitter Reduction in Hierarchical CoS Queues | 1172](#)

*max-queues*

## **CoS on Ethernet Pseudowires in Universal Edge Networks Overview**

You can apply *rewrite rules* and classifiers to an Ethernet pseudowire on MIC and MPC interfaces on MX Series routers. In an edge network, the pseudowire can represent a single customer.

To create the pseudowires, you use logical tunnel (LT) interfaces that connect two virtual routing forwarding (VRF) instances. To provide CoS to the LT interface, you can apply classifiers and rewrite rules. Rewrite rules enable you to rewrite packet header information by specifying various CoS values, including DiffServ code point (DSCP) and IP precedence.



**NOTE:** Scheduling is not supported on LT interfaces in the current release.



For example, a VPLS instance is connected to a Layer 3 routing instance. The logical tunnel labeled `lt-9/0/0.0` is configured with `vpls` as the family, and `lt-9/0/0.1` is configured with `inet` as the family. You can apply a rewrite rule and classifier for DSCP to `lt-9/0/0.1`, which can represent a business subscriber.

## RELATED DOCUMENTATION

[Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks | 1184](#)

## CoS Scheduling Policy on Logical Tunnel Interfaces Overview

You can configure a CoS scheduling policy on a logical tunnel interface (LT ifl). Logical tunnel interfaces can be used to terminate a pseudowire into a virtual routing and forwarding (VRF) instance. If an LT device is used to terminate a pseudowire, CoS scheduling policies can be applied on the LT interface to manage traffic entering the pseudowire. You accomplish this by configuring the hierarchical-scheduler attribute on the physical interface.

This feature is supported on MIC and MPC interfaces.

## RELATED DOCUMENTATION

[Configuring a CoS Scheduling Policy on Logical Tunnel Interfaces | 1154](#)

[Hierarchical CoS for Metro Ethernet Environments | 448](#)

[CoS on Ethernet Pseudowires in Universal Edge Networks Overview | 1183](#)

[Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks | 1184](#)

## Configuring CoS on an Ethernet Pseudowire for Multiservice Edge Networks

You can configure rewrite rules and classifiers to logical tunnel (LT) interfaces that are configured to represent Ethernet pseudowires.

This feature is supported on MIC and MPC interfaces.

To configure CoS on an LT interface configured for an Ethernet pseudowire:

1. Configure a pair of LT interfaces to represent a pseudowire.



To apply rewrite rules and classifiers to the pseudowire, you must assign one of the LT interfaces to the `inet` family.

```
[edit]
user@host#edit interfaces lt-fpc/pic/port
user@host#edit unit logical-unit-number
user@host#set encapsulation encapsulation
user@host#set family (inet | inet6 | iso | mpls)
user@host#set peer-unit unit-number
```

## 2. Configure the rewrite rule.

The available rewrite rule types for an LT interface are `dscp` and `inet-precedence`.

```
[edit class-of-service]
user@host#edit rewrite-rules (dscp | inet-precedence) rewrite-name
user@host#edit forwarding-class class-name
user@host#set loss-priority class-name code-point (alias | bits)
```

## 3. Configure the classifier.

The available classifier types for an LT interface are `dscp` and `inet-precedence`.

```
[edit class-of-service]
user@host#edit classifiers (dscp | inet-precedence) classifier-name
user@host#edit forwarding-class class-name
user@host#set loss-priority class-name code-points [aliases] [bit-patterns]
```

## 4. Apply the rewrite rule and classifier to the LT interface that you assigned to the `inet` family.

```
[edit class-of-service interfaces interface-name unit logical-unit-number]
user@host#set rewrite-rule (dscp | inet-precedence) (rewrite-name | default) protocol
protocol-types
user@host# set classifiers (dscp | inet-precedence) (classifier-name | default)
```

## RELATED DOCUMENTATION

[CoS on Ethernet Pseudowires in Universal Edge Networks Overview | 1183](#)

[Rewriting Packet Headers to Ensure Forwarding Behavior | 532](#)

[Understanding How Behavior Aggregate Classifiers Prioritize Trusted Traffic | 44](#)



## CoS for L2TP LNS Inline Services Overview

### IN THIS SECTION

- [Guidelines for Applying CoS to the LNS | 1186](#)
- [Hardware Requirements for Inline Services on the LNS | 1187](#)

You can apply hierarchical scheduling and per-session shaping to Layer 2 Tunnel Protocol (L2TP) network server (LNS) inline services using a static or dynamic CoS configuration.

This feature is supported on MIC and MPC interfaces on MX240, MX480, and MX960 routers.

### Guidelines for Applying CoS to the LNS

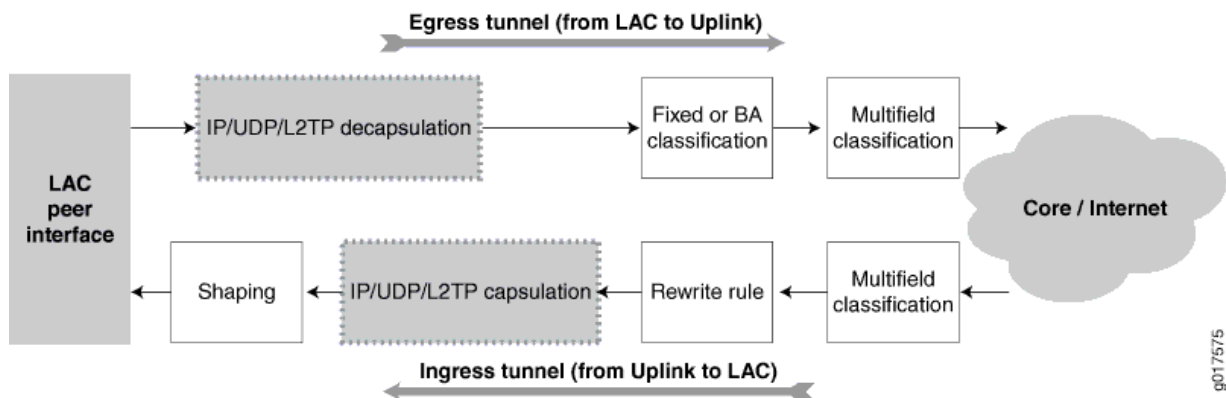
In L2TP configurations, IP, UDP, and L2TP headers are added to packets arriving at a PPP subscriber interface on the L2TP access concentrator (LAC) before being tunneled to the LNS.

When a service interface is configured for an L2TP LNS session, it has an *inner* IP header and an outer IP header. You can configure CoS for an LNS session that corresponds to the inner IP header only. The *outer* IP header is used for L2TP tunnel processing only.

However, we recommend that you configure classifiers and rewrite-rules to transfer the ToS (type of service) value from the inner IP header to the outer IP header of the L2TP packet.

[Figure 71 on page 1186](#) shows the classifier and *rewrite rules* that you can configure on an LNS inline service.

**Figure 71: Processing of CoS Parameters in an L2TP LNS Inline Service**





By default, the shaping calculation on the service interface includes the L2TP encapsulation. If necessary, you can configure additional adjustments for downstream ATM traffic from the LAC or differences in Layer 2 protocols.

## Hardware Requirements for Inline Services on the LNS

Hierarchical scheduling for L2TP LNS inline services is supported on MIC and MPC interfaces only. The services that you can configure depend on the hardware combination. [Table 146 on page 1187](#) lists the supported inline services and peer interfaces for each MIC and MPC combination.

**Table 146: Hardware Requirements for L2TP LNS Inline Services**

| MPC Module            | Inline Service Support—With Per-Session Shaping | Inline Service Support—Without Per-Session Shaping |
|-----------------------|-------------------------------------------------|----------------------------------------------------|
| MPC2E-3D-NG           | No                                              | Yes                                                |
| MPC2E-3D-NG-Q<br>MX80 | Yes                                             | Yes                                                |
| MPC-3D-16XGE-SFPP     | No                                              | No                                                 |

## RELATED DOCUMENTATION

[Configuring Static CoS for an L2TP LNS Inline Service | 1187](#)

*Configuring Dynamic CoS for an L2TP LNS Inline Service*

## Configuring Static CoS for an L2TP LNS Inline Service

Before you begin, configure the L2TP LNS inline service interface. See *Configuring an L2TP LNS with Inline Service Interfaces*.

You can configure hierarchical scheduling for an L2TP LNS inline service and manage the IP header values using rewrite rules and classifiers.

To configure static CoS for an L2TP LNS inline service:



1. Configure the hierarchical scheduler for the service interface (si) interface.

```
[edit interfaces si-fpc/port/pic ]
user@host# set hierarchical-scheduler maximum-hierarchy-levels 2
```



**BEST PRACTICE:** To enable Level 3 nodes in the LNS scheduler hierarchy and to provide better scaling, we recommend that you also specify a maximum of two hierarchy levels.

2. Configure the LNS to reflect the IP ToS value in the inner IP header to the outer IP header.

```
[edit services l2tp tunnel-group name]
user@host# set tos-reflect
```

3. Configure the classifier for egress traffic from the LAC:
  - a. Define the fixed or behavior aggregate (BA) classifier.

- To configure a fixed classifier:

```
[edit class-of-service interfaces si-fpc/port/pic unit logical-unit-number]
user@host# set forwarding-class class-name
```

- To configure a BA classifier:

```
[edit class-of-service]
user@host# set classifiers (dscp | dscp-ipv6 | inet-precedence) classifier-name
forwarding-class class-name loss-priority level code-points [ aliases ] [ bit-patterns]
```

- b. Apply the classifier to the service interface.

- To apply the classifier for the DSCP or DSCP IPv6 value:

```
[edit class-of-service interfaces si-fpc/port/pic unit logical-unit-number classifiers]
user@host# set dscp (classifier-name | default)
user@host# set dscp-ipv6 (classifier-name | default)
```



- To apply the classifier for the ToS value:

```
[edit class-of-service interfaces si-fpc/port/pic unit logical-unit-number classifiers]
user@host# set inet-precedence (classifier-name | default)
```

#### 4. Configure and apply a rewrite-rule to ingress traffic to the LAC:

- Configure the rewrite rule with the forwarding class and the loss priority value.

```
[edit class-of-service]
user@host# set rewrite-rules (dscp | dscp-ipv6 | inet-precedence) rewrite-name forwarding-
class class-name loss-priority level code-point (alias | bits)
```

- Apply the rewrite rule to the service interface.

- To apply the rewrite rule for the DSCP or DSCP IPv6 value:

```
[edit class-of-service interfaces si-fpc/port/pic unit logical-unit-number rewrite-
rules]
user@host# set dscp(rewrite-name | <default>) protocol protocol-types
user@host# set dscp-ipv6 (rewrite-name | <default>)
```

- To apply the rewrite rule for the ToS value:

```
[edit class-of-service interfaces si-fpc/port/pic unit logical-unit-number rewrite-
rules]
user@host# set inet-precedence (rewrite-name | <default>) protocol protocol-types
```

#### 5. (Optional) Configure additional adjustments for downstream ATM traffic.

By default, the shaping calculation on the service interface includes the L2TP encapsulation. If necessary, you can configure additional adjustments for downstream ATM traffic from the LAC or differences in Layer 2 protocols.

```
[edit class-of-service traffic-control-profiles profile-name]
user@host# set overhead-accounting (frame-mode | cell-mode) <bytes byte-value>
```



## 6. Apply the traffic-control profile.

```
[edit class-of-service interfaces si-fpc/port/pic unit logical-unit-number]
user@host# set output-traffic-control-profile profile-name
```



**BEST PRACTICE:** To limit bandwidth for tunneled sessions with default CoS configurations, we recommend that you also configure CoS for the remaining traffic on the static service interface.

```
[edit class-of-service interfaces si-fpc/port/pic ]
user@host# set output-traffic-control-profile-remaining profile-name
```

## RELATED DOCUMENTATION

[CoS for L2TP LNS Inline Services Overview | 1186](#)

*Configuring Static Shaping Parameters to Account for Overhead in Downstream Traffic Rates*

## CoS on Circuit Emulation ATM MICs Overview

The following class-of-service features are supported on Circuit Emulation ATM MICs:

- Traffic shaping and scheduling—Traffic shaping determines the maximum amount of traffic that can be transmitted on an interface.

You can configure three different categories of ATM service: constant bit rate (cbr), non-real-time variable bit rate (nrvbr), and real-time variable bit rate (rtvbr). The service category works in conjunction with ATM cell parameters `peak-rate`, `sustained-rate`, and `max burst-size` to impose traffic shaping, transmit rate, shaping rate, and default excess rate for an ATM queue.

Beginning with Junos OS Release 14.2, for an ATM MIC with SFP (MIC-3D-8OC3-2OC12-ATM), you can configure CBR bandwidth speeds up to 622 Mbps (OC12 speed) per virtual circuit (VC). To enable each VC on an ATM MIC with SFP to support up to line rate traffic in OC12 mode, define both of the following parameters in the traffic shaping and scheduling profile:

- Configure `cbr` as the ATM service category (`atm-service`).



- Configure up to a maximum of 1,412,829 cells per second (cps) as the ATM peak cell rate (peak-rate) .

To set the OC12 per-VC port speed for an ATM MIC with SFP, configure the `oc12-stm4` port speed option for the desired port.



**NOTE:** When you configure up to OC12 CBR bandwidth speed per VC on an ATM MIC with SFP, the actual Layer 3 payload throughput you obtain depends on the ATM encapsulation type and IP packet size that you use. The *Layer 3 payload* rate is the maximum Layer 3 (IP) payload throughput achieved for a given Layer 2 traffic rate.

- Policing—Policing, or rate limiting, enables you to limit the amount of traffic that passes into or out of the interface. It works with firewall filters to thwart denial-of-service (DoS) attacks.

Networks police traffic by limiting the input or output transmission rate of a class of traffic on the basis of user-defined criteria. The ATM policer controls the maximum rate of traffic sent from or received on the interface on which it is applied.

To apply limits to the traffic flow, configure the `cdvt` and `peak-rate` parameters within the policer. Define the `policing-action` parameter as `discard`, `discard-tag`, and `count` to set a consequence for the packets that exceed these limits. The consequence is usually a higher loss priority so that if the packets encounter downstream congestion, they are discarded first.

Change History Table

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

| Release | Description                                                                                                                                                                         |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 14.2    | Beginning with Junos OS Release 14.2, for an ATM MIC with SFP (MIC-3D-8OC3-2OC12-ATM), you can configure CBR bandwidth speeds up to 622 Mbps (OC12 speed) per virtual circuit (VC). |

RELATED DOCUMENTATION

[Configuring CoS on Circuit Emulation ATM MICs | 1192](#)

[Configuring Port Speed on Multi-Rate MICs](#)



## Configuring CoS on Circuit Emulation ATM MICs

On MX Series routers, you can configure the following class-of-service features on Circuit Emulation ATM MICs:

- Traffic shaping and scheduling—Traffic shaping determines the maximum amount of traffic that can be transmitted on an interface.
- Policing—Policing, or rate limiting, enables you to limit the amount of traffic that passes into or out of the interface. It works with firewall filters to thwart denial-of-service (DoS) attacks.

To configure a traffic shaping and scheduling profile on a Circuit Emulation ATM MIC:

1. Configure the traffic shaping and scheduling profile and specify the service category that determines the traffic shaping parameter for the ATM queue at the ATM MIC.

```
[edit class-of-service traffic-control-profile traffic-control-profile-name]
user@host# set atm-service (cbr | nrtvbr| rtvbr)
```



**NOTE:** Beginning with Junos OS Release 14.2, to configure up to OC12 constant bit rate (CBR) bandwidth speed per virtual circuit (VC) on an ATM MIC with SFP (MIC-3D-8OC3-2OC12-ATM), specify *cbr* as the ATM service category.

2. Configure the transmit rate, shaping rate, and default excess rate for the ATM queue.

```
[edit class-of-service traffic-control-profile traffic-control-profile-name]
user@host# set peak-rate peak-rate
user@host# set sustained-rate rate
user@host# set max-burst-size cells
```



**NOTE:** Beginning with Junos OS Release 14.2, to configure up to OC12 CBR bandwidth speed per VC on an ATM MIC with SFP, specify up to 1,412,829 cells per second (cps) as the ATM peak cell rate.

To configure an ATM policer for a Circuit Emulation ATM MIC:



1. Create a policer for each cell in the ATM packet. A policer defines the maximum traffic that can flow through an interface and further determines the actions to be taken when the traffic exceeds the defined limits.

```
[edit firewall]
user@host# set atm-policer atm-policer-name
```

2. Define the policer parameters. Configure the `atm-service` option. Apply limits to the traffic flow by configuring the `cdvt` and `peak-rate` parameters within the policer and define the `policing-action` parameter to set a consequence for the packets that exceed the traffic limits.

```
[edit firewall atm-policer atm-policer-name]
user@host# set logical-interface-policer
user@host# set atm-service (cbr | rtvbr | nrtvbr | ubr)
user@host# set cdvt rate
user@host# set peak-rate rate
user@host# set policing-action (discard | discard-tag | count)
```

3. Apply the traffic-shaping profile at the class-of-service interface level.

```
[edit class-of-service interfaces at-fpc/pic/port unit unit-number]
user@host# set output-traffic-control-profile traffic-control-profile-name
```

4. Apply the policer at the interface level.

```
[edit interfaces at-fpc/pic/port unit unit-number]
user@host# set atm-policer input-atm-policer atm-policer-name
```

You can verify the configuration by using the `show class-of-service traffic-control-profile traffic-control-profile-name` command.

## Change History Table



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

| Release | Description                                                                                                                                                                                                             |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 14.2    | Beginning with Junos OS Release 14.2, to configure up to OC12 constant bit rate (CBR) bandwidth speed per virtual circuit (VC) on an ATM MIC with SFP (MIC-3D-8OC3-2OC12-ATM), specify cbr as the ATM service category. |
| 14.2    | Beginning with Junos OS Release 14.2, to configure up to OC12 CBR bandwidth speed per VC on an ATM MIC with SFP, specify up to 1,412,829 cells per second (cps) as the ATM peak cell rate.                              |

## RELATED DOCUMENTATION

[ATM Support on Circuit Emulation PICs Overview](#)

[CoS on Circuit Emulation ATM MICs Overview | 1190](#)

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

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

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

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

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



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

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



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

## RELATED DOCUMENTATION

[swap-by-poppush](#)

[transparent](#)

[Understanding swap-by-poppush](#)

[Configuring IEEE 802.1p Inheritance push and swap from the Transparent Tag](#)

[Understanding Transparent Tag Operations and IEEE 802.1p Inheritance](#)

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

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

### Tagged Interface Example

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

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



```

    }
  }
}

```

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

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

```

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

```

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

### Untagged Interface Example



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

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

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

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

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



## RELATED DOCUMENTATION

*transparent*

[swap-by-poppush](#)

[Understanding IEEE 802.1p Inheritance push and swap from a Transparent Tag | 1194](#)

[Understanding swap-by-poppush](#)

[Understanding Transparent Tag Operations and IEEE 802.1p Inheritance](#)

## CoS on Application Services Modular Line Card Overview

### IN THIS SECTION

- [CoS Implementation in HTTP Reverse Proxy Scenario | 1199](#)
- [CoS Implementation in Transparent Proxy Scenario | 1200](#)
- [CoS Implementation in Mixed-Mode Scenario | 1200](#)

The Application Services Modular Line Card (AS MLC) is designed to run services for real-time traffic on MX240, MX480, and MX960 routers. It consists of three main components:

- Application Services Modular Carrier Card (AS MCC)
- Application Services Modular Processing Card (AS MXC)
- Application Services Modular Storage Card (AS MSC)

It supports class-of-service (CoS) features to ensure the *quality of service* (QoS) for real-time traffic that is sensitive to latency on a network. The AS MLC supports the following CoS features:

- **Code-point Aliases**—A code-point alias assigns a name to a pattern of code-point bits. On the AS MLC, you can use the code-point alias name for CoS components such as classifiers and drop-profile maps.
- **Classification**—Packet classification refers to the examination of ingress packets. On the AS MLC, the traffic flowing from the Modular Processing Card (AS MXC) towards the Modular Carrier Card (AS MCC) supports three types of classification:
  - **Behavior Aggregate (BA)**—BA classifier can be configured on the aggregated logical interfaces to classify traffic flowing from the AS MXC towards the AS MCC. With BA classification you can set the forwarding class and loss priority of a packet based on its code points. The AS MLC only



supports IP classification (classification based on Type of Service (ToS) and Differentiated Services Code Point (DSCP)) and classification is supported for the IPv4 family only. The Media Flow Controller application sets appropriate DSCP/ToS code-point in the packet that is evaluated by the BA classifier on the AS MCC to classify the packet.

- **Multifield Classification**—With multifield classifiers you can set the class and loss priority based on one or more of the following packet header fields: destination address, destination port, DSCP, IP protocol, and source address. Multifield classifiers are used when a simple BA classifier is insufficient to classify a packet.
- **Fixed Classification**—Fixed classification can be configured on logical interfaces by specifying a forwarding class to be applied to all packets received by the *logical interface*, regardless of the packet contents.
- **Scheduling**—Schedulers enable you to define the buffer sizes, delay buffer size, drop profile map, excess priority, excess rate percentage, output-traffic-control profile, priority, scheduler-map, shaping rate, transmit rate, and random early detection (RED) drop profiles to be applied to a particular queue for packet transmission.

The AS MLC provides CoS features in the following deployment scenarios:

- **HTTP Reverse Proxy**— In HTTP reverse proxy configurations, the service provider provides services to a set of domains (content providers) that buy content caching capability from the service provider. Clients connect to content providers through virtual IP (VIP) addresses. Service providers in the reverse proxy scenario generally deploy the routers with AS MLC hardware to honor service requests (such as caching) from the domain users.
- **HTTP Transparent Proxy**—In HTTP transparent proxy configurations, the service provider implements the AS MLC to improve its own caching capability and reduce the load on its own network. Implementing caching on an MX Series router with an AS MLC improves the retrieval speeds for data and optimizes the back-end network utilization.
- **Mixed Mode**—In mixed mode both reverse proxy and transparent proxy are configured on the same router.

## CoS Implementation in HTTP Reverse Proxy Scenario

In the reverse proxy configuration, the AS MXC provides content to multiple domains. The Media Flow Controller application on an AS MXC implements the differentiated services by setting the DSCP or IP precedence value for the IP packets traversing from the AS MXC to an AS MCC on the AS MLC hardware. The Modular Carrier Card uses these values to classify the packet and provide a suitable level of service.

The Media Flow Controller application detects the domain it serves and marks the DSCP values or the IP precedence bit value based on how important the traffic corresponding to that particular domain is. The



service provider operator also sets a behavior aggregate (BA) classifier on the aggregated interfaces on the AS MCC. Based on the DSCP/IP precedence bits, the classifier sets the forwarding class and packet loss priority for the packet. The forwarding class and the packet loss priority values govern the next-hop behavior of the packet traversing the Juniper Networks router.

Unlike a firewall, the Media Flow Controller application implemented on the AS MLC hardware marks the DSCP/IP precedence values based on the application layer protocols. This feature ensures that important traffic flowing from the AS MXC gets a higher priority and is processed accordingly. For example, if MPEG is implemented on the egress, the drop precedence for each frame can be different such that the P and B frames (which require more processing) are dropped before the I frames, resulting in a better quality video for the end user.

For traffic received on the ingress interfaces, end-to-end quality-of-service (QoS) policies ensure that the traffic arriving at the interface has the right DSCP values and the traffic is prioritized based on the forwarding class and packet loss priority values.

## **CoS Implementation in Transparent Proxy Scenario**

In the HTTP transparent proxy configuration, the service provider deploys the AS MLC hardware to reduce its own traffic instead of serving a particular domain. The Media Flow Controller application marks the DSCP bits based on its own requirements rather than those of the domains. Besides this, the CoS implementation for the egress interface is similar to the reverse proxy configuration scenario. The incoming packets follow the QoS policies applied at the WAN interface.

## **CoS Implementation in Mixed-Mode Scenario**

In mixed mode both reverse proxy and transparent proxy configuration coexist on the same AS MLC hardware. In such a scenario, reverse proxy is configured on an aggregated interface and transparent proxy is configured on a regular interface with the Media Flow Controller application marking the appropriate DSCP values for both the configurations. The individual CoS implementation in both the scenarios remains similar to the implementation discussed in ["CoS Implementation in HTTP Reverse Proxy Scenario" on page 1199](#) and ["CoS Implementation in Transparent Proxy Scenario" on page 1200](#)



# Configuring Class of Service on Aggregated, Channelized, and Gigabit Ethernet Interfaces

## IN THIS CHAPTER

- [Limitations on CoS for Aggregated Interfaces | 1201](#)
- [Policer Support for Aggregated Ethernet Interfaces Overview | 1203](#)
- [Understanding Schedulers on Aggregated Interfaces | 1205](#)
- [Examples: Configuring CoS on Aggregated Interfaces | 1205](#)
- [Hierarchical Schedulers on Aggregated Ethernet Interfaces Overview | 1209](#)
- [Configuring Hierarchical Schedulers on Aggregated Ethernet Interfaces | 1210](#)
- [Example: Configuring Scheduling Modes on Aggregated Interfaces | 1211](#)
- [Enabling VLAN Shaping and Scheduling on Aggregated Interfaces | 1217](#)
- [Class of Service on demux Interfaces | 1219](#)
- [Example: Configuring Per-Unit Schedulers for Channelized Interfaces | 1223](#)
- [Applying Layer 2 Policers to Gigabit Ethernet Interfaces | 1226](#)

## Limitations on CoS for Aggregated Interfaces

Both Ethernet and SONET/SDH interfaces can be aggregated. The limitations covered here apply to both.

There are some restrictions when you configure CoS on aggregated Ethernet and SONET/SDH interfaces:

- Chassis scheduling, described in ["Applying Scheduler Maps to Chassis-Level Queues" on page 991](#), is not supported on aggregated interfaces, because a chassis scheduler applies to the entire PIC and not just to one interface.
- An aggregated interface is a pseudo-interface. Therefore, CoS queues are not associated with the aggregated interface. Instead, CoS queues are associated with the member link interfaces of the aggregated interface.



- When you apply CoS parameters to the aggregated interface, they are applied to the CoS queues of the member link interfaces. You do not, however, apply CoS classifiers and *rewrite rules* directly to the member link interfaces.
- You cannot apply a scheduler map to a member link of an aggregate interface.
- Rate-based CoS components such as scheduler, shaper, and policer are not supported on mixed rate aggregated Ethernet links. However, the default CoS settings are supported by default on the mixed rate aggregated Ethernet links.
- Ingress queuing is not supported on aggregated interfaces.

When the scheduler map of the aggregate interface has schedulers configured for absolute transmit rate, the scheduler for the member link interfaces is scaled to the speed of each member link interface. Each member link interface has an automatic scheduler map that is not visible in the CLI. This scheduler map is allocated when the member link is added to the aggregate interface and is deleted when the member link is removed from the aggregate interface.

- If you configure the scheduler transmit rate of the aggregate interface as an absolute rate, the software uses the following formula to scale the transmit rate of each member link:

```
transmit rate of member link interface =
(configured transmit rate of aggregate interface /
total speed of aggregate interface) *
(total speed of member link interface / total configured percent) * 100
```

- If you configure the scheduler transmit rate of the aggregate interface as a percentage, the software uses the following formula to scale the transmit rate of each member link:

```
transmit rate percent of member link interface =
(configured transmit rate percent of aggregate interface /
total configured percent) * 100
```

The total configured percent is the sum of the configured transmit rate of all schedulers in terms of percentage of the total speed of the aggregate interface.

- All the other parameters for the schedulers, including priority, drop profile, and buffer size, are copied without change from the scheduler of the aggregated interface to the member link interfaces.
- The configuration related to the logical interfaces, including classifiers and rewrite rules, is copied from the aggregated *logical interface* configuration to the member link logical interfaces.



- For the scheduler map applied to an aggregated interface, if you configure a transmission rate in absolute terms, then the traffic of all the member link interfaces might be affected if any of the member link interfaces go up or down.

When applying CoS configurations to bundles, you must apply the CoS configuration directly to the bundle, not to the physical ports that are part of the bundle. The device may give you a false commit if you apply a CoS configuration directly to a physical port that is part of a bundle. This limitation applies if you attempt to configure a physical port that is already a member of a bundle or if you attempt to add a physical port to a bundle that already has a CoS configuration applied to it.

If you want to add a physical port to a bundle that already has a CoS configuration, you must:

1. Remove the CoS configuration from the port.
2. Commit your changes on the device.
3. Add the port to the bundle. The CoS configurations that are present on the bundle will be applied to the port you are adding to the bundle.
4. Commit your changes on the device.

In addition, if you want to remove a physical port from a bundle and ensure the physical port has the appropriate CoS configurations, you must:

1. Remove the port from the bundle.
2. Commit your changes on the device.
3. Apply the applicable CoS configuration to the port.
4. Commit your changes on the device.

## RELATED DOCUMENTATION

| [Examples: Configuring CoS on Aggregated Interfaces](#) | 1205

## Policer Support for Aggregated Ethernet Interfaces Overview

Aggregated interfaces support single-rate policers, three-color marking policers, two-rate three-color marking policers, hierarchical policers, and percentage-based policers. By default, policer bandwidth and burst-size applied on aggregated bundles is not matched to the user-configured bandwidth and burst-size.



You can configure interface-specific policers applied on an aggregated Ethernet bundle or an aggregated SONET bundle to match the effective bandwidth and burst-size to user-configured values. The `shared-bandwidth-policer` statement is required to achieve this match behavior.

This capability applies to all interface-specific policers of the following types: single-rate policers, single-rate three-color marking policers, two-rate three-color marking policers, and hierarchical policers. Percentage-based policers match the bandwidth to the user-configured values by default, and do not require `shared-bandwidth-policer` configuration. The `shared-bandwidth-policer` statement causes a split in burst-size for percentage-based policers.



**NOTE:** This feature is supported on the following platforms: T Series routers (excluding T4000 Type 5 FPCs), M120, M10i, M7i (CFEB-E only), M320 (SFPC only), MX240, MX480, and MX960 with DPC, MIC, and MPC interfaces, and EX Series switches.

The following usage scenarios are supported:

- Interface policers used by the following configuration:

```
[edit] interfaces (aeX | asX) unit unit-num family family policer [input | output | arp]
```

- Policers and three-color policers (both single-rate three-color marking and two-rate three-color marking) used inside interface-specific filters; that is, filters that have an interface-specific keyword and are used by the following configuration:

```
[edit] interfaces (aeX | asX) unit unit-num family family filter [input | output]
```

- Common-edge service filters, which are derived from CLI-configured filters and thus inherit interface-specific properties. All policers and three-color policers used by these filters are also affected.

The following usage scenarios are not supported:

- Policers and three-color policers used inside filters that are not interface specific; such a filter is meant to be shared across multiple interfaces.
- Any implicit policers or policers that are part of implicit filters; for example, the default ARP policer applied to an aggregate Ethernet interface. Such a policer is meant to be shared across multiple interfaces.
- Prefix-specific action policers.



To configure this feature, include the `shared-bandwidth-policer` statement at the following hierarchy levels: `[edit firewall policer policer-name]`, `[edit firewall three-color-policer policer-name]`, or `[edit firewall hierarchical-policer policer-name]`.

## RELATED DOCUMENTATION

| [\*shared-bandwidth-policer\*](#)

## Understanding Schedulers on Aggregated Interfaces

You can apply a class-of-service (CoS) configuration to aggregated Ethernet and aggregated SONET/SDH interfaces. The CoS configuration applies to all member links included in the aggregated interface. You cannot apply different CoS configurations to the individual member links.

You can configure shaping for aggregated Ethernet interfaces that use interfaces originating from Gigabit Ethernet IQ2 PICs. However, you cannot enable shaping on aggregated Ethernet interfaces when there is a mixture of ports from Intelligent Queuing (IQ) and Intelligent Queuing 2 (IQ2) PICs in the same bundle.

You cannot configure a shaping rate and guaranteed rate on an aggregated Ethernet interface with member interfaces on IQ or IQ2 PICs. The commit will fail. These statements are allowed only when the member interfaces are Enhanced Queuing DPC Gigabit Ethernet interfaces.

To view the summation of the queue statistics for the member links of an aggregate interface, issue the `show interfaces queue` command. To view the queue statistics for each member link, issue the `show interfaces queue aggregated-interface-name` command.

## RELATED DOCUMENTATION

| [Configuring Schedulers | 338](#)

| [Configuring Scheduler Maps | 338](#)

| [Applying Scheduler Maps Overview | 339](#)

## Examples: Configuring CoS on Aggregated Interfaces

This example illustrates how CoS scheduler parameters are configured and applied to aggregated interfaces.



### Applying Scaling Formula to Absolute Rates

Configure queues as follows when the total speed of member link interfaces is 100 Mbps (the available bandwidth is 100 Mbps):

```
[edit class-of-service]
schedulers {
  be {
    transmit-rate 10m;
  }
  af {
    transmit-rate 20m;
  }
  ef {
    transmit-rate 80m;
  }
  nc {
    transmit-rate 30m;
  }
}
```

The total configured transmit rates of the aggregated interface is  $10\text{m} + 20\text{m} + 80\text{m} + 30\text{m} = 140\text{ Mbps}$ , meaning the transmit rate is overconfigured by 40 percent. Therefore, the software scales down the configuration to match the 100 Mbps of available bandwidth, as follows:

```
be = (10/140) * 100 = 7 percent of 100 Mbps = 7 Mbps
af = (20/140) * 100 = 14 percent of 100 Mbps = 14 Mbps
ef = (80/140) * 100 = 57 percent of 100 Mbps = 57 Mbps
nc = (30/140) * 100 = 21 percent of 100 Mbps = 21 Mbps
```

### Applying Scaling Formula to Mixture of Percent and Absolute Rates

Configure the following mixture of percent and absolute rates:

```
[edit class-of-service]
schedulers {
  be {
    transmit-rate 20 percent;
  }
  af {
    transmit-rate 40 percent;
```



```

    }
    ef {
        transmit-rate 150m;
    }
    nc {
        transmit-rate 10 percent;
    }
}

```

Assuming 300 Mbps of available bandwidth, the configured percentages correlate with the following absolute rates:

```

schedulers {
    be {
        transmit-rate 60m;
    }
    af {
        transmit-rate 120m;
    }
    ef {
        transmit-rate 150m;
    }
    nc {
        transmit-rate 30m;
    }
}

```

The software scales the bandwidth allocation as follows:

```

be = (60/360) * 100 = 17 percent of 300 Mbps = 51 Mbps
af = (120/360) * 100 = 33 percent of 300 Mbps = 99 Mbps
ef = (150/360) * 100 = 42 percent of 300 Mbps = 126 Mbps
nc = (30/360) * 100 = 8 percent of 300 Mbps = 24 Mbps

```

### Configuring an Aggregated Ethernet Interface

Configure an aggregated Ethernet interface with the following scheduler map:

```

[edit class-of-service]
scheduler-maps {
    aggregated-sched {

```



```

        forwarding-class be scheduler be;
        forwarding-class af scheduler af;
        forwarding-class ef scheduler ef;
        forwarding-class nc scheduler nc;
    }
}
schedulers {
    be {
        transmit-rate percent 10;
        buffer-size percent 25;
    }
    af {
        transmit-rate percent 20;
        buffer-size percent 25;
    }
    ef {
        transmit-rate 80m;
        buffer-size percent 25;
    }
    nc {
        transmit-rate percent 30;
        buffer-size percent 25;
    }
}

```

In this case, the transmission rate for the member link scheduler map is as follows:

- be—7 percent
- af—14 percent
- ef—57 percent
- nc—21 percent

If you add a Fast Ethernet interface to the aggregate, the aggregate bandwidth is 200 Mbps, and the transmission rate for the member link scheduler map is as follows:

- be—10 percent
- af—20 percent
- ef—40 percent
- nc—30 percent



## RELATED DOCUMENTATION

[Limitations on CoS for Aggregated Interfaces | 1201](#)

## Hierarchical Schedulers on Aggregated Ethernet Interfaces Overview

On MX Series routers, you can apply hierarchical schedulers on aggregated ethernet bundles using interface sets. This feature enables you to configure a group of virtual LANs (VLANs) and control their bandwidth. This feature is supported at egress only.

You can configure interface sets for aggregated Ethernet (AE) interfaces created under static configurations. You can configure class-of-service parameters on AE interfaces, in either link-protect or non-link-protect mode. You can configure these parameters at the AE physical interface level. The CoS configuration is fully replicated for all AE member links in link-protect mode. You can control the way these parameters are applied to member links in non-link-protect mode by configuring the AE interface to operate in scaled mode or replicate mode.

The link membership list and scheduler mode of the interface set is inherited from the underlying aggregated Ethernet interface over which the interface set is configured. When an aggregated Ethernet interface operates in link protection mode, or if scheduler mode is configured to replicate member links, the scheduling parameters of the interface set are copied to each of the member links.

If the scheduler mode of the aggregated Ethernet interface is set to scale member links, the scheduling parameters are scaled based on the number of active member links (scaling factor is  $1/A$  where  $A$  is the number of active links in the bundle) and applied to each of the AE interface member links.

To configure an interface set, include the `interface-set` statement at the `[edit class-of-service interfaces]` hierarchy level.

To apply scheduling and queuing parameters to the interface set, include the `output-traffic-control-profile profile-name` statement at the `[edit class-of-service interfaces interface-name interface-set interface-set-name]` hierarchy level.

To apply an output traffic scheduling and shaping profile for the remaining traffic to the *logical interface* or interface set, include the `output-traffic-control-profile-remaining profile-name` statement at the `[edit class-of-service interfaces interface-name]` hierarchy level or the `[edit class-of-service interfaces interface-name interface-set interface-set-name]` hierarchy level.

## RELATED DOCUMENTATION

[Configuring Hierarchical Schedulers on Aggregated Ethernet Interfaces | 1210](#)

*output-traffic-control-profile-remaining*



## Configuring Hierarchical Schedulers on Aggregated Ethernet Interfaces

The following example shows the creation of an interface set for aggregated Ethernet interfaces in a static Ethernet configuration.

To configure interface sets for aggregated Ethernet (AE) interfaces created under static configurations:

1. Create the AE interfaces.

```
[edit]
user@host# show chassis | display set
set chassis aggregated-devices ethernet device-count 10
```

2. Configure the AE physical interfaces and member links.

```
user@host# show interfaces | display set

set interfaces ge-5/2/0 gigether-options 802.3ad ae0
set interfaces ge-5/2/1 gigether-options 802.3ad ae0
set interfaces ae0 hierarchical-scheduler maximum-hierarchy-levels 2
set interfaces ae0 flexible-vlan-tagging
set interfaces ae0 unit 0 vlan-id 100
set interfaces ae0 unit 1 vlan-id 101
set interfaces ae0 unit 2 vlan-id 102
set interfaces ae0 unit 3 vlan-id 103
set interfaces ae0 unit 4 vlan-id 104
```

3. Configure the interface set.

```
set interfaces interface-set ifset1-ae0 interface ae0 unit 0
set interfaces interface-set ifset1-ae0 interface ae0 unit 1
```

4. Configure class-of-service parameters for the interface sets.

```
set class-of-service interfaces interface-set ifset1-ae0 output-traffic-control-profile tcp
```





**NOTE:** You also need to configure the parameters of the traffic control profile. For more information, see the Related Documentation section on this page.

## 5. Configure scheduler mode.

```
set class-of-service interfaces ae0 member-link-scheduler scale
```

## RELATED DOCUMENTATION

[Configuring Traffic Control Profiles for Shared Scheduling and Shaping | 340](#)

[Hierarchical Schedulers on Aggregated Ethernet Interfaces Overview | 1209](#)

[Example: Configuring Shared Resources on Ethernet IQ2 Interfaces | 1049](#)

[Configuring Traffic Control Profiles for Shared Scheduling and Shaping | 340](#)

## Example: Configuring Scheduling Modes on Aggregated Interfaces

You can configure class-of-service parameters, such as queuing or shaping parameters on aggregated interfaces, in either link-protect or non-link-protect mode. You can configure these parameters for per-unit schedulers, hierarchical schedulers, or shaping at the physical and logical interface level. You can control the way these parameters are applied by configuring the aggregated interface to operate in scale or replicate mode.

You can apply these parameters on the following routers:

- MX Series router interfaces on EQ DPCs
- MX Series router interfaces on MICs or MPCs through Junos OS Release 10.2 (non-link-protect mode only)
- M120 or M320 routers
- T Series router interfaces on IQ2 PICs
- PTX Series Packet Transport Routers

You can configure the applied parameters for aggregated interfaces operating in non-link-protected mode. In link-protected mode, only one link in the bundle is active at a time (the other link is a backup link) so schedulers cannot be scaled or replicated. In non-link-protected mode, all the links in the bundle



are active and send traffic; however, there is no backup link. If a link fails or is added to the bundle in non-link-protected mode, the links' traffic is redistributed among the active links.

To set the scheduling mode for aggregated interfaces, include the `scale` or `replicate` option of the `member-link-scheduler` statement at the `[edit class-of-service interfaces ae $n$ ]` hierarchy level, where  $n$  is the configured number of the interface:

```
[edit class-of-service interfaces ae $n$ ]
member-link-scheduler (replicate | scale);
```

By default, if you do not include the `member-link-scheduler` statement, scheduler parameters are applied to the member links in the `scale` mode (also called “equal division mode”).

The aggregated Ethernet interfaces are otherwise configured as usual. For more information on configuring aggregated Ethernet interfaces, see the [Junos OS Network Interfaces Library for Routing Devices](#).

The following examples set `scale` mode on the `ae0` interface and `replicate` mode on the `ae1` interface.

```
[edit class-of-service]
interfaces ae0 {
    member-link-scheduler scale;
}

[edit class-of-service]
interfaces ae1 {
    member-link-scheduler replicate;
}
```



**NOTE:** The `member-link-scheduler` statement only appears for aggregated interfaces. You configure this statement for aggregated interfaces in non-link-protected mode. For more information about link protection modes, see the *Network Interfaces Configuration Guide*.

Aggregated interfaces support both hierarchical and per-unit schedulers.



**NOTE:** The `traffic-control-profiles` statement is not supported for PTX Series Packet Transport Routers.



When interface parameters are using the `scale` option of the `member-link-scheduler` statement, the following parameters under the `[edit class-of-service traffic-control-profiles traffic-control-profile-name]` configuration are scaled on egress when hierarchical schedulers are configured:

- `shaping-rate` (PIR)
- `guaranteed-rate` (CIR)
- `delay-buffer-rate`

When interface parameters are using the `scale` option of the `member-link-scheduler` statement, the following parameters under the `[edit class-of-service schedulers scheduler-name]` configuration are scaled on egress when per-unit schedulers are configured:

- `transmit-rate`
- `buffer-size`



**NOTE:** You cannot apply a hierarchical scheduler at the interface set level for an `ae` interface. (Interface sets cannot be configured under an `ae` interface.)

The following configuration parameters are not supported on `ae` interfaces in non-link-protection mode:

- Input scheduler maps
- Input traffic control profiles
- Input shaping rates

The following configuration conventions are also not supported:

- Scaling of the `input-traffic-control-profile-remaining` statement.
- The `scheduler-map-chassis` statement and the `derived` option for the `ae` interface. Chassis scheduler maps should be applied under the physical interfaces.
- Dynamic and demux interfaces are not supported as part of the `ae` bundle.

Depending on whether the `scale` or `replicate` option is configured, the `member-link-scheduler` statement operates in either scaled mode (also called “equal division mode”) or replicated mode, respectively.

In scaled mode, a VLAN can have multiple flows that can be sent over multiple member links of the `ae` interface. Likewise, a member link can receive traffic from any VLAN in the `ae` bundle. In scaled mode, the physical interface bandwidth is divided equally among all member links of the `ae` bundle.

In scaled mode, the following scheduler parameter values are divided equally among the member links:



- When the parameters are configured using traffic control profiles, then the parameters scaled are the shaping rate, guaranteed rate, and delay buffer rate.
- When the parameters are configured using scheduler maps, then the parameters scaled are the transmit rate and buffer size. Shaping rate is also scaled if you configure it in bits per second (bps). Shaping rate is not scaled if you configure it as a percentage of the available interface bandwidth.

For example, consider an ae bundle between routers R1 and R2 consisting of three links. These are ge-0/0/1, ge-0/0/2 and ge-0/0/3 (ae0) on R1; and ge-1/0/0, ge-1/0/1, and ge-1/0/2 (ae2) on R2. Two logical interfaces (units) are also configured on the ae0 bundle on R1: ae0.0 and ae0.1.

On ae0, traffic control profiles on R1 are configured as follows:

- ae0 (the physical interface level) has a PIR of 450 Mbps.
- ae0.0 (VLAN 100 at the logical interface level) has a PIR of 150 Mbps and a CIR of 90 Mbps.
- ae0.1 (VLAN 200 at the logical interface level) has a PIR of 90 Mbps and a CIR of 60 Mbps.

In scaled mode, the ae0 PIR is first divided among the member physical interfaces. Because there are three members, each receives  $450 / 3 = 150$  Mbps as a derived value. So the scaled PIR for the members interfaces is 150 Mbps each.

However, there are also two logical interfaces (ae0.0 and ae0.1) and VLANs (100 and 200) on ae0. Traffic can leave on any of the three physical interfaces (ge-0/0/1, ge-0/0/2, or ge-0/0/3) in the bundle. Therefore, two derived logical interfaces are added to the member links to represent the two VLANs.

There are now six logical interfaces on the physical interfaces of the links making up the ae bundle, one set for VLAN 100 and the other for VLAN 200:

- ge-0/0/1.0 and ge-0/0/1.1
- ge-0/0/2.0 and ge-0/0/2.1
- ge-0/0/3.0 and ge-0/0/3.1

The traffic control profile parameters configured on ae0.0 are divided across all the underlying logical interfaces (the unit 0s). In the same way, the traffic control profile parameters configured on ae0.1 are divided across all the underlying logical interfaces (the unit 1s).

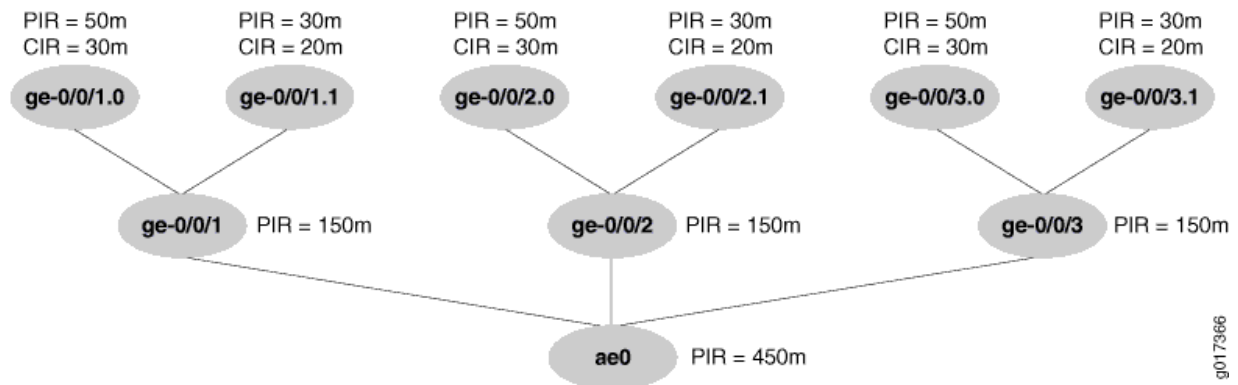
Therefore, the derived values of the scaled parameters on the interfaces are:

- For ge-0/0/1.0 and ge-0/0/2.0 and ge-0/0/3.0, each CIR =  $90 / 3 = 30$  Mbps, and each PIR =  $150 / 3 = 50$  Mbps.
- For ge-0/0/1.1 and ge-0/0/2.1 and ge-0/0/3.1, each CIR =  $60 / 3 = 20$  Mbps, and each PIR =  $90 / 3 = 30$  Mbps.



The scaled values are shown in [Figure 72 on page 1215](#).

**Figure 72: Scaled Mode for Aggregated Ethernet Interfaces**



In scaled mode, when a new member link is added to the bundle, or an existing member link is either removed or fails, then the scaling factor (based on the number of active links) is recomputed and the new scheduler or traffic control profile parameters are reassigned. Only the PIR, CIR, and buffer parameters are recomputed: all other parameters are simply copied at each level.



**NOTE:** In `show class-of-service scheduler-map` commands, values derived in scaled mode instead of explicitly configured are flagged with `&sf*n` suffix, where *n* indicates the value of the scaling factor.

The following sample shows the output for the scheduler map named `smap-all-abs` with and without a scaling factor:

```
user@host> show class-of-service scheduler-map
Scheduler map: smap-all-abs, Index: 65452

Scheduler: q0_sch_abs, Forwarding class: be, Index: 6775
Transmit rate: 40000000 bps, Rate Limit: none, Buffer size: remainder,
Priority: low
  Excess Priority: unspecified
  Drop profiles:
    Loss priority  Protocol  Index  Name
    Low           any      1      <default-drop-profile>
    Medium low    any      1      <default-drop-profile>
    Medium high   any      1      <default-drop-profile>
    High          any      1      <default-drop-profile>
```



```

user@host> show class-of-service scheduler-map
Scheduler map: smap-all-abs, Index: 65452

Scheduler: q0_sch_abs&***sf**3, Forwarding class: be, Index: 2128
Transmit rate: 13333333 bps, Rate Limit: none, Buffer size: remainder,
Priority: low
  Excess Priority: unspecified
  Drop profiles:
    Loss priority  Protocol  Index  Name
    Low           any       1      <default-drop-profile>
    Medium low    any       1      <default-drop-profile>
    Medium high   any       1      <default-drop-profile>
    High          any       1      <default-drop-profile>

```



**NOTE:** There can be multiple scheduler maps created with different scaling factors, depending on when the child interfaces come up. For example, if there are only two active children on a parent interface, a new scheduler map with a scaling factor of 2 is created. The scheduler map name is `smap-all-abs&***sf**2`.

In replicated mode, in contrast to scaled mode, the configured scheduler parameters are simply replicated, not divided, among all member links of the ae bundle.

In replicated mode, the following scheduler parameter values are replicated among the member links and logical interfaces:

- When the parameters are configured using traffic control profiles, then the parameters replicated are the shaping rate, guaranteed rate, and delay buffer rate.
- When the parameters are configured using scheduler maps, then the parameters replicated are the transmit rate and buffer size.

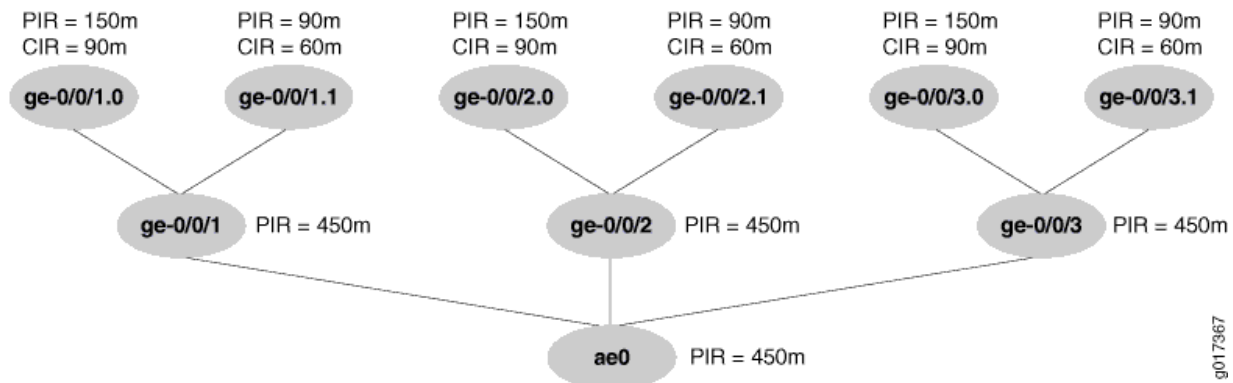
If the scheduler parameters in the example configuration between routers R1 and R2 are applied with the `member-link-scheduler replicate` statement and option, the following parameters are applied:

- The ae0 PIR is copied among the member physical interfaces. Each receives 450 Mbps as a PIR.
- For each logical interface unit .0, the configured PIR and CIR for ae0.0 is replicated (copied). Each logical interface unit .0 receives a PIR of 150 Mbps and a CIR of 90 Mbps.
- For each logical interface unit .1, the configured PIR and CIR for ae0.1 is replicated (copied). Each logical interface unit .1 receives a PIR of 90 Mbps and a CIR of 60 Mbps.

The replicated values are shown in [Figure 73 on page 1217](#).



Figure 73: Replicated Mode for Aggregated Ethernet Interfaces



In replicated mode, when a new member link is added to the bundle, or an existing member link is either removed or fails, the values are either copied or deleted from the required levels.

## RELATED DOCUMENTATION

[How Schedulers Define Output Queue Properties | 332](#)

[Default Schedulers Overview | 336](#)

[Configuring Schedulers | 338](#)

## Enabling VLAN Shaping and Scheduling on Aggregated Interfaces

Virtual LAN (VLAN) shaping (per-unit scheduling) is supported on aggregated Ethernet interfaces when link protection is enabled on the aggregated Ethernet interface. When VLAN shaping is configured on aggregate Ethernet interfaces with link protection enabled, the shaping is applied to the active child link.

To configure link protection on aggregated Ethernet interfaces, include the `link-protection` statement at the `[edit interfaces aex aggregated-ether-options]` hierarchy level.

Traffic passes only through the designated primary link. This includes transit traffic and locally generated traffic on the router. When the primary link fails, traffic is routed through the backup link. You also can reverse traffic, from the designated backup link to the designated primary link. To revert back to sending traffic to the primary designated link when traffic is passing through the designated backup link, use the `revert` command. For example, `request interfaces revert ae0`.

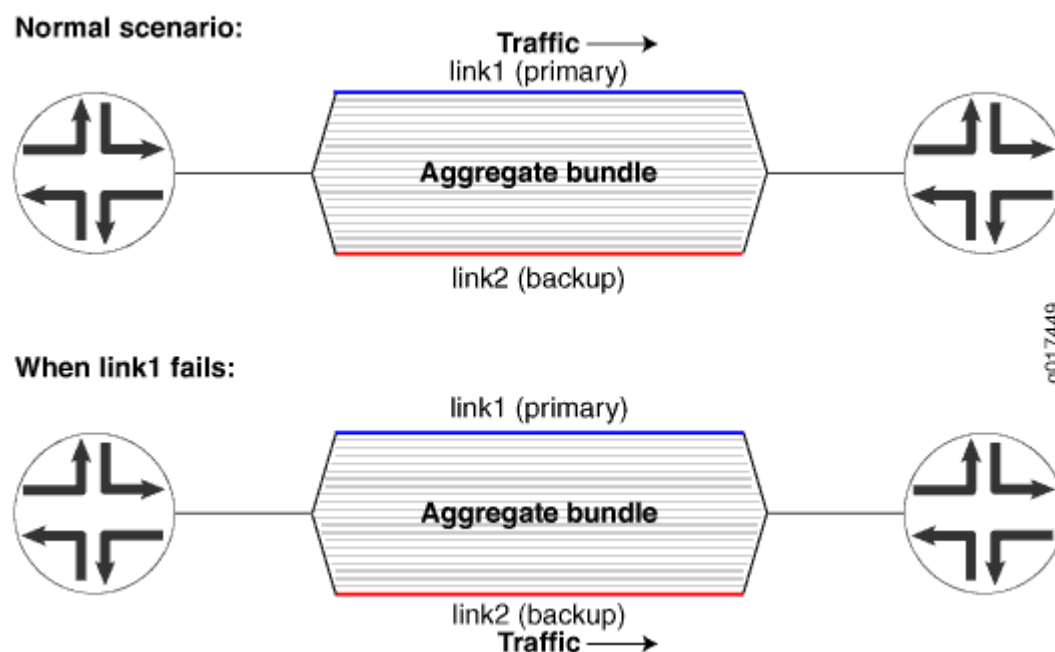
To configure a primary and a backup link, include the `primary` and `backup` statements at the `[edit interfaces ge-fpc/pic/port gigether-options 802.3ad aex]` hierarchy level or the `[edit interfaces xe-fpc/pic/port fastether-options 802.3ad aex]` hierarchy level.



To disable link protection, delete the link-protection statement at [edit interfaces aex aggregated-ether-options link-protection] hierarchy level. To display the active, primary, and backup link for an aggregated Ethernet interface, use the operational mode command `show interfaces redundancy aex`.

Figure 74 on page 1218 shows how the flow of traffic changes from primary to backup when the primary link in an aggregate bundle fails.

**Figure 74: Aggregated Ethernet Primary and Backup Links**



This example configures two Gigabit Ethernet interfaces (primary and backup) as an aggregated Ethernet bundle (ae0) and enables link protection so that a shaping rate can be applied to VLANs on the aggregated Ethernet interface.

```
[edit class-of-service]
interface ae0 {
  shaping-rate 300m;
}
[edit interfaces]
ge-1/0/0 {
  together-options {
    802.3ad ae0 primary;
  }
}
ge-1/0/1 {
```



```

    gigether-options {
        802.3ad ae0 backup;
    }
}
ae0 {
    aggregated-ether-options {
        lacp {
            periodic slow;
        }
        link-protection {
            enable;
        }
    }
}

```

## RELATED DOCUMENTATION

[Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs | 391](#)

[Example: Applying Scheduling and Shaping to VLANs | 407](#)

## Class of Service on demux Interfaces

### SUMMARY

This topic describes class of service (CoS) functionality and configuration on demux interfaces. demux interfaces support basic CoS functionality including behavior aggregate (BA) classifiers, rewrite rules, fixed classification, and port-level scheduling with targeted distribution.

### IN THIS SECTION

- [Overview of CoS on demux Interfaces | 1220](#)
- [Configuring CoS on demux Interfaces | 1220](#)

Demultiplexing (demux) interfaces are logical interfaces that share a common, underlying interface. You can create logical subscriber interfaces using static or dynamic demultiplexing interfaces. In addition, you can use IP demultiplexing interfaces or VLAN demultiplexing interfaces when creating logical subscriber interfaces.



## Overview of CoS on demux Interfaces

demux interfaces support basic CoS functionality, including:

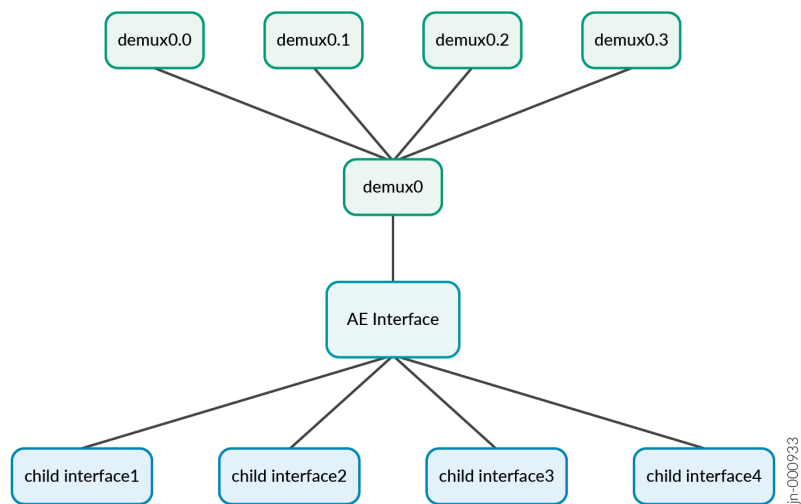
- Classifiers including INET precedence, DSCP, IEEE 802.1p, and IEEE 802.1ad
- Rewrite rules including INET precedence, DSCP, IEEE 802.1p, and IEEE 802.1ad
- Fixed classification



**NOTE:** demux interfaces do not support MPLS EXP classifiers or rewrite rules.

Figure 75 on page 1220 shows the demux interface (demux0) with an underlying aggregated Ethernet (AE) interface. You can assign classifiers and rewrite rules to the logical interfaces (demux0.x) of the demux interface. You can then configure schedulers, scheduler maps, traffic control profiles, and so on, for the underlying AE interface.

**Figure 75: demux interface with an underlying aggregated Ethernet interface**



## Configuring CoS on demux Interfaces

This section provides a simple example of how to configure a classifier and a rewrite rule for demux logical interfaces as well as a traffic control profile for the underlying AE interface.



1. Configure a classifier and apply it to a demux logical interface.

```
[edit class-of-service]
user@host# set classifiers ieee-802.1ad test forwarding-class expedited-forwarding loss-
priority low code-points 0000
user@host# set interfaces demux0 unit 0 classifiers ieee-802.1ad test
```

2. Configure a rewrite rule and apply it to a demux logical interface.

```
[edit class-of-service]
user@host# set rewrite-rules ieee-802.1ad test forwarding-class expedited-forwarding loss-
priority low code-point 1011
user@host# set interfaces demux0 unit 1 rewrite-rules ieee-802.1ad test
```

3. Configure a traffic control profile and apply it to the underlying AE interface.

```
[edit class-of-service]
user@host# set schedulers s1 shaping-rate 500m
user@host# set scheduler-maps testmap forwarding-class expedited-forwarding scheduler s1
user@host# set traffic-control-profiles tcp1 scheduler-map testmap
user@host# set interfaces ae0 output-traffic-control-profile tcp1
user@host# set interfaces ae0 member-link-scheduler replicate
```

4. Commit and show the configuration.

```
[edit class-of-service]
user@host# commit
user@host# show
classifiers {
  ieee-802.1ad test {
    forwarding-class expedited-forwarding {
      loss-priority low code-points 0000;
    }
  }
}
traffic-control-profiles {
  tcp1 {
    scheduler-map testmap;
  }
}
```



```

interfaces {
    ae0 {
        output-traffic-control-profile tcp1;
        member-link-scheduler replicate;
    }
    demux0 {
        unit 0 {
            classifiers {
                ieee-802.1ad test;
            }
        }
        unit 1 {
            rewrite-rules {
                ieee-802.1ad test;
            }
        }
    }
}
rewrite-rules {
    ieee-802.1ad test {
        forwarding-class expedited-forwarding {
            loss-priority low code-point 1011;
        }
    }
}
scheduler-maps {
    testmap {
        forwarding-class expedited-forwarding scheduler s1;
    }
}
schedulers {
    s1 {
        shaping-rate 500m;
    }
}

```

In the very simple example above, IEEE802.1ad traffic arriving on demux0.0 with a code point of 0000 is assigned a forwarding class of expedited-forwarding and a loss priority of low. Through the traffic control profile, this traffic has up to 500 Mb of bandwidth as it traverses the router. Upon exiting the router through demux0.1, the traffic's code point is rewritten to 1011.



## Example: Configuring Per-Unit Schedulers for Channelized Interfaces

You can configure per-unit scheduling on T1 and DS0 physical interfaces configured on channelized DS3 and STM1 IQ PICs. To enable per-unit scheduling, configure the `per-unit-scheduler` statements at the `[edit interfaces interface-name]` hierarchy level.

When per-unit scheduling is enabled on the channelized PICs, you can associate a scheduler map with the physical interface. For more information about configuring scheduler maps, see ["Configuring Scheduler Maps" on page 338](#).



**NOTE:** If you configure the `per-unit-scheduler` statement on the physical interface of a 4-port channelized OC-12 IQ PIC and configure 975 logical interfaces or data link connection identifiers (DLCIs), some of the logical interfaces or DLCIs will drop all packets intermittently.

The following example configures per-unit scheduling on a channelized DS3 PIC and an STM1 IQ PIC.

```
[edit interfaces]
ct3-5/3/1 {
    partition 1 interface-type t1;
}
t1-5/3/1:1 {
    per-unit-scheduler; # This enables per-unit scheduling
    encapsulation frame-relay;
    unit 0 {
        dlci 1;
        family inet {
            address 10.0.0.2/32;
        }
    }
}
ct3-5/3/0 {
    partition 1 interface-type ct1;
}
ct1-5/3/0:1 {
    partition 1 timeslots 1 interface-type ds;
}
ds-5/3/0:1:1 {
    per-unit-scheduler; # This enables per-unit scheduling
    encapsulation frame-relay;
    unit 0 {
```



```

        dlci 1;
        family inet {
            address 10.0.0.1/32;
        }
    }
}
cau4-3/0/0 {
    partition 1 interface-type ce1;
}
cstm1-3/0/0 {
    no-partition 1 interface-type cau4;
}
ce1-3/0/0:1 {
    partition 1 timeslots 1 interface-type ds;
}
ds-3/0/0:1:1 {
    per-unit-scheduler; # This enables per-unit scheduling
    encapsulation frame-relay;
    unit 0 {
        dlci 1;
        family inet {
            address 10.1.1.1/32;
        }
    }
}

[edit class-of-service]
classifiers {
    dscp all-traffic-dscp {
        forwarding-class assured-forwarding {
            loss-priority low code-points 001010;
        }
        forwarding-class expedited-forwarding {
            loss-priority low code-points 101110;
        }
        forwarding-class best-effort {
            loss-priority low code-points 101010;
        }
        forwarding-class network-control {
            loss-priority low code-points 000110;
        }
    }
}

```



```

forwarding-classes {
    queue 0 best-effort;
    queue 1 assured-forwarding;
    queue 2 expedited-forwarding;
    queue 3 network-control;
}
interfaces {
    ds-3/0/0:1:1 {
        unit 0 {
            scheduler-map schedule-mlppp;
        }
    }
    ds-5/3/0:1:1 {
        unit 0 {
            scheduler-map schedule-mlppp;
        }
    }
    t1-5/3/1:1 {
        unit 0 {
            scheduler-map schedule-mlppp;
        }
    }
}
scheduler-maps {
    schedule-mlppp {
        forwarding-class expedited-forwarding scheduler expedited-forwarding;
        forwarding-class assured-forwarding scheduler assured-forwarding;
        forwarding-class best-effort scheduler best-effort;
        forwarding-class network-control scheduler network-control;
    }
}
schedulers {
    best-effort {
        transmit-rate percent 2;
        buffer-size percent 5;
        priority low;
    }
    assured-forwarding {
        transmit-rate percent 7;
        buffer-size percent 30;
        priority low;
    }
    expedited-forwarding {

```



```

        transmit-rate percent 90 exact;
        buffer-size percent 60;
        priority high;
    }
    network-control {
        transmit-rate percent 1;
        buffer-size percent 5;
        priority strict-high;
    }
}

```

## RELATED DOCUMENTATION

[Applying Scheduler Maps and Shaping Rate to DLCIs and VLANs | 391](#)

[Example: Applying Scheduler Maps and Shaping Rate to DLCIs | 401](#)

## Applying Layer 2 Policers to Gigabit Ethernet Interfaces

### IN THIS SECTION

- [Examples: Applying Layer 2 Policers to a Gigabit Ethernet Interface | 1227](#)

To rate-limit traffic by applying a policer to a Gigabit Ethernet interface (or a 10-Gigabit Ethernet interface [xe-fpc/pic/port]), include the `layer2-policer` statement with the direction, type, and name of the policer:

```

[edit interfaces ge-fpc/pic/port unit 0]
layer2-policer {
    input-policer policer-name;
    input-three-color policer-name;
    output-policer policer-name;
    output-three-color policer-name;
}

```



The direction (input or output) and type (policer or three-color) are combined into one statement and the policer named must be properly configured.

One input or output policer of either type can be configured on the interface.

### Examples: Applying Layer 2 Policers to a Gigabit Ethernet Interface

Apply color-blind and color-aware two-rate TCM policers as input and output policers to a Gigabit Ethernet interface:

```
ge-1/0/0 {
  unit 0
  layer2-policer {
    input-three-color trTCM1-cb;    # Apply the trTCM1-color-blind policer.
    output-three-color trTCM1-ca;    # Apply the trTCM1-color-aware policer.
  }
}
```

Apply two-level and color-blind single-rate TCM policers as input and output policers to a Gigabit Ethernet interface:

```
ge-1/0/0 {
  unit 1
  layer2-policer {
    input-policer two-color-policer;    # Apply a two-color policer.
    output-three-color srTCM2-cb;    # Apply the srTCM1-color-blind policer.
  }
}
```

Apply a color-aware single-rate TCM policer as output policer on a Gigabit Ethernet interface:

```
ge-1/0/0 {
  unit 2
  layer2-policer {
    output-three-color srTCM3-ca {    # Apply the srTCM3-color-aware policer.
    }
  }
}
```



## RELATED DOCUMENTATION

*Controlling Network Access Using Traffic Policing Overview*

---

[Overview of Tricolor Marking Architecture](#) | 229



# 5

PART

## Configuration Statements and Operational Commands

---

[Junos CLI Reference Overview](#) | 1230

---



# Junos CLI Reference Overview

We've consolidated all Junos CLI commands and configuration statements in one place. Learn about the syntax and options that make up the statements and commands and understand the contexts in which you'll use these CLI elements in your network configurations and operations.

- [Junos CLI Reference](#)

Click the links to access Junos OS and Junos OS Evolved configuration statement and command summary topics.

- [Configuration Statements](#)
- [Operational Commands](#)