Modular QoS Configuration Guide for Cisco NCS 540 Series Routers, Cisco IOS XR Release 24.1.x, 24.2.x, 24.3.x, 24.4.x

First Published: 2024-03-14

Last Modified: 2024-12-16

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: https://www.cisco.com/c/en/us/about/legal/trademarks.html. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2024 Cisco Systems, Inc. All rights reserved.

CONTENTS

CHAPTER 1 **YANG Data Models for QoS Features** 1

Using YANG Data Models 1

CHAPTER 2 **Key Hardware Resources for QoS** 3

View Packet Processing and Traffic Management Resources 5

CHAPTER 3 **Configuring Modular QoS Service Packet Classification** 11

Packet Classification Overview 11

Traffic Class Elements 12

Default Traffic Class 14

Create a Traffic Class 14

Traffic Policy Elements 16

Create a Traffic Policy 17

Scaling of Unique Ingress Policy Maps 19

Limitations and Restrictions 20

Attach a Traffic Policy to an Interface 20

Packet Marking 22

Handling QoS for Locally Originated Packets 25

QoS packet marking across network layers 29

Protecting Locally Originated BFD Packets 35

Prioritization of IS-IS and ARP Packets to Manage Transit Traffic 36

QoS Re-marking of IP Packets in Egress Direction 38

QoS Re-marking of Ethernet Packets in Egress Direction 39

QoS L2 Re-marking of Ethernet Packets in Egress Direction 39

QoS L2 Re-Marking of Ethernet Packets on L3 Flows in Egress Direction 40

Layer 2 Ingress QoS Matching for IPv4 and IPv6 Destination Addresses 43

Overview 43
Guidelines and Limitations 43
Configure Layer 2 Ingress QoS Matching for IPv4 and IPv6 Destination Addresses 44
Bundle Traffic Policies 46
Shared Policy Instance 47
Restrictions and Guidelines 48
Attaching a Shared Policy Instance to Multiple Subinterfaces 48
Ingress Short-Pipe 50
Restrictions and Other Important Points 51
Configure Ingress Short-Pipe 51
Selective Egress Policy-Based Queue Mapping 53
Restrictions and Other Important Points 54
Configure Selective Egress Policy-Based Queue Mapping 55
Configuring QoS Groups with an ACL 57
Configuring an ACL with Fragment Match 60
Restrictions and Guidelines 61
Configuring an ACL with Fragment Match 61
QoS Egress Marking and Queuing Using Dual Policy-Map 65
Restrictions Specific to NCS 540 Variants 67
In-Place Policy Modification 71
References for Modular QoS Service Packet Classification 72
Specification of the CoS for a Packet with IP Precedence 72
IP Precedence Bits Used to Classify Packets 73
IP Precedence Value Settings 73
IP Precedence Compared to IP DSCP Marking 73
Conditional Marking of MPLS Experimental bits for L3VPN Traffic 73
Conditional Marking of MPLS Experimental bits for L2VPN Traffic 74
QoS IP DSCP Preservation for SR-TE 76
Guidelines and Limitations for QoS IP DSCP Preservation for SR-TE 78
Configuration for QoS IP DSCP Preservation for SR-TE 79
Policy-map for conditional marking of incoming traffic 81
Policy-map for conditional marking of outgoing MPLS traffic 82
Conditional Marking of MPLS Experimental Bits for EVPN-VPWS Single-Homing Services 82
QPPB 85

Configuration Workflow 87
Configuring QPPB on an Interface 88
Egress Interface Configuration 89

CHAPTER 4 **Configuring Modular QoS Congestion Avoidance 91**
Modular QoS Congestion Avoidance 91
Tail Drop and the FIFO Queue 92
Configure Tail Drop 92
Random Early Detection and TCP 93
Configure Random Early Detection 94
Weighted Random Early Detection 96
Average Queue Size for WRED 96
Configure Weighted Random Early Detection 97
Explicit Congestion Notification (ECN) 99

CHAPTER 5 **Configuring Modular QoS Congestion Management 103**
Congestion Management Overview 103
Best practices for minimizing packet drops during sustained maximum line-rate traffic 104
Class-based Weighted Fair Queueing 104
Bandwidth Remaining 104
Configuring Bandwidth Remaining 104
Low-Latency Queuing with Strict Priority Queuing 107
Configuring Low Latency Queuing with Strict Priority queuing 107
Configuring Low Latency Queuing for Specific QoS Flows 110
Overhead Accounting 112
Traffic Shaping 114
Configure VOQ-Level Traffic Shaping 114
Burst Size for Port-Level Shaper 117
Burst Size for Port-Level Shaper: Why 118
Burst Size for Port-Level Shaper: Guidelines and Limitations 118
Configure Burst Size for Port-Level Shaper 119
Traffic Policing 123
Committed Bursts and Excess Bursts 124
Single-Rate Policer 124

Configure Traffic Policing (Single-Rate Two-Color) 125
Configure Traffic Policing (Single-Rate Three-Color) 127
Two-Rate Policer 128
Configure Traffic Policing (Two-Rate Three-Color) 129
Packets-Per-Second-Based Policer 131
Shared Policer 134
Policer Bucket Shared 134
Policer Bucket Referred 135
Shared Policer Statistics 135
Restrictions and Guidelines 135
Configuring Shared Policer 136
Prioritize Traffic during Oversubscription 138
Configuring Traffic Priorities on Tagged Interface 139
Configuring Traffic Priorities on Untagged Interface 140
Limitations 141
References for Modular QoS Congestion Management 141
Committed Bursts 141
Excess Bursts 141
Two-Rate Policer Details 142

CHAPTER 6 **Configuring Modular QoS on Link Bundles** 145
QoS on Link Bundles 145
Load Balancing 145
Configure QoS on Link Bundles 146

CHAPTER 7 **Configuring Hierarchical Modular QoS** 151
Overview of Hierarchical Modular QoS 151
Restrictions, Limitations, and Requirements for Configuring H-QoS 152
Configuring Hierarchical Queuing 153
Bandwidth Remaining Ratio at Parent Level 159
Conform Aware Hierarchical Policy Overview 162
Configuring Conform Aware Hierarchy Policy 163
Conform Aware Hierarchical Policy Restrictions 166

CHAPTER 8 **QoS for Bridge-Group Virtual Interfaces** 169

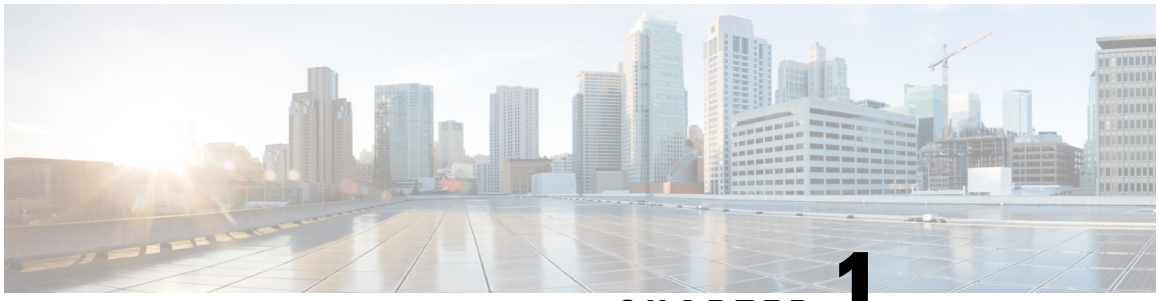
Information on Qos on BVI 169

Restrictions on BVI 169

Classification and Marking 170

Configuring QoS on BVI 171

Verifying QoS on BVI 172



CHAPTER 1

YANG Data Models for QoS Features

This chapter provides information about the YANG data models for QoS features.

- Using YANG Data Models, on page 1

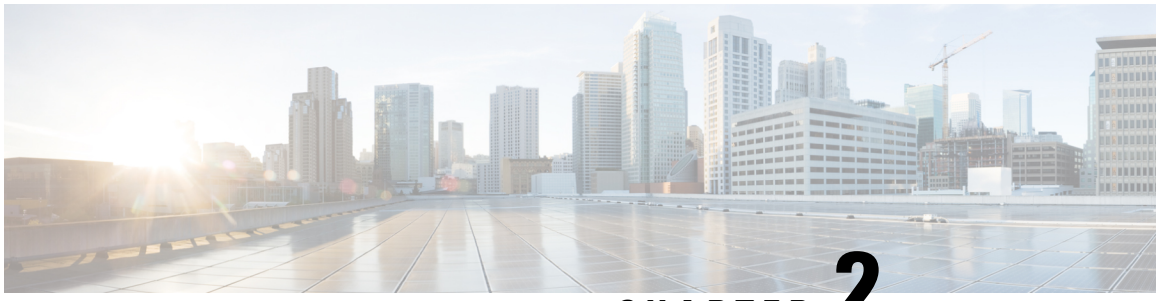
Using YANG Data Models

Cisco IOS XR supports a programmatic way of configuring and collecting operational data of a network device using YANG data models. Although configurations using CLIs are easier and human-readable, automating the configuration using model-driven programmability results in scalability.

The data models are available in the release image, and are also published in the Github repository. Navigate to the release folder of interest to view the list of supported data models and their definitions. Each data model defines a complete and cohesive model, or augments an existing data model with additional XPaths. To view a comprehensive list of the data models supported in a release, navigate to the **Available-Content.md** file in the repository.

You can also view the data model definitions using the YANG Data Models Navigator tool. This GUI-based and easy-to-use tool helps you explore the nuances of the data model and view the dependencies between various containers in the model. You can view the list of models supported across Cisco IOS XR releases and platforms, locate a specific model, view the containers and their respective lists, leaves, and leaf lists presented visually in a tree structure. This visual tree form helps you get insights into nodes that can help you automate your network.

To get started with using the data models, see the *Programmability Configuration Guide*.



CHAPTER 2

Key Hardware Resources for QoS

QoS implementations rely on various hardware resources to support efficient packet processing and traffic management.

See the table to know more about the key hardware resources that QoS utilizes.

Table 1: Key Hardware Resources that QoS Uses

Resource	Details
Switches and Routers	Switches and routers are fundamental networking devices that play a crucial role in QoS. They are responsible for forwarding traffic based on QoS policies and handling tasks such as packet classification, queuing, and scheduling.
Network Processors	Specialized hardware components designed to handle the processing and forwarding of network traffic. They typically offer high-speed packet processing capabilities, including packet classification, traffic shaping, and queuing tasks.
Traffic Shaper Devices	Hardware devices that regulate the outgoing traffic rate to meet specified bandwidth limits. They typically incorporate algorithms and buffers to shape the traffic flow and smooth out bursts or peaks in network utilization.
Application-Specific Integrated Circuits (ASICs)	ASICs are custom-designed integrated circuits tailored for specific networking tasks. They offer hardware acceleration for essential QoS functions such as packet classification, marking, queuing, and shaping, enabling high-performance and low-latency processing.
Line Cards	Components in networking devices such as switches and routers. They contain the necessary hardware resources, including ASICs, memory, and interfaces, to handle the packet processing and traffic management functions for specific ports or interfaces.

Resource	Details
Memory	Memory resources, including volatile (for example, RAM) and non-volatile (for example, flash memory), are essential for storing and managing QoS-related data, such as traffic classification rules, queuing parameters, and packet buffers.
Interfaces	Network interfaces, such as Ethernet ports or WAN interfaces, are hardware components that connect networking devices to the network. These interfaces often have QoS-related features, such as traffic shaping capabilities or support for differentiated services code points (DSCPs), enabling efficient packet processing and traffic management.

Regardless of whether you're setting up your network and plan to implement QoS or already managing your network and using QoS, it's essential to plan for these resources so you don't run out of them while, for example, deploying topologies for high-scale requirements. An essential part of this implementation is monitoring the availability of QoS resources.

- View Packet Processing and Traffic Management Resources, on page 5

View Packet Processing and Traffic Management Resources

Table 2: Feature History Table

Feature Name	Release Information	Feature Description
View Packet Processing and Traffic Management Resources	Release 7.11.1	You can now view the utilization of some packet processing and traffic management resources, such as policer banks and connectors. Insights into their consumption and availability help you prevent or mitigate an Out of Resource (OOR) situation, thus ensuring optimal QoS operations with minimal impact on network performance. The feature introduces the following changes: **CLI:** **show controllers npu resources qos** **YANG:** • ~~Cisco-IOS-XR-fretta-bcm-dpa-qos-resources-oper~~ • Cisco-IOS-XR-5500-qos-oper • ~~Cisco-IOS-XR-fretta-bcm-dpa-qos-ratepolice-resources-oper~~ • ~~Cisco-IOS-XR-fretta-bcm-dpa-qos-egpresources-oper~~ (see GitHub, YANG Data Models Navigator)

QoS utilizes hardware resources to classify, mark, and shape traffic. Given that these resources are limited, you require immediate access to information such as their consumption and availability, especially if you plan to implement high-scale setups. Without such information, you may run out of resources, and tracking down the impacted hardware resources could prove challenging.

From Release 7.11.1 onwards, you can view the utilization of some packet processing and traffic management resources by running the **show controllers npu resources qos** command. Packet processing and traffic management resources are crucial components of QoS implementations, helping ensure that network traffic is prioritized, managed, and controlled effectively. You can use this data to troubleshoot Out of Resource (OOR) situations by quickly identifying the exhausted resources or prevent OOR conditions by appropriately modifying their allocation.

Run the **show controllers npu resources qos** command to view the information listed in the table. To best use this information and plan the grouping of multiple resources for specific QoS tasks, you could initially add one policy and check how much of these resources QoS utilizes.

Table 3: Packet Processing and Traffic Management Resources and their Details

Resource	Type	More About This Resource	Important Because
Policer	Packet Processing	The number of policers that are allocated and that you can create. The total number of policer banks is inversely proportional to the class map size.	Helps you determine the ingress policy map scale based on available policer bank resources.
Egress QoS Map Information	Packet Processing	The resources used in creating traffic class and drop precedence (or discard class) maps for egress traffic.	Displays counters for L2 and L3 pools, helping you tune your policy map configurations at the ingress or egress based on the existing policy parameters and optimize the available pool usage.

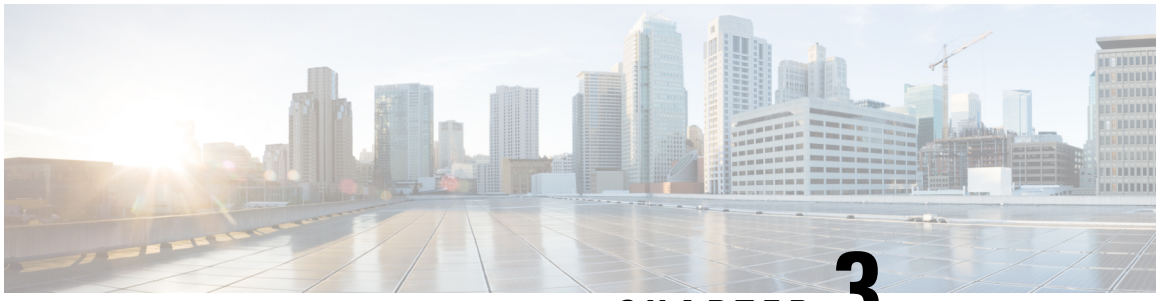
Resource	Type	More About This Resource	Important Because
Connectors	Traffic Management	The number of NPU connectors or Fabric Access Processors (FAPs) used.	The connector resource dictates the number of VOQs you can create. The egress policy map scale is bound to VOQ availability and, in turn, connector availability. It also tells you, among others: • the **Reserved** number of normal and low rate connector sets for internal VOQ creation; internal VOQs are used for punting and recycling operations. • the **Per Interface** counter which indicates the number of connectors used per interface or VOQ. **Note** By default, two connector regions per core are reserved for low connectors, which support lower traffic shaping rates.
Rate Profile	Traffic Management	The rate profile pool used. Interfaces with different line rates and policies with significantly differing shaper and queue lengths use rate profiles.	Helps you tune your egress policy parameters, such as shaper value and queue length, to optimize the available rate profile hardware resources.
Egress Queue (EGQ) Profile	Traffic Management	The number of egress queue maps used. They are consumed when you create egress policies with priority marking.	Helps you tune your egress policy parameters such as egress queue map to optimize the available hardware resources .

Resource	Type	More About This Resource	Important Because
VOQ	Traffic Management	The number of Virtual Output Queues (VOQs) used at a global level from a common pool.	Tells you the number of VOQs consumed (**Used**), the number of VOQs remaining (**Free**), and the **Total** number of VOQs available.

Verification

```
Router#show controllers npu resources qos all instance all location 0/0/CPU0

========================================================================
QoS TM Connectors Information For Location: 0/0/CPU0

System information for NPU 0:
Core 0:
                               Used(   %)       Free      Total   Reserved        Per
Interface Consumption
    Normal Connector Set:        384(  0%)      59008      59392       6144
             16
   Lowrate connector Set:          0(  0%)          0          0

Core 1:
                               Used(   %)       Free      Total   Reserved        Per
Interface Consumption
    Normal Connector Set:        384(  0%)      59008      59392       6144
             16
   Lowrate connector Set:          0(  0%)          0          0

========================================================================
QoS TM Rate Profile Information:
Note: - This is a global resource and shared across all LCs

                               Used(   %)       Free      Total     Reserved      Highest
 Used
                                  3(  5%)         56         59            5
  3

========================================================================
QoS TM EGQ Profile Information For Location: 0/0/CPU0

System information for NPU 0:
Core 0:
                               Used(   %)       Free      Total
                                  8(100%)          0          8

Core 1:
                               Used(   %)       Free      Total
                                  8(100%)          0          8

========================================================================
QoS PP Policer Banks Information For Location: 0/0/CPU0

System information for NPU 0:
Core 0:
                               Used(   %)       Free      Total         Class-map size
      Policer Banks info:         0(  0%)        218        218                   32
```

```
Core 1:
                                Used(   %)       Free      Total          Class-map size
      Policer Banks info:          0(  0%)        218        218                    32

========================================================================
QoS PP Egress QoS Map Information For Location: 0/0/CPU0

System information for NPU 0:      Used(   %)       Free      Total
         L2 Profile Pool:             0(  0%)         14         14
         L3 Profile Pool:             0(  0%)          3          3

========================================================================
QoS VOQ Information For Location: 0/0/CPU0

Note:- This is a global resources shared by all NPUs

                                Used(   %)       Free      Total
               VOQ info:        175(  1%)      11984      12159
```



CHAPTER 3

Configuring Modular QoS Service Packet Classification

This chapter covers these topics:

- Packet Classification Overview, on page 11
- Traffic Class Elements, on page 12
- Traffic Policy Elements, on page 16
- Ingress Short-Pipe, on page 50
- Selective Egress Policy-Based Queue Mapping, on page 53
- Configuring QoS Groups with an ACL, on page 57
- Configuring an ACL with Fragment Match, on page 60
- QoS Egress Marking and Queuing Using Dual Policy-Map, on page 65
- Restrictions Specific to NCS 540 Variants, on page 67
- In-Place Policy Modification, on page 71
- References for Modular QoS Service Packet Classification, on page 72
- Conditional Marking of MPLS Experimental bits for L2VPN Traffic, on page 74
- Conditional Marking of MPLS Experimental Bits for EVPN-VPWS Single-Homing Services, on page 82
- QPPB, on page 85

Packet Classification Overview

Packet classification involves categorizing a packet within a specific group (or class) and assigning it a traffic descriptor to make it accessible for QoS handling on the network. The traffic descriptor contains information about the forwarding treatment (quality of service) that the packet should receive. Using packet classification, you can partition network traffic into multiple priority levels or classes of service. The source agrees to adhere to the contracted terms and the network promises a quality of service. Traffic policers and traffic shapers use the traffic descriptor of a packet to ensure adherence to the contract.

Traffic policers and traffic shapers rely on packet classification features, such as IP precedence, to select packets (or traffic flows) traversing a router or interface for different types of QoS service. After you classify packets, you can use other QoS features to assign the appropriate traffic handling policies including congestion management, bandwidth allocation, and delay bounds for each traffic class.

The Modular Quality of Service (QoS) CLI (MQC) defines the traffic flows that must be classified, where each traffic flow is called a class of service, or class. Later, a traffic policy is created and applied to a class. All traffic not identified by defined classes fall into the category of a default class.

You can classify packets at the ingress on L3 subinterfaces for (CoS, DEI) for IPv4, IPv6, and MPLS flows. IPv6 packets are forwarded by paths that are different from those for IPv4. To enable classification of IPv6 packets based on (CoS, DEI) on L3 subinterfaces, run the hw-module profile qos ipv6 short-l2qos-enable command and reboot the line card for the command to take effect.

Guidelines and Limitations

- Ingress CoS marking is not supported on Cisco NCS 5500 Series routers; however, it is supported on Cisco NCS 5700 Series routers.

Note Ingress CoS marking is not supported on L3 interfaces because it can overwrite the Port Control Protocol (PCP) for traffic passing through the core network. To implement CoS marking, always configure it on egress L2 interfaces.

Traffic Class Elements

The purpose of a traffic class is to classify traffic on your router. Use the **class-map** command to define a traffic class.

A traffic class contains three major elements:

- A name
- A series of **match** commands - to specify various criteria for classifying packets.
- An instruction on how to evaluate these **match** commands (if more than one **match** command exists in the traffic class)

Packets are checked to determine whether they match the criteria that are specified in the **match** commands. If a packet matches the specified criteria, that packet is considered a member of the class and is forwarded according to the QoS specifications set in the traffic policy. Packets that fail to meet any of the matching criteria are classified as members of the default traffic class.

This table shows the details of match types that are supported on the router.

Match Type Supported	Min, Max	Max Entries	Support for Match NOT	Support for Ranges	Direction Supported on Interfaces
IPv4 DSCP IPv6 DSCP DSCP	(0,63)	64	Yes	Yes	Ingress
IP encapsulation	(0,63)	64	Yes	Yes	Ingress

Match Type Supported	Min, Max	Max Entries	Support for Match NOT	Support for Ranges	Direction Supported on Interfaces
IPv4 Precedence IPv6 Precedence Precedence	(0,7)	8	Yes	No	Ingress
MPLS Experimental Topmost	(0,7)	8	Yes	No	Ingress
Access-group	Not applicable	8	No	Not applicable	Ingress
QoS-group	(1,7) (1,511) for peering profile	7	No	No	• Egress • Ingress for QoS Policy Propagation Using Border Gateway Protocol (QPPB) • Ingress for peering profile
Traffic-class	(1,7)	7	No	No	• Egress
Protocol	(0,255)	1	Yes	Not applicable	Ingress

Note Egress queue statistics are displayed only for those classes which have a corresponding match criteria in the egress. Therefore, if you have a **set traffic-class** *x* configured in the ingress, you must have a corresponding **match traffic-class** *x* in the egress, in order to see the statistics in the egress side.

Note A maximum value of up to 64 unique queues is supported. Each unique queue-limit consumes one rate profile in the Traffic manager. Out of 64 unique queues, few are reserved for default configs and the remaining are usable.

Depending on the interface speeds, default configurations consume some of the rate profiles. The remaining rate profiles can be exhausted in the following scenarios:

- Different shape rates without configuring queue limits could exhaust the rate profiles as 10ms of guaranteed service rate converts to a different value in bytes based on the shape rate.
- Configuring queue limits in units of time could exhaust the rate profiles. For example, 20 ms of 50 Mbps and 20 ms of 100 Mbps are two different values in bytes.

Tip You can avoid exhausting rate profiles by configuring queue limits in absolute units (such as bytes, kilobytes, or megabytes) for class maps and sharing these limits with the policy maps.

Default Traffic Class

Unclassified traffic (traffic that does not meet the match criteria specified in the traffic classes) is treated as belonging to the default traffic class.

If the user does not configure a default class, packets are still treated as members of the default class. However, by default, the default class has no enabled features. Therefore, packets belonging to a default class with no configured features have no QoS functionality. These packets are then placed into a first in, first out (FIFO) queue and forwarded at a rate determined by the available underlying link bandwidth. This FIFO queue is managed by a congestion avoidance technique called tail drop.

For egress classification, match on **traffic-class** (1-7) is supported. Match **traffic-class 0** cannot be configured. The class-default in the egress policy maps to **traffic-class 0**.

This example shows how to configure a traffic policy for the default class:

```
configure
 policy-map ingress_policy1
 class class-default
  police rate percent 30
 !
```

Create a Traffic Class

To create a traffic class containing match criteria, use the **class-map** command to specify the traffic class name, and then use the **match** commands in class-map configuration mode, as needed.

Guidelines

- Match criteria evaluation order: You can provide multiple values for a match type in a single line of configuration; that is, if the first value does not meet the match criteria, then the next value indicated in the match statement is considered for classification.
- Excluding values with **not** keyword: Use the **not** keyword with the **match** command to perform a match based on the values of a field that are not specified.
- Match commands are optional, but one required: All **match** commands specified in this configuration task are optional, but you must configure at least one match criterion for a class.
- **match-any** versus **match-all** behavior: If you specify **match-any**, one of the match criteria must be met for traffic entering the traffic class to be classified as part of the traffic class. This is the default. If you specify **match-all**, the traffic must match all the match criteria.
- Packet length match support: From Release 7.11.1 onwards, for the **match access-group** command, QoS classification based on the packet length field in the IPv4 and IPv6 headers is supported. Prior to this, support was not available for packet length and TTL (time to live) fields.

- Deny or empty ACLs not supported in class maps: For the **match access-group** command, when an ACL list is used within a class-map, the deny action of the ACL is ignored and the traffic is classified based on the specified ACL match parameters.

  An empty ACL (contains no rules, only remarks), when used within a class-map permits all traffic by default, and the implicit deny condition doesn't work with an empty ACL. The corresponding **class-map** matches all traffic not yet matched by the preceding traffic classes.

- Egress matching restrictions: The **traffic-class** and **discard-class** are supported only in egress direction, and these are the only match criteria supported in egress direction.

- Default class matching behavior: The egress default class implicitly matches **qos-group** 0 for marking policy and **traffic-class** 0 for queuing policy.

- Multicast and unicast egress ratio: Multicast takes a system path that is different than unicast on router, and they meet later on the egress in a multicast-to-unicast ratio of 20:80 on a per interface basis. This ratio is maintained on the same priority level as that of the traffic.

- MPLS EXP value for multicast in conditional marking: When conditional marking policy map is applied, the MPLS EXP value is set to 0 for multicast traffic.

- Topmost and inner MPLS labels marked for multicast: When an ingress policy-map is applied to mark an MPLS EXP topmost label, the MPLS EXP topmost and inner MPLS labels are marked for multicast traffic.

- Multicast QoS prioritization for egress: Egress QoS for multicast traffic treats traffic classes 0-5 as low-priority and traffic classes 6-7 as high priority. Currently, this is not user-configurable.

- Egress shaping not applied to high-priority multicast: Egress shaping does not take effect for multicast traffic in the high priority (HP) traffic classes. It only applies to unicast traffic.

- No egress mapping for ingress classes: If you set a traffic class at the ingress policy and do not have a matching class at egress for the corresponding traffic class value, then the traffic at ingress with this class will not be accounted for in the default class at the egress policy map.

- Only Class 0 maps to default class: Only traffic class 0 falls in the default class. A non-zero traffic class assigned on ingress but with no assigned egress queue, falls neither in the default class nor any other class.

- EDM timeout during line card reload: EDM timeouts for **show policy-map** CLI commands are expected during line card reload with scale configuration.

- Egress rate calculation variation on routers with Cisco NC57 line cards:

  - On Cisco NCS 5700 modular routers (in both compatibility and native modes), the **show policy-map interface** command may display noticeable variations for the egress rates. The ingress rates are displayed accurately.

  - This behavior is specific to routers with external TCAM and more than one NPU.

  - This behavior might not be noticeable when viewing the statistics for the first time but can become evident in subsequent checks. No action is required.

Configuration Example

You have to accomplish the following to complete the traffic class configuration:

1. Creating a class map
2. Specifying the match criteria for classifying the packet as a member of that particular class

   (For a list of supported match types, see Traffic Class Elements, on page 12.)

```
Router# configure
Router(config)# class-map match-any qos-1
Router(config-cmap)# match qos-group 1
Router(config-cmap)# end-class-map
Router(config-cmap)# commit
```

Use this command to verify the class-map configuration:

```
Router#show class-map qos-1
1) ClassMap: qos-1    Type: qos
    Referenced by 2 Policymaps
```

Also see, Running Configuration, on page 20.

Also see, Verification, on page 21.

Related Topics

- Traffic Class Elements, on page 12
- Traffic Policy Elements, on page 16

Associated Commands

Traffic Policy Elements

A traffic policy contains three elements:

- Name
- Traffic class
- QoS policies

After choosing the traffic class that is used to classify traffic to the traffic policy, the user can enter the QoS features to be applied to the classified traffic.

The MQC does not necessarily require that the users associate only one traffic class to one traffic policy.

The order in which classes are configured in a policy map is important. The match rules of the classes are programmed into the TCAM in the order in which the classes are specified in a policy map. Therefore, if a packet can possibly match multiple classes, only the first matching class is returned and the corresponding policy is applied.

The router supports 32 classes per policy-map in the ingress direction and 8 classes per policy-map in the egress direction.

This table shows the supported class-actions on the router.

Supported Action Types	Direction supported on Interfaces
minimum-bandwidth	egress
bandwidth-remaining*	egress
mark	(See Packet Marking, on page 22)
police	ingress
priority	egress (level 1 to level 7)
queue-limit	egress
shape	egress
wred	egress

*Bandwidth and Bandwidth remaining configurations are not supported simultaneously within the same policy-map in H-QoS mode.

WRED supports **default** and **discard-class** options; the only values to be passed to the discard-class being 0 and 1.

Create a Traffic Policy

The purpose of a traffic policy is to configure the QoS features that should be associated with the traffic that has been classified in a user-specified traffic class or classes.

Note Starting from Cisco IOS XR Release 7.6.1, the ingress QoS policy you configure, also applies to all the transit Ethernet Connectivity Fault Management (CFM) packets. With QoS features such as marking and policing now applied on the transit CFM packets, you ensure that these packets don't consume excess bandwidth and other QoS resources in the network.

To configure a traffic class, see Create a Traffic Class, on page 14.

After you define a traffic policy with the **policy-map** command, you can attach it to one, or more interfaces to specify the traffic policy for those interfaces by using the **service-policy** command in interface configuration mode. With dual policy support, you can have two traffic policies, one marking and one queuing attached at the output. See, Attach a Traffic Policy to an Interface, on page 20.

Configuration Example

You have to accomplish the following to complete the traffic policy configuration:

1. Creating a policy map that can be attached to one or more interfaces to specify a service policy
2. Associating the traffic class with the traffic policy
3. Specifying the class-action(s) (see Traffic Policy Elements, on page 16)

```
Router# configure
```

```
Router(config)# policy-map  test-shape-1
Router(config-pmap)# class qos-1

/* Configure class-action ('shape' in this example).
Repeat as required, to specify other class-actions */
Router(config-pmap-c)# shape average percent 40
Router(config-pmap-c)# exit

/* Repeat class configuration as required, to specify other classes */

Router(config-pmap)# end-policy-map
Router(config)# commit
```

See, Running Configuration, on page 20.

See, Verification, on page 21.

Related Topics

- Traffic Policy Elements, on page 16
- Traffic Class Elements, on page 12

Associated Commands

- bandwidth
- bandwidth remaining
- class
- police
- policy-map
- priority
- queue-limit
- service-policy
- set discard-class
- set dscp
- set mpls experimental
- set precedence
- set qos-group
- shape

Scaling of Unique Ingress Policy Maps

Table 4: Feature History Table

Feature Name	Release Information	Feature Description
Scaling of Unique Ingress Policy Maps	Release 7.3.1	With this feature, unique policy maps associated to the same template are shared in TCAM, thus enabling scaling of — or creating more number of — policy maps.

Traditionally, when unique policy maps were associated to the same template — that is, having the same match criteria and actions in the *same* order — each map was assigned a different TCAM entry. This resulted in inefficient TCAM entry management and also restricted the number of policy maps that could be created.

With this functionality, unique policy maps associated to the same template are shared in TCAM, thus enabling scaling of—in other words, creating more number of—policy maps. The other way to understand this functionality is that two policy maps with the same combination of criteria and actions use one template. This way, up to 250 templates are supported for association to policy map combinations.

As an example, consider the following policy maps (**policy-map ncs_input1** and **policy-map ncs_input2**) having the same class maps (**class COS7_DEI0** and class **COS7_DEI1**):

```
class-map match-all COS7_DEI0
 match cos 0
 end-class-map
class-map match-all COS7_DEI1
 match cos 1
 end-class-map

policy-map ncs_input1
 class COS7_DEI0
  set trafiic class 1
  police rate 10 mbps
!
 class COS7_DEI1
  set traffic class 2
  policer rate 20 mbps
!

policy-map ncs_input2
 class COS7_DEI0
  set traffic class 1
  police rate 30 mbps
!
 class COS7_DEI1
  set traffic class 2
  policer rate 40 mbps
!
```

Earlier, when the policy maps were attached to interface, they used different TCAM entries, although the match criteria and actions were the same, except for the policer action.

With this functionality, both policy maps share the TCAM entry instead of selecting different entries, thus freeing up TCAM entries for more policy maps.

Limitations and Restrictions

- Policy Maps share TCAM entries only for the same match criteria and actions or template. However, the policer action can be different for the same template.
- For all unique policy maps the maximum number of templates supported is 250.

Attach a Traffic Policy to an Interface

After the traffic class and the traffic policy are created, you must attach the traffic policy to interface, and specify the direction in which the policy should be applied.

Note When a policy-map is applied to an interface, the transmission rate counter of each class is not accurate. This is because the transmission rate counter is calculated based on the exponential decay filter.

Configuration Example

You have to accomplish the following to attach a traffic policy to an interface:

1. Creating a traffic class and the associated rules that match packets to the class (see #unique_17)
2. Creating a traffic policy that can be attached to one or more interfaces to specify a service policy (see Create a Traffic Policy, on page 17)
3. Associating the traffic class with the traffic policy
4. Attaching the traffic policy to an interface, in the ingress or egress direction

```
Router# configure
Router(config)# interface HundredGigE 0/6/0/18
Router(config-int)# service-policy output
Router(config-int)# commit


RP/0/RP0/CPU0:R1(config)# interface twentyFiveGigE 0/0/0/26.1
RP/0/RP0/CPU0:R1(config-if)# service-policy input cos
RP/0/RP0/CPU0:R1(config-if)# commit
```

Running Configuration

```
RP/0/RP0/CPU0:R1# show run interface TwentyFiveGigE0/0/0/26.1

interface TwentyFiveGigE0/0/0/26.1 l2transport
encapsulation dot1q 25
service-policy input cos
!

RP/0/RP0/CPU0:R1# show run policy-map cos

policy-map cos
class cos1
police rate 3 mbps
!
!
```

```
class cos2
police rate 2 mbps
!
!
class cos3
police rate 3 mbps
!
!
class class-default
police rate 4 mbps
!
!
end-policy-map
!

RP/0/RP0/CPU0:R1#
```

Verification

```
Router# show qos interface hundredGigE 0/6/0/18 output

NOTE:- Configured values are displayed within parentheses Interface HundredGigE0/6/0/18 ifh
 0x30001f8  -- output policy
NPU Id:                        3
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
VOQ Base:                      11112
VOQ Stats Handle:              0x88430698
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   qos-1
Egressq Queue ID                         =   11113 (LP queue)
Queue Max. BW.                           =   40329846 kbps (40 %)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 / (BWR not configured)
Guaranteed service rate                  =   40000000 kbps
TailDrop Threshold                       =   50069504 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class                             =   class-default
Egressq Queue ID                         =   11112 (Default LP queue)
Queue Max. BW.                           =   101803495 kbps (default)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 / (BWR not configured)
Guaranteed service rate                  =   50000000 kbps
TailDrop Threshold                       =   62652416 bytes / 10 ms (default)
WRED not configured for this class
```

Related Topics

- Traffic Policy Elements, on page 16
- Traffic Class Elements, on page 12

Associated Commands

- service-policy

Packet Marking

The packet marking feature provides users with a means to differentiate packets based on the designated markings.

Egress packet marking

- The router supports egress packet marking. Match on **discard-class** on egress, if configured, can be used for a marking policy only.
- An egress policy is mandatory to create a user-defined header by stripping the initial 3 bits and copying the DSCP marking to the QoS group.

L2 ingress marking

- **Ingress marking limitations:** For the ingress pop operation, re-marking the customer VLAN tag (CoS, DEI) is not supported.
- **Egress traffic behavior:** The ingress 'pop VLAN' is translated to a 'push VLAN' for the egress traffic, and (CoS, DEI) marking is supported for newly pushed VLAN tags. If two VLAN tags are pushed to the packet header at the egress side, both inner and outer VLAN tags are marked. For example:

```
rewrite ingress tag pop 1 symmetric
rewrite ingress tag pop 2 symmetric
rewrite ingress tag translate 2-to-1 dot1q/dot1ad <> symmetric
```

VLAN tag details

- **Single tag operations:** When symmetrical pop 1 action is performed, the outer tag (CoS, DEI) is retained as the original frame.

IP traffic class imposition for SRv6 encapsulation

Starting Cisco IOS XR Software Release 24.2.1, you can set the IP traffic class imposition for SRv6 encapsulation using the new **set ip encapsulation class-of-service** CLI.

This feature allows you to set the DSCP traffic-class marking to 6 bits using the **hw-module profile segment-routing srv6 mode microsegment f3216 encapsulation traffic-class policy-map-extend** command. This configuration replaces the 3-bit DSCP marking set by the **hw-module profile segment-routing srv6 mode microsegment f3216 encapsulation traffic-class policy-map** command to 6 bits.

Packet Marking Guidelines and Limitations

Avoid conflicting packet marking configurations: While marking a packet, ensure you don't set the IP DSCP (using the **set dscp** command) and the MPLS experimental imposition values (using the **set mpls experimental imposition** command) for the same class map. Else, neither the DSCP remarking nor the MPLS EXP values may take effect at the ingress. This will cause, per default QoS behavior, the IP precedence values to be copied to the EXP bits on the imposed packets. Such an action could lead to unintended packets marked as high-priority by your customer being forwarded as high-priority MPLS packets in the network.

Viewing statistics and counters for egress policies: The statistics and counters for the egress marking policy cannot be viewed on the router.

QoS EXP matching for Layer 2 and Layer 3 VPN traffic:

QoS EXP matching for egress doesn't work for Layer 2 VPN and Layer 3 VPN traffic flowing from:

- Cisco NCS 5700 series line cards at ingress to Cisco NCS 5500 series line cards at the egress and
- Cisco NCS 5500 series line cards at ingress to Cisco NCS 5700 series line cards at egress.

Policy map limits for egress marking:

- For QOS EXP-Egress marking applied on a Layer 3 interface on Cisco routers, there is a limit of two unique policy maps per NPU.

  You can apply these policies to as many interfaces as your system resources allow. However, if you apply more than the permitted limit of unique policies, you may encounter unexpected failure.
- For QOS egress marking (CoS, DEI) applied on a Layer 2 interface, there is a limit of 13 unique policy-maps per NPU. If you exceed this number, you may encounter unexpected failure.
- Cisco NCS series routers do not support push or translate operations for dot1ad.

Usage guidelines for SRv6 encapsulation marking:

- The **set ip encapsulation class-of-service** command is not allowed in egress QoS policies.
- The IP encapsulation marking (**set ip encapsulation class-of-service**) and the MPLS experimental imposition marking (**set mpls experimental**) features are mutually exclusive. Both the configurations are not allowed at the same time.
- On Cisco NCS 5500 series routers, the **set ip encapsulation class-of-service** command is allowed only when the **policy-map-extend** option is enabled. On Cisco NCS 5700 series routers, the command is supported in native mode without enabling the **policy-map-extend** option.
- The **policy-map-extend** option enables 6-bit DSCP marking for both L2 and L3 modes at the same time. It is not feasible to enable the **policy-map-extend** option exclusively for L2 and L3 modes due to the shared hardware resources for L2 and L3 SRv6 traffic-class.
- To support DSCP marking in NCS 5700, the **propagate** option must be enabled in the SRv6 configurations.

Egress EXP Marking

Table 5: Feature History Table

Feature Name	Release Information	Feature Description
Egress EXP bits Marking Support on Single-Label	Release 7.11.1	With this feature support, you can now mark MPLS EXP bits on single-label scenario. You can now design the QoS for your network based on either single-label or global label. If the directly connected peer router doesn't advertise the explicit-null label, then the router encapsulates the packets with only the global label. Previously, the router was limited to mark MPLS EXP bits with a minimum of two labels. But even with a single EXP marking, you can prioritize certain types of traffic over others based on their importance.

Supported Packet Marking Operations

This table shows the supported packet marking operations.

Table 6: Supported Ingress Marking Types

Supported Mark Types	Range	Layer	Supports Unconditional Marking	Supports Conditional Marking	Description
DSCP	0-63	L3	Ingress	No	Supported flows: IP to IP, IP to MPLS, MPLS to IP.
precedence	0-7	L3	Ingress	No	Supported flows: IP to IP, IP to MPLS, MPLS to IP.
mpls experimental topmost	0-7	L3	Ingress	No	Supported flows: MPLS to MPLS.
mpls experimental imposition	0-7	L2 L3	Ingress	No	Supported flows: L2 to MPLS, IP to MPLS.
ip encapsulation	0-63	L2 L3	Ingress	No	Used for marking encapsulated SRv6 packets.
DEI	0-1	L2	Ingress	No	Supported flows: L2 to L2, L2 to MPLS.

Supported Mark Types	Range	Layer	Supports Unconditional Marking	Supports Conditional Marking	Description
CoS	0-7	L2	Ingress	No	Supported flows: L2 to L2, L2 to MPLS.
discard-class	0-2	L2 L3	Ingress	No	Used in conjunction with random-detect for discard behavior identification.
qos-group	0-7	L2 L3	Ingress	No	Marks traffic for QoS management.
traffic-class	0-7	L2 L3	Ingress	No	Used for queuing purposes.

Class-based Unconditional Packet Marking

The packet marking feature allows you to partition your network into multiple priority levels or classes of service, as follows:

- Use QoS unconditional packet marking to set the IP precedence or IP DSCP values for packets entering the network. Routers within your network can then use the newly marked IP precedence values to determine how the traffic should be treated.

  On ingress direction, after matching the traffic based on either the IP Precedence or DSCP value, you can set it to a particular discard-class. Weighted random early detection (WRED), a congestion avoidance technique, thereby uses discard-class values to determine the probability that a packet is dropped.

  If however, you set a discard-class of 3, the packet is dropped at ingress itself.

- Use QoS unconditional packet marking to assign MPLS packets to a QoS group. The router uses the QoS group to determine how to prioritize packets for transmission. To set the traffic class identifier on MPLS packets, use the **set traffic-class** command in policy map class configuration mode.

Note Setting the traffic class identifier does not automatically prioritize the packets for transmission. You must first configure an egress policy that uses the traffic class.

Note

- Unless otherwise indicated, the class-based unconditional packet marking for Layer 3 physical interfaces applies to bundle interfaces.

Handling QoS for Locally Originated Packets

What are Locally Originated Packets?

Locally Originated Packets (LOPs) are packets generated and transmitted by the router itself. These differ from transit packets that pass through the router. LOPs include routing protocols, management protocols, and other locally initiated control messages.

Default QoS markings for protocols

The router assigns default precedence or Differentiated Services Code Point (DSCP) values to various locally originated protocols

Protocol or application	Precedence and DSCP value	L2 PCP value
BGP, OSPF, CCM (CSM), BFD, RSVP	• Precedence: 6 • DSCP: 48	• BGP: 6 (with or without MPLS encap) • BFD: 7 (with or without MPLS encap)
Telnet, SSH	• Precedence: 2 • DSCP: 16	Not applicable
SNMP	Precedence: 0	6 (with or without MPLS encap)
SSH, SFTP	Not applicable	7 (with or without MPLS encap)

Some protocols such as BGP, RSVP, CFM, and LDP and the management protocols allow setting explicit precedence or DSCP values.

Traffic class assignment by Cisco IOS XR release

Cisco IOS XR release	Traffic class value
Before Release 7.6.1	• Locally generated control plane packets, such as IS-IS and BGP, are generated using traffic-class 6. • Locally generated BFD over Bundle (IETF) packets, which are generated on the Network Processing Unit (NPU), are generated using traffic-class 6.
From Release 7.6.1 onwards	• Locally generated control plane packets, such as IS-IS and BGP, are generated using traffic-class 7. • Locally generated BFD over Bundle (IETF) packets, which are generated on the Network Processing Unit (NPU), are generated using traffic-class 7.

Managing datapath and locally originated packets

On the router, datapath packets and injected packets aren't differentiated if both their traffic classes share the same Virtual Output Queues (VOQs). Therefore, in the case of a congested VOQ, the LOCP packets are

dropped. To avoid the LOCP packets drop, Cisco recommends that you have a different traffic class for data path traffic. Alternatively, you can also specify a higher bandwidth for traffic-class 7 (if ingress traffic rate is predictable).

Optimizing traffic handling by early classification and marking

Classifying traffic helps the router to recognize traffic as a certain type and mark that traffic. By marking traffic early on its travel, you can prevent excessive reclassification later. You can mark traffic at the protocol level as shown in the following examples:

Ethernet

The following configuration shows that the outbound Control Hub packets are marked with a precedence value of 2 and EXP of 2, instead of a precedence and EXP value of 6. The SSH packets have a precedence value of 3 instead of 2.

```
ethernet cfm
  mep domain FOO service FOOBAR mep-id 1
   cos 2
ssh server dscp 24
```

BGP

```
neighbor x.x.x.x dscp
```

MPLS LDP

```
mpls ldp signalling dscp
```

Telnet

```
telnet ipv4 dscp
```

SNMP

```
snmp-server ipv4 precedence/dscp
```

Syslog

```
logging ipv4 precedence/dscp
netflow
flow exporter-map TEST dscp
```

NTP

```
ntp ipv4 precedence/dscp
ssh client dscp 56
ssh server dscp 56
```

Note By default, the router marks the Precision Time Protocol (PTP) traffic as high priority. Therefore, the need to prioritize PTP traffic in the QoS configuration is not required.

LOCPs and QoS policies

LOCPs are packets generated by the router itself, such as IS-IS, ARP, and other non-IP-based control packets. These packets are critical for network operations and are treated with high priority to ensure they are not dropped under normal circumstances.

Note By default, all LOCPs are assigned to traffic-class 7. Considering that LOCPs and LOMPs are generated by the RP, an Ingress QoS policy cannot be applied. Therefore, you must ensure that the egress QoS policy includes a class-map which matches traffic-class 7. By definition, the egress QoS policy matches all implicitly marked packets.

Key features of LOCPs

Discard priority

- LOCPs originating from the RP or LC CPU have a discard priority set in the appended Buffer Header (BHDR).
- This ensures LOCPs are not dropped internally under normal conditions.
- LOMPs do not have this discard priority and are treated as normal traffic.

Queue management

- LOCPs with discard priority are placed in an implicitly allocated high-priority queue for each physical egress interface.
- During congestion, LOCPs are not subject to Weighted Random Early Detection (WRED) or Tail-drop queue-limit operations.
- The tail-drop queue-limit must be hit before LOCP packets are dropped.

QoS policy configuration

- QoS policies can be attached to physical interfaces or sub-interfaces.
- If attached to sub-interfaces, the operator cannot attach a QoS policy to the physical interface.
- LOCPs are always sent out on the default high-priority queue of the physical interface.

Bandwidth reservation

- A minimum bandwidth of MIN (1% of interface bandwidth, 10 Mbps) is reserved for the default high-priority queue on physical interfaces without a QoS policy.
- If a QoS policy is applied, the minimum bandwidth for the high-priority queue is controlled by the configured policy.

Classification and re-marking

- LOCPs are not subject to traffic policing or re-marking.
- Non-IP LOCPs, such as IS-IS and ARP, are always sent to the high-priority queue, irrespective of the QoS policy.

Session termination prevention

- During over-subscription, LOCP drops may occur, leading to session termination.
- Proper bandwidth allocation and QoS configuration are essential to prevent such scenarios.

QoS packet marking across network layers

QoS packet marking is a networking management technique used to classify and prioritize data packets. Using this, you can enable the application of traffic handling policies such as bandwidth allocation and congestion management.

This table describes the key QoS fields in packet headers.

Table 7:

Field	Layer	Purpose
L3 DSCP	Layer 3	Classify and prioritize packets in the IP header.
MPLS EXP	Layer 2.5 (MPLS)	Carry QoS information in MPLS networks.
QoS Group	Cisco	Internal classification for flexible QoS policy application.
Traffic Class	Layer 3 (IPv6)	Define QoS policies in IPv6 headers.
L2 PCP	Layer 2	Prioritize traffic in Ethernet networks using the 802.1Q VLAN tag.

QoS packet marking across network layers is explained with and without MPLS encapsulation.

- QoS packet marking across network layers with MPLS encapsulation
- QoS packet marking across network layers without MPLS encapsulation

QoS packet marking across network layers with MPLS encapsulation

In MPLS networks, the QoS marking is done using the 3-bit MPLS EXP field in the MPLS label. This field is used to classify and prioritize traffic as it traverses the MPLS network. The EXP field is often derived from the IP precedence or DSCP value of the original packet at the ingress of the MPLS network.

This table illustrates how QoS packet marking is applied to network protocols when using MPLS encapsulation across various large density routers.

Table 8:

Router	Routing Protocol	L3 DSCP	MPLS EXP	QoS group	Traffic class	L2 PCP
Large • N540-24Q8L2DD-SYS • N540-24Q2C2DD-SYS	BFD	Cs6	6	QG6	7	6
	BFD (HW Offload)	Cs6	6	QG6	7	6
	SRPM	Cs6	6	QG6	7	6
	SRPM (NPU Offload)	Cs6	6	QG6	7	6
	ISIS	NA	6	QG6	7	6
	BGP	Cs6	6	QG6	7	6
	SSH/SFTP	Cs6	6	QG6	7	6
	NTP	Cs6	6	QG6	7	6
	TACACS	Cs6	6	QG6	7	6
	SNMP	Cs6	6	QG6	7	6
	SYSLOG	Cs6	6	QG6	7	6

This table illustrates how QoS packet marking is applied to network protocols when using MPLS encapsulation across various medium density routers.

Router	Routing Protocol	L3 DSCP	MPLS EXP	QoS group	Traffic class	L2 PCP
Medium • N540-ACC-SYS • N540X-ACC-SYS • N540-24Z8Q2C-SYS • N540X-16Z4G8Q2C-D/A • N540X-16Z8Q2C-D • N540-28Z4C-SYS-D/A • N540X-12Z16G-SYS-D/A • N540-12Z20G-SYS-D/A	BFD	Cs6	6	QG7	7	7
	BFD (HW Offload)	Cs6	6	QG7	7	7
	SRPM	Cs7	6	QG7	7	7
	SRPM (NPU Offload)	Cs7	6	QG7	7	7
	ISIS	NA	6	QG7	7	7
	BGP	Cs6	6	QG6	7	6
	SSH/SFTP	Cs6	6	QG6	7	6
	NTP	Cs6	6	QG6	7	6
	TACACS	Cs6	6	QG6	7	6
	SNMP	Cs6	6	QG6	7	6
	SYSLOG	Cs6	6	QG6	7	6
	ICMP	Cs7	6	QG7	7	6

This table illustrates how QoS packet marking is applied to network protocols when using MPLS encapsulation across various small density routers.

Router	Routing Protocol	L3 DSCP	MPLS EXP	QoS group	Traffic class	L2 PCP
Small • N540X-4Z14G2Q-D/A • N540X-8Z16G-SYS-D/A • N540X-6Z18G-SYS-D/A • N540-6Z18G-SYS-D/A • N540-6Z14S-SYS-D	BFD	Cs6	6	QG7	7	7
	BFD (HW Offload)	Cs6	6	QG7	7	7
	SRPM	Cs7	6	QG7	7	7
	SRPM (NPU Offload)	Cs7	6	QG7	7	7
	ISIS	NA	6	QG7	7	7
	BGP	Cs6	6	QG6	7	6
	SSH/SFTP	Cs6	6	QG6	7	7
	NTP	Cs6	6	QG6	7	6
	TACACS	Cs6	6	QG6	7	6
	SNMP	Cs6	6	QG6	7	6
	SYSLOG	Cs6	6	QG6	7	6
	ICMP	Cs6	6	QG6	7	6

QoS packet marking across network layers without MPLS encapsulation

The layerwise QoS packet marking refers to the classification and marking of packets at different layers to prioritize traffic based on QoS policies.

This table shows how QoS packet marking is applied without MPLS encapsulation across various large density routers.

Table 9:

Router	Routing Protocol	L3 DSCP	QoS group	Traffic class	L2 PCP
Large • N540-24Q8L2DD-SYS • N540-24Q2C2DD-SYS	BFD	Cs6	QG6	7	6
	BFD (HW Offload)	Cs6	QG6	7	6
	SRPM	Cs6	QG6	7	6
	SRPM (NPU Offload)	Cs6	QG6	7	6
	ISIS	NA	QG6	7	6
	BGP	Cs6	QG6	7	6
	SSH/SFTP	Cs6	QG6	7	6
	NTP	Cs6	QG6	7	6
	TACACS	Cs6	QG6	7	6
	SNMP	Cs6	QG6	7	6
	SYSLOG	Cs6	QG6	7	6

This table shows how QoS packet marking is applied without MPLS encapsulation across various medium density routers.

Router	Routing Protocol	L3 DSCP	QoS group	Traffic class	L2 PCP
Medium • N540-ACC-SYS • N540X-ACC-SYS • N540-24Z8Q2C-SYS • N540X-16Z4G8Q2C-D/A • N540X-16Z8Q2C-D • N540-28Z4C-SYS-D/A • N540X-12Z16G-SYS-D/A • N540-12Z20G-SYS-D/A	BFD	Cs6	QG7	7	7
	BFD (HW Offload)	Cs6	QG7	7	7
	SRPM	Cs7	QG7	7	7
	SRPM (NPU Offload)	Cs7	QG7	7	7
	ISIS	NA	QG7	7	7
	BGP	Cs6	QG6	7	6
	SSH/SFTP	Cs6	QG6	7	6
	NTP	Cs6	QG6	7	6
	TACACS	Cs6	QG6	7	6
	SNMP	Cs6	QG6	7	6
	SYSLOG	Cs6	QG6	7	6
	ICMP	Cs6	QG6	7	6

This table shows how QoS packet marking is applied without MPLS encapsulation across various small density routers.

Router	Routing Protocol	L3 DSCP	QoS group	Traffic class	L2 PCP
Small • N540X-4Z14G2Q-D/A • N540X-8Z16G-SYS-D/A • N540X-6Z18G-SYS-D/A • N540-6Z18G-SYS-D/A • N540-6Z14S-SYS-D	BFD	Cs6	QG7	7	7
	BFD (HW Offload)	Cs6	QG7	7	7
	SRPM	Cs7	QG7	7	7
	SRPM (NPU Offload)	Cs7	QG7	7	7
	ISIS	NA	QG7	7	7
	BGP	Cs6	QG6	7	6
	SSH/SFTP	Cs6	QG6	7	7
	NTP	Cs6	QG6	7	6
	TACACS	Cs6	QG6	7	6
	SNMP	Cs6	QG6	7	6
	SYSLOG	Cs6	QG6	7	6
	ICMP	Cs6	QG6	7	6

Protecting Locally Originated BFD Packets

For releases before Release 7.6.1, BFD packets are injected into traffic-class 6, with drop priority 0 (equivalent of discard-class 0). If transit traffic is also classified into traffic-class 7 and the associated egress queue is congested, BFD packets may be dropped.

From Release 7.6.1, BFD packets are injected into traffic-class 7, with drop priority 0 (equivalent of discard-class 0). If transit traffic is also classified into traffic-class 7 and the associated egress queue is congested, BFD packets may be dropped.

The recommendation is to configure transit traffic-class 7 in the ingress QoS policy with discard-class 1 or 2. You must then configure WRED in the egress QoS policy that drops these packets before dropping discard-class 0.

Note The default queue length is 16 ms.

Example

```
class-map match-any NC
 match traffic-class 6 (for releases before Release 7.6.1)
 match traffic-class 7 (from Release 7.6.1)
!
policy-map CORE-OUT
```

```
 class NC
  random-detect discard-class 1 7 ms 8 ms
```

Hardware Programming

```
RP/0/RP0/CPU0:R26-Spine2-5508#sh qos interface hu0/0/0/35 output
NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/0/0/35 ifh 0x1408  -- output policy
NPU Id:                        5
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
Policy Name:                 7  wred-out
SPI Id:                        0x0
VOQ Base:                      1592
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   NC
Egressq Queue ID                         =   1598 (LP queue)
Queue Max. BW.                           =   20480000 kbps (20 %)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 / (BWR not configured)
Guaranteed service rate                  =   20000000 kbps
Peak burst                               =   33600 bytes (default)
TailDrop Threshold                       =   40108032 bytes / 16 ms (default)

WRED profile for Discard_Class 1
WRED Min. Threshold                      =   17563648 bytes (7 ms)
WRED Max. Threshold                      =   20054016 bytes (8 ms)

Default RED profile
WRED Min. Threshold                      =   0 bytes (0 ms)
WRED Max. Threshold                      =   0 bytes (0 ms)

WRED ECN                                 =   Disabled
```

Prioritization of IS-IS and ARP Packets to Manage Transit Traffic

Table 10: Feature History Table

Feature Name	Release Information	Feature Description
Prioritization of IS-IS and ARP Packets to Manage Transit Traffic	Release 7.5.1	This feature gives you the option to assign the highest priority to IS-IS and Address Resolution Protocol (ARP) packets in transit. This feature is disabled by default. The feature provides more flexibility in transit traffic management on a per-hop basis and also fine-tunes the traffic profile management for transit traffic. This feature introduces the command.

Overview of IS-IS and ARP Traffic Prioritization

Transit traffic refers to all traffic that enters an ingress interface, is compared against the forwarding table entries, and forwarded out an egress interface toward its destination. While the exact path of the transit path

may not be of interest to the sender or receiver, you may still want some of the Integrated Intermediate System-to-Intermediate System (IS-IS) and Address Resolution Protocol (ARP) transit traffic to be managed and routed efficiently between specific source and destination addresses. You can now achieve higher levels of flexibility and fine-tune the traffic profile management for transit traffic by enabling the ability to assign the highest priority level to IS-IS and ARP traffic on Layer 2 networks.

This feature is useful if you manage environments such as data centers where you have complete end-to-end control over your network, and you want to avoid any drops in IS-IS and ARP traffic during congestion.

Cisco IOS XR Release 7.5.1 introduces the **hw-module profile qos arp-isis-priority-enable** command to enable prioritization of IS-IS and ARP traffic in transit on Layer 2 networks. Configuring this command assigns a priority level of TC 7 to transit traffic.

Note Assigning highest priority levels to IS-IS and ARP traffic lead to higher volumes of ARP traffic and flood high-priority queues, which may cause other traffic to drop. It's best to assign priority levels based on your network configuration and traffic volume.

Guidelines

- This feature is disabled by default. Configure the **hw-module profile qos arp-isis-priority-enable** command to enable prioritization of IS-IS and ARP traffic in transit on Layer 2 networks.
- Configuring **hw-module profile qos arp-isis-priority-enable** assigns a priority level of TC 7 for ISIS and ARP traffic. When this feature is in its default state of disabled, the default priority level is TC 0.
- This feature doesn't allow you to assign TC values.
- Reload the line card for the **hw-module** command to be functional.

Enabling IS-IS and ARP Traffic Prioritization

To enable IS-IS and ARP traffic prioritization, configure the **hw-module profile qos arp-isis-priority-enable** command.

```
Router#config
Router(config)#hw-module profile qos arp-isis-priority-enable
Router(config)#commit
Router(config)#exit
Router# reload location <lc location>
```

Prioritization is based on IS-IS destination MAC address (01:80:c2:00:00:14 and 01:80:c2:00:00:15) and ARP ether type 0x080. When you configure the **hw-module profile qos arp-isis-priority-enable** command, priority level for IS-IS and ARP and traffic is set as TC 7.

Verification

The following example shows the verification command for line cards. The assigned priority level is **TC 07.**

```
RP/0/RP0/CPU0:ios#show controllers fia diag 0 "pp vis pkttm" location 0/0/CPU0

Node ID: 0/0/CPU0

R/S/I: 0/0/0
================================================
|     Packet TM Information Core=0 Results      |
================================================
```

```
| Name                         | Value    | Size    |
|                              |          | (bits)  |
================================================
| Fwd_Action_Destination       | 0c353e   |  21     |
|   type                       | PORT_ID  |  6      |
|   value                      | 353e     |  15     |
| Tc                           | 07       |  3      |
| Dp                           | 00       |  2      |
| Int_Stat_Meter_Dp_Cmd        | 00       |  4      |
| Lag_Lb_Key                   | 1772     |  16     |
| St_Vsq_Ptr                   | 00       |  8      |
| Visibility                   | 01       |  1      |
| System_Headers_Size          | 28       |  7      |
| NWK_Header_Truncate_Size     | 00       |  8      |
| NWK_Header_Append_Size_Ptr   | 02       |  8      |
| Counter_ID_0                 | N/A      |  0      |
| Counter_ID_1                 | N/A      |  0      |
| Counter_ID_2                 | N/A      |  0      |
| Counter_ID_3                 | N/A      |  0      |
| Counter_ID_4                 | N/A      |  0      |
| Counter_ID_5                 | N/A      |  0      |
| Counter_ID_6                 | N/A      |  0      |
| Counter_ID_7                 | 00a000   |  20     |
| Counter_ID_8                 | N/A      |  0      |
| Counter_ID_9                 | N/A      |  0      |
| Meter_ID_0                   | N/A      |  0      |
| Meter_ID_1                   | N/A      |  0      |
| Meter_ID_2                   | N/A      |  0      |
| Ethernet_Meter_ID            | 02       |  3      |
| snif0_cmd                    | 00       |  5      |
| snif0_code                   | 0009     |  9      |
| snif0_qualifier              | 00       |  8      |
| snif1_cmd                    | 00       |  5      |
| snif1_code                   | 01e0     |  9      |
| snif1_qualifier              | 00       |  8      |
| snif2_cmd                    | 00       |  5      |
| snif2_code                   | 0000     |  9      |
| snif2_qualifier              | 00       |  8      |
================================================
```

QoS Re-marking of IP Packets in Egress Direction

The router support the marking of IP DSCP bits of all IP packets to zero, in the egress direction. This feature helps to re-mark the priority of IP packets, which is mostly used in scenarios like IP over Ethernet over MPLS over GRE. This functionality is achieved using the ingress policy-map with **set dscp 0** option configured in class-default.

Configuration Example

```
Router# configure
Router(config)# policy-map ingress-set-dscp-zero-policy
Router(config-pmap)# class class-default
Router(config-pmap-c)# set dscp 0
Router(config-pmap-c)# end-policy-map
Router(config-pmap)# commit
```

Running Configuration

```
policy-map ingress-set-dscp-zero-policy
class class-default
  set dscp 0
!
end-policy-map
!
```

QoS Re-marking of Ethernet Packets in Egress Direction

The router supports Layer 2 marking of Ethernet packets in the egress direction.

QoS L2 Re-marking of Ethernet Packets in Egress Direction

The router supports Layer 2 marking of Ethernet packets in the egress direction.

To enable this feature, you must:

- Configure the policy maps for queuing and marking at the egress interface.
- Set traffic-class in the ingress and use **match traffic-class** in the egress for queuing.
- Ensure that the **set qos-group** command is configured in ingress policy and the corresponding **match qos-group** command is configured in the egress marking policy. If there is no corresponding QoS group, you will experience traffic failure.

  The ingress 'push VLAN' is translated to 'pop VLAN' for the egress traffic. In this case, (CoS, DEI) re-marking is not supported for the VLAN tag. For example:

  1. rewrite ingress tag push dot1q/dot1ad <> symmetric
  2. rewrite ingress tag push dot1q/dot1ad <> second-dot1q <> symmetric
  3. rewrite ingress tag translate 1-to-2 dot1q/dot1ad <> second-dot1q <> symmetric

Running Configuration

```
policy-map egress-marking
class qos1
set cos 1
!
class qos2
set cos 2
set dei 1
!
class qos3
set cos 3
!
class class-default
set cos 7
!
end-policy-map
!
```

QoS L2 Re-Marking of Ethernet Packets on L3 Flows in Egress Direction

The router supports Layer 2 marking of Ethernet packets on Layer 3 flows in the egress direction. To enable this feature, you must:

- Configure the policy maps for marking at the egress interface.
- Ensure that the **set qos-group** command is configured in ingress policy and the corresponding **match qos-group** command is configured in the egress marking policy. If there is no corresponding QoS group, you will experience traffic failure.

Restrictions

The following restrictions apply while configuring the Layer 2 marking of Ethernet packets on Layer 3 flows in the egress direction.

- **set discard-class** is not supported in ingress policy with peering mode.
- Egress marking statistics are not available.
- Layer 2 (802.1p) Egress marking is supported on Layer 3 flows for these types of traffic: IP-to-IP, IP-to-MPLS, and MPLS-to-IP traffic.
- Layer 2 marking of Ethernet packets on Layer 3 flows in the egress direction is supported only in the peering mode.

Running Configuration

Ingress Policy:

You must first set up the qos-group at ingress.

```
class-map match-any Class0
 match mpls experimental topmost 0

 end-class-map
class-map match-any Class1
 match mpls experimental topmost 1

 end-class-map
class-map match-any Class2
 match mpls experimental topmost 2

 end-class-map
class-map match-any Class3
 match mpls experimental topmost 3

 end-class-map
class-map match-any Class4
 match mpls experimental topmost 4

 end-class-map
class-map match-any Class5
 match mpls experimental topmost 5

 end-class-map
class-map match-any Class6
 match mpls experimental topmost 6
end-class-map
class-map match-any Class7
```

```
 match mpls experimental topmost 7

 end-class-map
!

policy-map ncs_input
 class Class7
  set traffic-class 7
    set qos-group 7
!
 class Class6
  set traffic-class 6
  set qos-group 6
!
 class Class5
  set traffic-class 5
    set qos-group 5
!
 class Class4
  set traffic-class 4
  set qos-group 4
 !
 class Class3
  set traffic-class 4
    set qos-group 3
 !
 class Class2
  set traffic-class 2
    set qos-group 2
 !
 class Class1
  set traffic-class 2
    set qos-group 1
 !
 class Class0
  set traffic-class 0
    set qos-group 0
 !
 end-policy-map
!
```

Egress Policy:

At the egress, run these commands to mark the packets.

```
class-map match-any qos7
match gos-group 7
 end-class-map
!
class-map match-any qos6
match gos-group 6
 end-class-map
!
class-map match-any qos5
match qos-group 5
 end-class-map
!
class-map match-any qos4
match gos-group 4
 end-class-map
!
class-map match-any qos3
match gos-group 3
 end-class-map
!
```

```
class-map match-any qos2
 match gos-group 2
 end-class-map
!
class-map match-any qos1
match gos-group 1
 end-class-map
!

policy-map ncs_output
 class qos7
  set cos 7
  !
 class qos6
  set cos 6
!
 class qos5
  set cos 5
  !
 class qos4
  set cos 4
!
 class qos3
  set cos 3
!
 class qos2
  set cos 2

!
 class qos1
  set cos 1
  !
 end-policy-map
!
```

Layer 2 Ingress QoS Matching for IPv4 and IPv6 Destination Addresses

Table 11: Feature History Table

Feature Name	Release Information	Feature Description
Layer 2 Ingress QoS Matching for IPv4 and IPv6 Destination Addresses	Release 7.5.1	Using this feature, you can match class maps to IPv4 and IPv6 destination addresses on Layer 2 networks. The Layer 2 interface service policy has the relevant class maps, actioning them for ingress QoS operations. This feature provides you with an additional level of classification for aggregated customer traffic at your ingress, thus giving you granular control on traffic flows. This feature introduces the following commands: • **match destination-address** • **hw-module profile qos l2-match-dest-addr-v4v6**

Overview

As a service provider, you provide Layer 2 connectivity for different classes of customer traffic across your network. With aggregated customer traffic arriving at your ingress, you need to provide differential treatment depending on specific destination addresses for the traffic. Such ability gives you granular control over traffic, allowing you to classify specific traffic flows depending on the type of services for which your customers have signed up.

You can match class maps to IPv4 and IPv6 destination addresses on Layer 2 networks to ensure such granular control. The interface service policy has the relevant class maps, actioning them for ingress QoS marking.

Guidelines and Limitations

- You can match up to 4 IPv4 and IPv6 addresses each in a class.
- For match on IPv6, only up to 64-bit prefix match is supported.
- The L2VPN traffic can be Virtual Private Wire Service (VPWS) or Virtual Private LAN Service (VPLS).
- Redundant and non-redundant pseudowires are supported.
- This feature isn't supported with egress ACL enabled.
- Traffic classification for VLAN tags is supported as shown in the following table.

VLAN Tag Condition	IPv4 Addresses	IPv6 Addresses	Combination of IPv4 and IPv6 Addresses
With no VLAN tags	☑	☑	☑
With a single VLAN tag	☑	☑	☑
With a double VLAN tag	☑	☑	☑

Configure Layer 2 Ingress QoS Matching for IPv4 and IPv6 Destination Addresses

Perform the following steps to configure Layer 2 ingress QoS matching for IPv4 and IPv6 destination addresses. This example covers:

- **match-all** criteria for an IPv4 address and a Layer 2 classification (**match dscp**) in the same class map.

Note You can use the **match-all** criteria only when you want to match one specific destination address with other Layer 2 classification options such (CoS, DEI) or DSCP.

- **match-any** criteria for IPv4 and IPv6 addresses in the same class map.

1. Enable the ability to match class maps to IPv4 and IPv6 destination addresses on Layer 2 networks. Reload the router for the **hw-module** command to be functional.
2. Create a class map and specify **match-all** criteria for an IPv4 address and DSCP.
3. Create a class map and specify **match-any** criteria for IPv4 and IPv6 addresses.
4. Create a policy map and associate the class maps you created with the traffic policy and specify **class-action**.
5. Attach the policy map to the interface.

Configuration

```
/*Enable the ability to match class maps to IPv4 and IPv6 destination addresses on Layer 2
 networks*/
Router(config)#hw-module profile qos l2-match-dest-addr-v4v6
Router(config)#commit
Router#reload

/*Create a class map and specify match-all criteria for an IPv4 address and DSCP*/
Router(config)#class-map match-all ipv4_dst_cs1
Router(config-cmap)#match destination-address ipv4 192.168.1.4 255.255.255.255
Router(config-cmap)#match dscp cs1

/*Create a class map and specify match-any criteria for IPv4 and IPv6 addresses*/
Router(config-cmap)#class-map match-any V4_V6_MATCH
Router(config-cmap)#match destination-address ipv4 10.0.0.0 255.0.0.0
Router(config-cmap)#match destination-address ipv4 20.1.0.0 255.255.0.0
Router(config-cmap)#match destination-address ipv4 20.1.1.1 255.255.255.255
Router(config-cmap)#match destination-address ipv4 30.1.0.1 255.255.255.0
Router(config-cmap)#match destination-address ipv6 101:1:12::1/64
```

```
Router(config-cmap)#match destination-address ipv6 201:1:1::1/32
Router(config-cmap)#match destination-address ipv6 201:1:3::2/64
Router(config-cmap)#match destination-address ipv6 301:1:3::2/64
Router(config-cmap)#commit

/*Create a policy map, associate the class maps with the traffic policy; specify class-action:
 police rate, in this example*/
Router(config-cmap)#policy-map PMAP_L2_V4_V6_MATCH
Router(config-pmap)#class ipv4_dst_cs1
Router(config-pmap-c)#police rate 10 mbps
Router(config-pmap-c-police)#class V4_V6_MATCH
Router(config-pmap-c)#police rate 10 mbps
Router(config-pmap-c-police)#commit

/*Attach the policy map with class-actions that you set in the class maps*/
Router(config-pmap-c-police)#int Bundle-Ether100.2
Router(config-if)#service-policy input PMAP_L2_V4_V6_MATCH
Router(config-if)#commit
```

You have successfully configured Layer 2 ingress QoS matching for IPv4 and IPv6 destination addresses.

Running Configuration

```
qos l2-match-dest-addr-v4v6
!

class-map match-all ipv4_dst_cs1
 match destination-address ipv4 192.168.1.4 255.255.255.255
 match dscp cs1

!
 class-map match-any V4_V6_MATCH
  match destination-address ipv4 10.0.0.0 255.0.0.0
  match destination-address ipv4 20.1.0.0 255.255.0.0
  match destination-address ipv4 20.1.1.1 255.255.255.255
  match destination-address ipv4 30.1.0.1 255.255.255.0
  match destination-address ipv6 101:1:12::1/64
  match destination-address ipv6 201:1:1::1/32
  match destination-address ipv6 201:1:3::2/64
  match destination-address ipv6 301:1:3::2/64

!
!

  policy-map PMAP_L2_V4_V6_MATCH
   class ipv4_dst_cs1
    police rate 10 mbps
     class V4_V6_MATCH
    police rate 10 mbps
!
!
```

Verification

To verify that the configuration was successful, run the **sh policy-map pmap-name** command for the policy map you created with all class maps associated. The output displays all the **match-any** and **match-all** configurations for IPv4 and IPv6 addresses.

```
Router#sh policy-map pmap-name PMAP_L2_V4_V6_MATCH detail
class-map match-all ipv4_dst_cs1
 match destination-address ipv4 192.168.1.4 255.255.255.255
```

```
 match dscp cs1
 end-class-map

!
class-map match-any V4_V6_MATCH
 match destination-address ipv4 10.0.0.0 255.0.0.0
 match destination-address ipv4 20.1.0.0 255.255.0.0
 match destination-address ipv4 20.1.1.1 255.255.255.255
 match destination-address ipv4 30.1.0.1 255.255.255.0
 match destination-address ipv6 101:1:12::1/64
 match destination-address ipv6 201:1:1::1/32
 match destination-address ipv6 201:1:3::2/64
 match destination-address ipv6 301:1:3::2/64
 end-class-map
!
policy-map PMAP_L2_V4_V6_MATCH
 class ipv4_dst_cs1
  police rate 10 mbps
  !

 !
 class V4_V6_MATCH
  police rate 10 mbps
  !
 !
 class class-default
 !
 end-policy-map
!
!

Router#sh run interface bundle-ether 100
interface Bundle-Ether100
service-policy input ipv4_dst_cs1
 ipv4 address 192.168.1.4 255.255.255.255
 dscp cs1
 service-policy input PMAP_L2_V4_V6_MATCH
 ipv4 address 10.1.0.1 255.255.255.0
 ipv6 address 10:1::1/96
!
```

Bundle Traffic Policies

A policy can be bound to bundles. When a policy is bound to a bundle, the same policy is programmed on every bundle member (port). For example, if there is a policer or shaper rate, the same rate is configured on every port. Traffic is scheduled to bundle members based on the load balancing algorithm.

Both ingress and egress traffic is supported. Percentage-based policies , absolute rate-based policies, and time-based policies are supported.

Note Egress marking is not supported on BVI interfaces.

For details, see Configure QoS on Link Bundles, on page 146.

Shared Policy Instance

Table 12: Feature History Table

Feature Name	Release Information	Feature Description
Shared Policy Instance	Release 7.3.1	This feature allows you to share a single instance of QoS policy across multiple subinterfaces, allowing for aggregate shaping of the subinterfaces to one rate. The ability to facilitate queue consumption in this manner offers the advantage of saving on QoS and hardware resources, while ensuring that the specified rate is not exceeded.

Traditionally, when services required by your end-customers mapped one-on-one to an interface, attaching the QoS policy-map directly to the interface was the way to meet customer SLAs. However, with increasing demand for triple play configurations—requiring the management of voice and video queues in addition to data queues —you may have several forwarding constructs. This scenario calls for the need to apply an aggregate QoS policy across interfaces to provide the necessary traffic.

After you create the traffic class and traffic policy, you can optionally use a shared policy instance to allocate a single set of QoS resources and share them across a group of subinterfaces.

With shared policy instance, you can share a single instance of a QoS policy across multiple subinterfaces, allowing for aggregate shaping, policing, and marking of the subinterfaces to one rate. All the subinterfaces that share the instance of a QoS policy must belong to the same main interface. The number of subinterfaces that share the QoS policy instance can range from 2 to the maximum number of subinterfaces on the main interface.

When a shared policy instance of a policy map is shared by several subinterfaces, QoS operations such as aggregate shaping, policing, and marking are applied for traffic on all the interfaces that use the same shared policy instance.

Traditionally, policies were bound to interfaces. However, different types of interfaces, such as Layer 2 and Layer 3, can use a single shared-policy-instance, which allows flexibility in the "attachment point" that binds the policy map.

As an example, consider the following policy configuration:

```
policy-map hqos_gold
 class class-default
  service-policy child_hqos_gold
  shape average 20 mbps
 !
 end-policy-map
!
policy-map child_hqos_gold
 class voice
  priority level 1
  shape average 64 kbps
 !
 class video
  priority level 1
```

```
  shape average 4 mbps
 !
 class data
  bandwidth 5 mbps
 !
 class class-default
 !
 end-policy-map
!

interface TenGigE 0/1/0/10.300 l2transport
 service-policy output hqos_gold shared-policy-instance hqos_gold_customer1
!
interface TenGigE 0/1/0/10.400 l2transport
 service-policy output hqos_gold shared-policy-instance hqos_gold_customer1
!
```

The keyword **shared-policy-instance** and the instance name **hqos_gold_customer1** identify the subinterfaces that share an aggregate SLA. These are shared on a physical main interface or a bundle member. In other words, in a mix of Layer 2 and Layer 3 subinterfaces in the same shared policy instance, both layers support classification criteria and action.

In the case of bundles, sharing is applicable within a bundle member and not the entire bundle. Depending on the traffic hashing, shared policy instance may or may not take effect across the subinterface under the bundle main interface.

All subinterfaces that share the same shared policy instance share resources as well. Hence, the **show policy-map** statistics values and **show qos** values for all the subinterfaces are the same.

Restrictions and Guidelines

The following restrictions and guidelines apply while configuring shared policy instance for a policy map.

- Subinterfaces that are part of the same shared policy must belong to the same main interface. In other words, subinterfaces of different main interfaces cannot be part of the same shared policy.
- There is no restriction on the number of unique shared policies across a system. However, the limit of maximum number of subinterfaces with QoS policies applies.
- There is no restriction on the number of unique shared policies per main interface, port, core, NPU, or line card.
- You cannot use the same shared policy name on the ingress and egress of the same subinterface.
- Shared policy instance is not supported with multi-policies. For example, on the egress, you cannot apply a marking policy and a queueing policy under a shared policy.
- A shared policy can include a combination of Layer 2 and Layer 3 subinterfaces.

Attaching a Shared Policy Instance to Multiple Subinterfaces

To attach a shared policy instance to multiple subinterfaces:

1. Enter interface configuration mode and configure a subinterface.
2. Attach a policy map to an input or output subinterface for it to be the service policy for that subinterface.

```
RP/0/RP0/CPU0:router(config)#interface HundredGigE0/3/0/0.1
RP/0/RP0/CPU0:router(config-subif)#service-policy output pm-out shared-policy-instance spi1
```

Running Configuration

```
interface HundredGigE0/3/0/0.1
service-policy output pm-out shared-policy-instance spi1
ipv4 address 20.0.0.1 255.255.255.0
encapsulation dot1q 1
!
```

Verification

The show policy-map shared-policy-instance command includes an option to display counters for the shared policy instance.

Note

- For bundle subinterfaces, use the location keyword RP.
- For physical subinterfaces, use the location keyword LC.

For example, for a physical interface:

```
RP/0/RP0/CPU0:ios#show policy-map shared-policy-instance spi1 output location 0/3/CPU0

Shared Policy Instance spi1 output: pm-out

Class cm-tc-1
  Classification statistics          (packets/bytes)               (rate - kbps)
    Matched             :           772637560/1143503679080        9622860
    Transmitted         :           731260312/1082265352040        5052880
    Total Dropped       :            41377248/61238327040          4569980
  Queueing statistics
    Queue ID                             : 1433
    Taildropped(packets/bytes)           : 41377248/61238327040
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0                    0
    Transmitted         :                  0/0                    0
    Total Dropped       :                  0/0                    0
  Queueing statistics
    Queue ID                             : 1432
    Taildropped(packets/bytes)           : 0/0
Policy Bag Stats time: 1604675533816  [Local Time: 11/06/20 15:12:13.816]
```

Use the **clear qos counters shared-policy-instance** command to clear counters for the shared policy instance.

Note

- For bundle subinterfaces, use RP as the location keyword.
- For physical subinterfaces, use LC as the location keyword.

For example, for a physical interface:

```
RP/0/RP0/CPU0:ios#clear qos counters shared-policy-instance spi1 output location 0/3/CPU0
```

The **show qos shared-policy-instance** command allows you to display the QoS hardware programming values.

Note

- For bundle subinterfaces, use the location keyword RP.
- For physical subinterfaces, use the location keyword LC.

For example, for a physical interface:

```
RP/0/RP0/CPU0:ios#show qos shared-policy-instance spi1 output location 0/3/CPU0
Fri Nov  6 15:21:44.200 UTC
NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/3/0/0.1 ifh 0x60040c8  -- output policy
NPU Id:                        0
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
Policy Name:                   pm-out
SPI Id:                        0x3000001
VOQ Base:                      1432
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   cm-tc-1
Egressq Queue ID                         =   1433 (LP queue)
Queue Max. BW.                           =   5118857 kbps (5 %)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 / (BWR not configured)
Guaranteed service rate                  =   5000000 kbps
Peak burst                               =   33600 bytes (default)
TailDrop Threshold                       =   6258688 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class                             =   class-default
Egressq Queue ID                         =   1432 (Default LP queue)
Queue Max. BW.                           =   no max (default)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 / (BWR not configured)
Guaranteed service rate                  =   50000000 kbps
Peak burst                               =   33600 bytes (default)
TailDrop Threshold                       =   62652416 bytes / 10 ms (default)
WRED not configured for this class
```

Ingress Short-Pipe

When QoS traffic leaves an MPLS network, the MPLS label stack is removed on the penultimate ingress Label Switch Router (LSR), leaving an IPv4 or IPv6 packet to be forwarded. MPLS experimental bits (or EXP or pipe mode) carries out this disposition process and the packet is marked with a Differentiated Services Code Point (DSCP) or precedence value (also called DSCP or Precedence-based classification).

Usually, QoS traffic supports DSCP and precedence-based classifications only when there is no MPLS label in the packet. Using the ingress short-pipe feature, however, you can classify a packet that contains one MPLS label using the type-of-service (ToS) field of the IPv4 or IPv6 header. This classification method is called ingress short-pipe. To classify an IP packet this way, you must:

1. Create a child class map.
2. Specify a ToS value in the child class map.
3. Attach the child class map to a parent class map.

4. Create a policy map containing the parent class map.
5. Set any ingress action such as traffic class or QoS group. From Release 7.1.1 onwards, you can also set ingress action DSCP (or precedence value).

With the ingress short-pipe feature, you get an increased visibility into traffic packets. Plus, the feature also removes the limitation of classifying MPLS packets that come into IPv4 or IPv6 networks.

Restrictions and Other Important Points

Ensure that you read these points before you configure the ingress short-pipe feature.

- This feature works only when there is one MPLS header in the traffic packet. If there are two or more MPLS headers, the ingress-short pipe feature fails. For example, in case of Explicit Null where there are two labels at the disposition, this feature will not work.
- You can carry out ingress classification using either the MPLS experimental bits (or EXP or pipe mode) classification OR the DSCP/precedence (or short-pipe) classification. Ensure that you do not mix the classification methods, else it may result in an unknown behavior, and the classification may not work at all.
- This feature is supported only on L3VPN, and not supported on L2VPN.
- This feature works for regular IPv4/IPv6 traffic, but will not work for IPv6 VPN Provider Edge over MPLS (6VPE).
- You can add only one child class map to a parent class map.
- This feature supports the invocation of short-pipe and legacy DSCP classification for the same parent class map.
- The child class map can contain only match precedence and match dscp commands.
- This feature is not supported in peering mode.

Configure Ingress Short-Pipe

This section details a sample configuration for the ingress short-pipe feature and another sample to configure classification for labeled and non-labeled packets under the same parent class.

Sample configuration to classify a packet that contains one MPLS label using the type-of-service (ToS) field of the IPv4 or IPv6 header (or the ingress short-pipe method):

```
class-map match-any in_pipe
 match mpls disposition class-map child_pipe
end-class-map
!
class-map match-any child_pipe
 match precedence 1
match dscp ipv4 af11
 end-class-map
!
class-map match-any ingress-business-high
match dscp af21 af22
end-class-map

class-map match-any ingress-business-low
```

```
match dscp af11 af12
end-class-map

policy-map ingress-classifier
class in_pipe
set traffic-class 5
set dscp af31
class ingress-business-high
set traffic-class 4
class ingress-business-low
set traffic-class 2
class class-default
set traffic-class 0
!
```

Note The **set dscp** option is available from Release 7.1.1 onwards.

You can configure classification for both labeled and non-labeled packets under the same parent class as in the following sample configuration. In this example, for MPLS labeled packets, DSCP configured under the child class is classified, while for non-labeled packets, DSCP/ToS configured in the **match dscp <value>** statement is classified.

DSCP value range is from 0 through 63. The range option is not supported. Up to 8 items per class are supported. Up to 64 **match dscp** values in total.

```
class-map match-any in_pipe
match mpls disposition class-map child_pipe  (labeled case)
match dscp af11 (non-labeled case)
end-class-map
!
class-map match-any child_pipe
match precedence 1
match dscp ipv4 af11
end-class-map
!
class-map match-any ingress-business-high
match dscp af21 af22
end-class-map

class-map match-any ingress-business-low
match dscp af11 af12
end-class-map

policy-map ingress-classifier
class in_pipe
set traffic-class 5
set dscp af31
class ingress-business-high
set traffic-class 4
class ingress-business-low
set traffic-class 2
class class-default
set traffic-class 0
!
```

Note The **set dscp** option is available from Release 7.1.1 onwards. A maximum of one set dscp command is supported per class-map.

Associated Commands

- match mpls disposition class-map

Selective Egress Policy-Based Queue Mapping

With selective egress policy-based queue mapping, you can combine traffic class (TC) maps in various permutations at the egress.

Note Modular chassis do not support this feature.

The primary aim of introducing the egress TC (traffic class) mapping is to classify the traffic in the ingress using a single policy and place the classified traffic into queues, by assigning the traffic classes. At the egress, you can support different groupings of TCs.

Based on different Service Level Agreements (SLAs) that each customer has signed up for, you can group some TCs into priority queues for real time (RT) traffic, other TCs into guaranteed bandwidth (BW) traffic, and the rest into best effort (BE) traffic delivery.

Let us consider an example where three customers have purchased these services, based on their requirements:

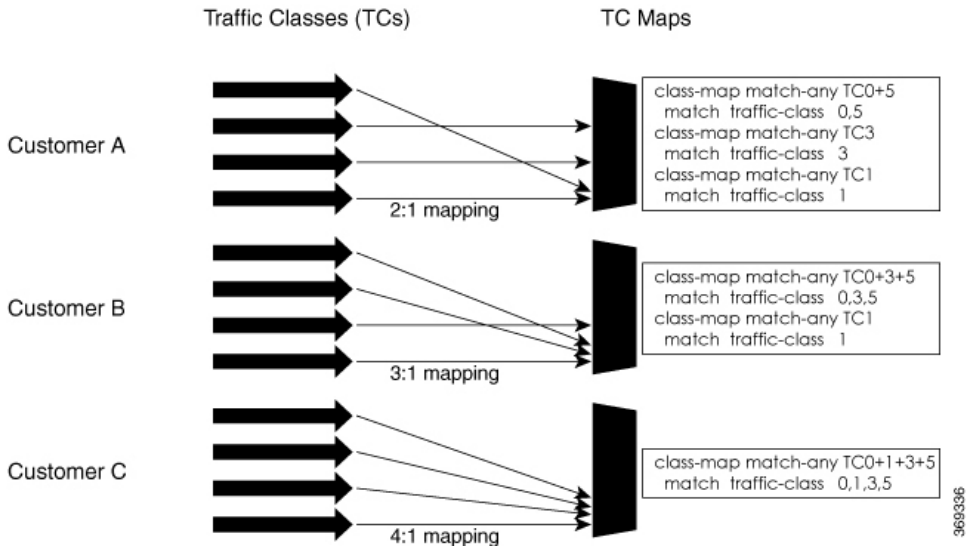
- Customer A - Requires RT traffic, reserved BW traffic and BE traffic delivery.
- Customer B – Requires reserved BW traffic and BE traffic delivery.
- Customer C – Needs only BE traffic delivery.

Using the selective egress policy-based queue mapping, you can create three profiles this way:

- Customer A – Priority queue RT traffic (TC1), Guaranteed BW traffic (TC3), Best effort traffic (TC0, TC5)
- Customer B – Guaranteed BW traffic (TC1), Best effort traffic (TC0, TC3, TC5)
- Customer C - Best effort traffic (TC0, TC1, TC3, TC5)

Using the egress TC-mapping, you can create three different profiles that you can use for each customer based on their SLAs with the provider.

Figure 1: Selective Egress Policy-Based Queue Mapping Helps Create Customer Profiles Based on Their SLAs



Restrictions and Other Important Points

- Ensure that you read these points before you configure the selective egress policy-based queue-mapping feature.
  - There can be only one TC (Traffic Class) mapped class to a PM (Policy Map).
  - You cannot use a TC that you used in a mapped class, in a non-mapped class under the same PM.
  - You can have a maximum of three unique TC mapped PMs or profiles per platform.
  - Every TC mapped class must include **traffic-class 0** in the range values.
  - The TC-mapping range is from 0 through 5.
  - When a TC-mapped class is present in a PM, the class default becomes a dummy class. This means that the class default statistics and QoS values are not applicable.
  - All the class default limitations apply to the TC-mapped class; for example, you cannot configure **priority** command under the TC mapped class.

Note A TC-mapped PM or profile is a PM that contains a TC-mapped class.

Example of a TC-mapped class:

match traffic-class 0 1 2 3

Example of a TC non-mapped class:

match traffic-class 1

Configure Selective Egress Policy-Based Queue Mapping

This section details a sample configuration for the selective egress policy-based queue-mapping feature and a use case to show how this feature works.

Sample configuration

```
class-map match-any <name>
 match traffic-class <value>
commit

policy-map tc_pmap
 class tc035
  shape average percent 1
 !
 class class-default
!
 end-policy-map
!
 class-map match-any tc035
match traffic-class 0 3 5
 end-class-map
!
```

Verification

Run the **show qos interface** and **show policy-map interface** commands.

When TC mapping class is present in a policy map, the class default does not have any values calculated.

show qos interface bundle-Ether 44 output sample

```
NOTE:- Configured values are displayed within parentheses
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           100000000 kbps
Policy Name:                   tc_pmap
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                               =   tc1

Level1 Class                               =   tc035

Level1 Class                               =   class-default

Interface HundredGigE0/0/0/30 Ifh 0xf000208 (Member) -- output policy
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           100000000 kbps
Policy Name:                   tc_pmap
VOQ Base:                      1264
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                               =   tc1
Egressq Queue ID                           =   1265 (LP queue)
Queue Max. BW.                             =   10063882 kbps (10 %)
Queue Min. BW.                             =   0 kbps (default)
Inverse Weight / Weight                    =   1 / (BWR not configured)
Guaranteed service rate                    =   10000000 kbps
TailDrop Threshold                         =   12517376 bytes / 10 ms (default)
WRED not configured for this class
```

```
Level1 Class                                =   tc035
Egressq Queue ID                            =   1264 (LP queue)
Queue Max. BW.                              =   1011732 kbps (1 %)
Queue Min. BW.                              =   0 kbps (default)
Inverse Weight / Weight                     =   1 / (BWR not configured)
Guaranteed service rate                     =   1000000 kbps
TailDrop Threshold                          =   1253376 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class                                =   class-default
Queue Max. BW.                              =   no max (default)
Queue Min. BW.                              =   0 kbps (default)
Inverse Weight / Weight                     =   0 / (BWR not configured)
```

show policy-map interface bundle-Ether 44 output sample

```
Bundle-Ether44 output: tc_pmap

Class tc1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :             429444/53823648             0
    Transmitted         :             429444/53823648             0
    Total Dropped       :                  0/0                    0
  Queueing statistics
    Queue ID                             : None (Bundle)
    Taildropped(packets/bytes)           : 0/0
Class tc035
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :            1288331/161470820            0
    Transmitted         :            1288331/161470820            0
    Total Dropped       :                  0/0                    0
  Queueing statistics
    Queue ID                             : None (Bundle)
    Taildropped(packets/bytes)           : 0/0
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0                    0
    Transmitted         :                  0/0                    0
    Total Dropped       :                  0/0                    0
  Queueing statistics
    Queue ID                             : None (Bundle)
    Taildropped(packets/bytes)           : 0/0
Policy Bag Stats time: 1557216940000  [Local Time: 05/07/19 08:15:40.000]
RP/0/RP0/CPU0:BB1#
```

Use Case

With the ingress traffic matching the same match criteria, you can group the egress traffic up to three unique TC mapped profiles. Using this feature, you can provide differentiated services to customers based on the SLAs they have signed up for.

In the example that follows, the ingress policy-map sets the ingress match criteria for the traffic class from 0 through 5. Based on the SLAs, you can group the TC values at the egress PM to deliver differentiated services.

After you group the TC values, you can apply specific egress actions under that class.

Ingress match:

```
class EXP1
  set traffic-class 1
!
class EXP2
  set traffic-class 2
```

```
!
class EXP3
  set traffic-class 3
!
class EXP4
  set traffic-class 4
!
class EXP5
  set traffic-class 5
!
class class-default
!
end-policy-map
!
```

Egress match:

Sample TC mapped class for policy-map PM1

```
class-map match-any TC2:1
match traffic-class 0 1
end-class-map
```

Sample TC mapped class for policy-map PM2

```
class-map match-any TC3:1
match traffic-class 0 1 2
end-class-map
```

Sample TC mapped class for policy-map PM3

```
class-map match-any TC6:1
match traffic-class 0 1 2 3 4 5
end-class-map
```

Configuring QoS Groups with an ACL

You can create QoS groups and configure ACLs to classify traffic into the groups based on a specified match condition. In this example, we match by the QoS group value (0-511).

Prerequisites

Before you can configure QoS groups with an ACL, the QoS peering profile must be enabled on the router or the line card. After enabling QoS peering, the router or line card must be reloaded, as shown in the following configuration.

Note If you enable QoS peering, the Source-based RTBH (S-RTBH) feature will not be supported due to hardware limitations related to copy engine availability.

Enabling QoS Peering Profile on the Router

Enter the global configuration mode and enable the QoS peering profile for the router as shown:

```
RP/0/RP0/CPU0:router(config)# hw-module profile qos ingress-model peering
RP/0/RP0/CPU0:router(config)# exit
RP/0/RP0/CPU0:router# reload
```

Enabling QoS Peering Profile on the Line Card

Enter the global configuration mode and enable the QoS peering profile for the line card as shown:

```
RP/0/RP0/CPU0:router(config)# hw-module profile qos ingress-model peering location 0/0/CPU0
RP/0/RP0/CPU0:router(config)# exit
RP/0/RP0/CPU0:router# reload location 0/0/CPU0
```

Configuration

Use the following set of configuration statements to configure an ACL with QoS groups.

```
/*
 Enter the global configuration mode, and configure an ACL with the required QoS groups.
*/
RP/0/RP0/CPU0:router# configure
RP/0/RP0/CPU0:router(config)# ipv4 access-list qos-acl
RP/0/RP0/CPU0:router(config-ipv4-acl)# 10 permit ipv4 host 5.0.0.1 any set qos-group 1
RP/0/RP0/CPU0:router(config-ipv4-acl)# 11 permit ipv4 host 6.0.0.1 any set qos-group 2
RP/0/RP0/CPU0:router(config-ipv4-acl)# 12 permit ipv4 host 7.0.0.1 any set qos-group 3
RP/0/RP0/CPU0:router(config-ipv4-acl)# 13 deny ipv4 any any


/* Create a policy map with the required classes.
In this example, we also create a default class for traffic that does not belong to any of
 the specified
classes. */
RP/0/RP0/CPU0:router(config)# policy-map qos-acl-map
RP/0/RP0/CPU0:router(config-pmap)# class qos1
RP/0/RP0/CPU0:router(config-pmap-c)# set dscp af43
RP/0/RP0/CPU0:router(config-pmap-c)# set traffic-class 2
RP/0/RP0/CPU0:router(config-pmap-c)# exit

RP/0/RP0/CPU0:router(config-pmap)# class qos2
RP/0/RP0/CPU0:router(config-pmap-c)# set precedence critical
RP/0/RP0/CPU0:router(config-pmap-c)# set traffic-class 7
RP/0/RP0/CPU0:router(config-pmap-c)# exit

RP/0/RP0/CPU0:router(config-pmap)# class qos3
RP/0/RP0/CPU0:router(config-pmap-c)# set precedence 2
RP/0/RP0/CPU0:router(config-pmap-c)# set traffic-class 2
RP/0/RP0/CPU0:router(config-pmap-c)# exit

RP/0/RP0/CPU0:router(config-pmap)# class qos4
RP/0/RP0/CPU0:router(config-pmap-c)# set traffic-class 4
RP/0/RP0/CPU0:router(config-pmap-c)# set dscp cs4
RP/0/RP0/CPU0:router(config-pmap-c)# exit

RP/0/RP0/CPU0:router(config-pmap)# class class-default
RP/0/RP0/CPU0:router(config-pmap-c)# police rate percent 20
RP/0/RP0/CPU0:router(config-pmap-c-police)# exit


/* Create the class maps for specifying the match conditions. */
RP/0/RP0/CPU0:router(config)# class-map match-any qos1
RP/0/RP0/CPU0:router(config-cmap)# match qos-group 1
RP/0/RP0/CPU0:router(config-cmap)# end-class-map

RP/0/RP0/CPU0:router(config)# class-map match-any qos2
RP/0/RP0/CPU0:router(config-cmap)#  match qos-group 2
RP/0/RP0/CPU0:router(config-cmap)# end-class-map

RP/0/RP0/CPU0:router(config)# class-map match-any qos3
RP/0/RP0/CPU0:router(config-cmap)# match qos-group 3
```

```
RP/0/RP0/CPU0:router(config-cmap)# end-class-map

RP/0/RP0/CPU0:router(config)# class-map match-any qos4
RP/0/RP0/CPU0:router(config-cmap)# match qos-group 4
RP/0/RP0/CPU0:router(config-cmap)# end-class-map


/* Apply the access list and the QoS map to the Gigabit interface, and commit your
configuration. */
RP/0/RP0/CPU0:router(config)# interface TenGigE0/0/0/1
RP/0/RP0/CPU0:router(config-if)# ipv4 address 12.0.0.1/24
RP/0/RP0/CPU0:router(config-if)# no shut
RP/0/RP0/CPU0:router(config-if)# service-policy input qos-acl-map
RP/0/RP0/CPU0:router

RP/0/RP0/CPU0:router(config-if)# commit
Tue Mar 28 10:23:34.106 IST

RP/0/0/CPU0:Mar 28 10:37:48.570 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : Interface
TenGigE0/0/0/1, changed state to Down
RP/0/0/CPU0:Mar 28 10:37:48.608 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : Interface
TenGigE0/0/0/1, changed state to Up

RP/0/RP0/CPU0:router(config-if)# exit
```

Running Configuration

Confirm your configuration.

```
RP/0/RP0/CPU0:router(config)# show run
Tue Mar 28 10:37:55.737 IST

Building configuration...
!! IOS XR Configuration 0.0.0

ipv4 access-list qos-acl
10 permit ipv4 host 5.0.1.1 any set qos-group 1
11 permit ipv4 host 6.0.1.1 any set qos-group 2
12 permit ipv4 host 7.0.1.1 any set qos-group 3
13 deny ipv4 any any

class-map match-any qos1
match qos-group 1
end-class-map
!
class-map match-any qos2
match qos-group 2
end-class-map
!
class-map match-any qos3
match qos-group 3
end-class-map
!
class-map match-any qos4
match qos-group 4
end-class-map
!

policy-map qos-acl-map
class qos1
  set dscp af43
  set traffic-class 2
```

```
!
class qos2
  set precedence critical
  set traffic-class 7
!
class qos3
  set precedence 2
  set traffic-class 2
!
class qos4
  set traffic-class 4
  set dscp cs4
!
class class-default
  police rate percent 20
  !
!
end-policy-map
!

interface TenGigE0/0/0/1
service-policy input qos-acl-map
ipv4 address 12.0.0.1 255.255.255.0
ipv4 access-group qos-acl ingress compress level 3

!
```

You have successfully configured an ACL with QoS groups.

Configuring an ACL with Fragment Match

Table 13: Feature History Table

Feature Name	Release Information	Feature Description
Additional Routers Supported for ACL with Fragment Match	Release 7.9.1	You can prevent malicious users from staging denial of service (DoS) attacks for non-initial IP by configuring an ACL with fragment match and specifying QoS match actions to rate-limit non-initial fragments for IPv4 traffic.

Usually, IP ACLs process non-fragmented packets and the first fragments of a packet using permit and deny actions. These packets may have Layer 3 and 4 information that the ACLs match for a permit or deny action. By default, however, ACLs permit noninitial fragments. This could lead to potential security issues with users with malicious intent using the noninitial fragments to launch denial of service (DoS) attacks.

With this feature, you can now to set QoS policies for noninitial fragment packets, thus having more granular control over noninitial IP fragments of a packet. Noninitial IP fragments have the fragment offset value non-zero. To know more about fragments, see the *IP Addresses and Services Configuration Guide for Cisco NCS 540 Series Routers*.

Restrictions and Guidelines

The following restrictions and guidelines apply while configuring an ACL with fragment match.

- From Cisco IOS XR Release 7.9.1 onwards, this feature is supported on all Cisco NCS 540 Series Routers. You can configure only IPv4 ACLs with fragment match on these systems. (See Configuring an ACL with Fragment Match, on page 61.)
- To enable IPv6 fragment classification support, configure **hw-module profile qos ipv6 short-l2qos-enable** or **hw-module profile qos ipv6 short**.
- IPv6 fragmentation is supported with only one Extension Header (EH).

Configuring an ACL with Fragment Match

For IPv4 and IPv6 ACLs

To configure an ACL with fragment match, you must:

1. Create IPv4 and IPv6 ACLs with fragment match.

Note To enable IPv6 fragment classification support, configure **hw-module profile qos ipv6 short-l2qos-enable** or **hw-module profile qos ipv6 short**.

2. Create two class maps, one for IPv4 and IPv6, and attach the respective ACLs to the class maps.
3. Create a policy map with these two class maps and set action.

```
Router(config)#ipv4 access-list v4_ace
Router(config-ipv4-acl)#permit ipv4 any any fragments
Router(config-ipv4-acl)#exit
Router(config)#
Router(config)#ipv6 access-list v6_ace
Router(config-ipv6-acl)#permit ipv6 any any fragments
Router(config-ipv6-acl)#exit
Router(config)#
Router(config)#class-map match-any v4_class
Router(config-cmap)#match access-group ipv4 v4_ace
Router(config-cmap)#exit
Router(config)#
Router(config)#class-map match-any v6_class
Router(config-cmap)#match access-group ipv6 v6_ace
Router(config-cmap)#exit
Router(config)#policy-map frag_policy
Router(config-pmap)#class v4_class
Router(config-pmap-c)#set traffic-class 3
Router(config-pmap-c)#police rate 100 mbps
Router(config-pmap-c-police)#exit
Router(config-pmap-c)#
Router(config-pmap-c)#class v6_class
Router(config-pmap-c)#police rate 150 mbps peak-rate 200 mbps
Router(config-pmap-c-police)#exit
Router(config-pmap-c)#exit
Router(config-pmap)#exit
```

Running Configuration

```
ipv4 access-list v4_ace
 permit ipv4 any any fragments
 exit
!
ipv6 access-list v6_ace
 permit ipv6 any any fragments
 exit
!
class-map match-any v4_class
 match access-group ipv4 v4_ace
 exit
!
class-map match-any v6_class
 match access-group ipv6 v6_ace
 exit

policy-map frag_policy
 class v4_class
  set traffic-class 3
  police rate 100 mbps
   exit
!
 class v6_class
  police rate 150 mbps peak-rate 200 mbps
```

Verification

Run the **show policy-map pmap-name frag_policy detail** command to confirm the ACL fragment matches and the **show qos int hundredGigE 0/5/0/2 input** command to confirm the policer details.

```
Router#show policy-map pmap-name frag_policy detail
ipv4 access-list v4_ace
10 permit ipv4 any any fragments

ipv6 access-list v6_ace
10 permit ipv6 any any fragments

class-map match-any v4_class
match access-group ipv4 v4_ace
 end-class-map
!
class-map match-any v6_class
match access-group ipv6 v6_ace
 end-class-map
!
policy-map frag_policy
class v4_class
  set traffic-class 3
  police rate 100 mbps
  !
 !
 class v6_class
  police rate 150 mbps peak-rate 200 mbps
  !
 !
 class class-default
!
 end-policy-map
!

Router#show qos int hundredGigE 0/5/0/2 input
```

```
NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/5/0/2 ifh 0xa000088  -- input policy
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           100000000 kbps
Policy Name:                   frag_policy
SPI Id:                        0x0
Accounting Type:               Layer2 (Include Layer 2 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                              =   v4_class
New traffic class                         =   3

Policer Bucket ID                         =   0x12
Policer Stats Handle                      =   0x0
Policer committed rate                    =   99609 kbps (100 mbits/sec)
Policer conform burst                     =   124672 bytes (default)

Level1 Class                              =   v6_class

Policer Bucket ID                         =   0x11
Policer Stats Handle                      =   0x0
Policer committed rate                    =   150390 kbps (150 mbits/sec)
Policer peak rate                         =   200195 kbps (200 mbits/sec)
Policer conform burst                     =   186624 bytes (default)
Policer exceed burst                      =   436096 bytes (default)

Level1 Class                              =   class-default

Default Policer Bucket ID                 =   0x10
Default Policer Stats Handle              =   0x0
Policer not configured for this class
```

For IPv4 ACLs

To configure an ACL with fragment match, you must:

1. Create an IPv4 ACL with fragment match.
2. Create a class map for IPv4 and attach the ACL to the class map.
3. Create a policy map with the class map and set action.

Note You can configure IPv4 ACLs with fragment match on Cisco NCS 540 Series Routers.

```
Router(config)#ipv4 access-list v4_ace
Router(config-ipv4-acl)#permit ipv4 any any fragments
Router(config-ipv4-acl)#exit
Router(config)#
Router(config)#class-map match-any v4_class
Router(config-cmap)#match access-group ipv4 v4_ace
Router(config-cmap)#exit
Router(config)#
Router(config)#policy-map frag_policy
Router(config-pmap)#class v4_class
Router(config-pmap-c)#set traffic-class 3
Router(config-pmap-c)#police rate 100 mbps
Router(config-pmap-c-police)#exit
Router(config-pmap-c)#exit
Router(config-pmap)#exit
```

Running Configuration

```
ipv4 access-list v4_ace
 permit ipv4 any any fragments
 exit

!
class-map match-any v4_class
 match access-group ipv4 v4_ace
 exit
!

policy-map frag_policy
 class v4_class
  set traffic-class 3
  police rate 100 mbps
   exit
!
```

Verification

Run the **show policy-map pmap-name frag_policy detail** command to confirm the ACL fragment matches and the **show qos int hundredGigE 0/5/0/2 input** command to confirm the policer details.

```
Router#show policy-map pmap-name frag_policy detail
ipv4 access-list v4_ace
10 permit ipv4 any any fragments


class-map match-any v4_class
match access-group ipv4 v4_ace
 end-class-map
!
!
policy-map frag_policy
class v4_class
  set traffic-class 3
  police rate 100 mbps
  !
 !
 class class-default
!
 end-policy-map
!

Router#show qos int hundredGigE 0/5/0/2 input
NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/5/0/2 ifh 0xa000088  -- input policy
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           100000000 kbps
Policy Name:                   frag_policy
SPI Id:                        0x0
Accounting Type:               Layer2 (Include Layer 2 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   v4_class
New traffic class                        =   3

Policer Bucket ID                        =   0x12
Policer Stats Handle                     =   0x0
Policer committed rate                   =   99609 kbps (100 mbits/sec)
Policer conform burst                    =   124672 bytes (default)
```

```
Level1 Class                            =   class-default

Default Policer Bucket ID               =   0x10
Default Policer Stats Handle            =   0x0
Policer not configured for this class
```

QoS Egress Marking and Queuing Using Dual Policy-Map

To achieve QoS Egress marking/queuing, the router utilizes the dual policy model on the Egress with independent policies for marking and queuing.

Egress marking can be achieved by applying a policy-map on the ingress interface by setting qos-group/discard-class. Then the qos-group which is set by the ingress policy-map is used by the egress-policy map along with DP (drop-precedence or discard class) value to remark the cos/dei bits of the outgoing L2 packet. Similarly Egress queuing can be achieved by applying a policy-map on the ingress interface by setting the traffic-class. Then the traffic-class is used by the egress-policy map to perform queuing actions.

Benefits

- This feature enables the users to make the marking decision based on the DP (drop precedence) field.
- In case of MPLS-to-Layer 2 traffic stream, the Layer 2 packet is within the MPLS data packet; therefore marking of the Layer 2 header is possible only at Egress after data transmission.
- In case of Egress rewrite operations, where the VLAN tags are modified or added, the cos or the dei fields can be marked with Egress marking.

QoS Egress Marking and Queueing can be summarized in the following three steps—

1. Configure a Ingress Policy-Map— classifying the incoming packet and setting the qos-group/discard-class or the traffic class.
2. Configure a Egress Policy-Map:
    - Configure Egress Marking Policy—
        - Create class-map to classify on qos-group/discard-class.
        - Create policy-map to mark cos/dei field in the L2 header.
    - Configure Egress Queuing Policy—
        - Create class-map to classify on traffic-class.
        - Create policy-map to perform the queuing actions (for example, bandwidth, shaping, priority).
3. Attaching the policies to the Interfaces.

Note While marking QinQ traffic, only outer dot1q header is effected and the inner header remains as is. However, in case of few rewrite operations where the new QinQ tags are added, the inner header is marked.

Example— Ingress Policy-Map Configuration:

```
/*Create class-map/*
Router#config
Router(config)#class-map match-any cos2
Router(config-cmap)#match cos 2
Router(config-cmap)#commit
Router(config)#class-map match-any cos3
Router(config-cmap)#match cos 3
Router(config-cmap)#commit
Router(config)#class-map match-any cos4
Router(config-cmap)#match cos 4
Router(config-cmap)#commit

/*Create classification policies*/
Router#config
Router(config)#policy-map ingress-classification
Route(config-pmap)#class cos2
Router(config-pmap-c)#set qos-group 1
Router(config-pmap-c)#set traffic-class 3
Router(config-pmap-c)#class cos3
Router(config-pmap-c)#set qos-group 2
Router(config-pmap-c)#set traffic-class 5
Router(config-pmap-c)#class cos4
Router(config-pmap-c)#set qos-group 3
Router(config-pmap-c)#set traffic-class 4
Router(config-pmap-c)#class class-default
Router(config-pmap-c)#set qos-group 7
Router(config-pmap-c)#set traffic-class 6
Router(config-pmap-c)#commit
```

Example— Egress Policy-Map Configuration:

```
*/Egress Marking Policy/*
Router#config
Router(config)#class-map match-any qos1
Router(config-cmap)#match qos-group 1
Router(config-cmap)#commit
Router(config)#class-map match-any qos2
Router(config-cmap)#match qos-group 2
Router(config-cmap)#commit
Router(config)#class-map match-any qos3
Router(config-cmap)#match qos-group 3
Router(config-cmap)#commit
Router#config
Router(config)#policy-map egress-marking
Route(config-pmap)#class qos1
Router(config-pmap-c)#set cos 1
Router(config-pmap-c)#class qos2
Router(config-pmap-c)#set cos 2
Router(config-pmap-c)#set dei 1
Router(config-pmap-c)#class qos3
Router(config-pmap-c)#set cos 3
Router(config-pmap-c)#class class-default
Router(config-pmap-c)#set cos 7
Router(config-pmap-c)#commit

*/Egress Queuing Policy/*
Router#config
Router(config)#class-map match-any tc3
Router(config-cmap)#match traffic-class 3
Router(config-cmap)#commit
Router(config)#class-map match-any tc4
```

```
Router(config-cmap)#match traffic-class 3
Router(config-cmap)#commit
Router(config)#class-map match-any tc5
Router(config-cmap)#match traffic-class 3
Router(config-cmap)#commit
Router#config
Router(config)#policy-map egress-queuing
Route(config-pmap)#class tc3
Router(config-pmap-c)#shape average 2 mbps
Router(config-pmap-c)#class tc4
Router(config-pmap-c)#shape average 5 mbps
Router(config-pmap-c)#class tc5
Router(config-pmap-c)#shape average 7 mbps
Router(config-pmap-c)#class class-default
Router(config-pmap-c)#commit
```

Example— Attaching the policies to the Interface

```
Router#config
Router(config)#interface tenGigE 0/0/0/1
Router(config-if)#service-policy input ingress-classification
Router(config-if)#service-policy output egress-marking
Router(config-if)#service-policy output egress-queuing
Router(config-if)#commit
```

Restrictions

- Statistics for marking policy is not supported, that is, the show policy-map interface command does not display any output.
- Statistics output is displayed only when the queuing policy is applied.
- Egress marking policy can classify only on qos-group/discard-class.
- Egress queueing policy can classify only on traffic-class.
- Egress marking policy can mark only the cos/dei field in L2 header.

Restrictions Specific to NCS 540 Variants

This is a tabular classification of NCS 540 series platform and related models, detailing their maximum supported policers and class-map entries. When enhanced mode is enabled, the number of supported policers is reduced by half compared to the standard mode.

Table 14: Policers and class maps support matrix

PID	Total Policers (Standard Mode)	Total Policers (Enhanced Mode)	Max Class-map Size	Number of Interfaces
• N540-28Z4C • N540-12Z20G • N540X-12Z16G • 540X-16Z4G8Q2C • N540X-16Z8Q2C • 540-FH-CSR • N540-FH-AGG • N540-FH-IP65	6976	3488	32	218
			16	434
			8	872
			4	1744
			2	3488
• N540X-2Z16G2Q • N540X-6Z18G-SYS • N540X-8Z16G-SYS • N540-6Z14S-SYS-D • N540-6Z18G-SYS	1536	768	32	48
			16	96
			8	192
			4	384
			2	768
N540-24Z8Q2C-SYS	6144	3072	32	192
			16	384
			8	768
			4	1536
			2	3072

Table 15: Feature History Table

Feature Name	Release Information	Feature Description
Support for 2 class-map per policy	Release 7.5.1	This feature enables you to configure 2 class maps per QoS policy. With a smaller class-map size supported, you can now apply QoS policies to more interfaces, thus increasing the scale numbers. This feature introduces the **class-map-size 2 hw-module** command.

Table 16: Feature History Table

Feature Name	Release Information	Feature Description
Increase in Number of Ingress QoS Policies Supported on Layer 2	Release 24.2.11	Layer 2 Virtual Private Network (L2VPN) now supports increase in number of ingress QoS policies on all the class-map sizes. Now you can attach the additional QoS policies to the interfaces, to manage, prioritize, and control the flow of traffic at ingress. This feature is supported on the following Cisco NCS 540 router variants: • N540X-6Z18G-SYS-A • N540X-6Z18G-SYS-D To enable the increased ingress QoS policies on all the class-map sizes in Layer 2, this feature introdces the following new command: • hw-module profile qos policer-scale l2-mode

Note The router has a single core, hence the per core scale is applicable.

Note If you apply an ingress policy map to a bundle that has bundle members only from a single core of an NPU, the QoS resources are consumed on both cores of that NPU.

Example: For Default Configuration, which is Normal (2 counter mode) QoS Mode & 32 Class Map-Size, you can configure 191 interfaces with Ingress Policy per core.

Other restrictions to follow:

- In Egress QoS marking, you cannot change the traffic class or remark the performance measurement packets on Layer 3 with hardware timestamping.
- If you have a **set traffic class** statement explicitly configured in the ingress service policy, it's mandatory to have a corresponding **match traffic class** on egress for the traffic to be correctly matched and the stats to be accounted in **show policy-map interface <> output** command. To match the ingress traffic to the egress class-default, traffic class should be set to 0 on ingress.

- If you have a **set traffic class that is configured** in ingress service policy, and no corresponding **match traffic class** on egress, the traffic won't go to class default and the stats for this traffic flow won't be seen in **show policy-map interface <> output** command.
- If you don't have any **set traffic class** statement in ingress, then traffic will hit the default-class on egress.
- If you have a **set discard-class** statement configured in the ingress service policy, it's mandatory to have a corresponding **match discard-class** on egress for the traffic to be correctly matched and the stats to be accounted in **show policy-map interface <> output** command.
- If you have a **set discard-class** statement configured in the ingress service policy and don't have a corresponding **match discard-class** on egress, the traffic won't hit the class-default and the stats for this flow won't be accounted in **show policy-map interface <> output** command.
- The system doesn't support class-map size on peering mode.
- Depending on the packet size, the traffic shaped value for low shaper rates, such as 10mbps, have greater deviation than 5% of tolerance from the shaper value. For higher shaper rates, the deviation is within the limit of 5% of tolerance from the shaper value for all packet sizes.

Restrictions for Peering QoS Profile

- After enabling the QoS peering feature using the **hw-module profile qos ingress-model peering** command, you can set the Layer 2 class of service (CoS) or drop eligible indicator (DEI) values at the egress using the **set cos** or **set dei** commands, respectively. However, at the egress, ensure that you don't set the MPLS experimental imposition (EXP) values (using the **set mpls experimental imposition** command). Otherwise, when committing the policy map with these configurations at the egress, you will encounter an error. This error occurs because the internal fields required for egress EXP marking are not available with peering enabled.
- **explicit set discard-class** statement isn't supported.
- This feature is supported only on L3 interfaces and is limited to 1000 L3 interfaces per system.
- **set mpls exp topmost** statement isn't supported within QoS in peering mode.
- **access group** statement isn't supported.
- (Only in Release 6.2.x and Release 6.3.x) **set mpls exp imposition** statement isn't supported on ingress interface.
- 2-Level ingress policers isn't supported.
- (From Release 6.5.x) Egress H-QOS with peering profile support is enabled, but ingress H-QOS with peering profile isn't supported.
- Depending on the packet size, the traffic shaped value for low shaper rates, such as 10mbps, have greater deviation than 5% of tolerance from the shaper value. For higher shaper rates, the deviation is within the limit of 5% of tolerance from the shaper value for all packet sizes.

Restrictions for QoS on BVI

- The system doesn't support the egress policy on Bridge-Group Virtual Interface (BVI), but BVI (CoS, DEI) marking is supported by applying the policy to its corresponding Layer 2 interface, which is part of the same bridge domain.

- If you apply L3 ingress QoS policy on L2 interface, which is a part of the same bridge-domain as BVI, the classification might not work if packets are destined to the BVI MAC address.
- If a QoS policy is attached to BVI, the policy is inherited by the L2 interfaces, which are part of the same bridge-domain. Hence, any other policy can't be applied on the L2 interfaces. Similarly, if a QoS policy is attached to any of the L2 interfaces, any QoS policy can't be applied on the BVI, which is part of the same bridge-domain.
- In the two-pass forwarding model for packets from Layer 2 to Layer 3 over BVI, where Layer 2 and Layer 3 forwarding are split across two paths and packet processing happens in two cycles, you can apply separate QoS policies for Layer 2 and BVI interfaces.

Restrictions for Egress Drop Action

- A maximum of 8 interfaces can have the drop action configured and a maximum of 8 classes in any single policy can have the drop action.
- A drop action in any particular class can't be combined with other actions.
- Drop action in a policy applied on the main interface isn't inherited onto sub-interfaces.
- Match condition for drop action PM can only be based on qos-group, discard class based match isn't supported.

Restrictions for egress QoS policy scale per line card NPU

Before applying egress QoS service policies, verify that the maximum supported number of egress policies per line card NPU core is not exceeded on the system. If the configured number of egress policies exceeds the supported limit for the platform, the configuration fails at commit time, and the router reports a VOQ resource exhaustion error.

Table 17: Supported egress QoS policy scale (per NPU core)

Platform	Maximum supported egress QoS policies (per NPU core)
NCS 5516	38
NCS 5508	77
NCS 5504	154

In-Place Policy Modification

The In-Place policy modification feature allows you to modify a QoS policy even when the QoS policy is attached to one or more interfaces. A modified policy is subjected to the same checks that a new policy is subject to when it is bound to an interface. If the policy-modification is successful, the modified policy takes effect on all the interfaces to which the policy is attached. However, if the policy modification fails on any one of the interfaces, an automatic rollback is initiated to ensure that the pre-modification policy is in effect on all the interfaces.

You can also modify any class map used in the policy map. The changes made to the class map take effect on all the interfaces to which the policy is attached.

Note

- The QoS statistics for the policy that is attached to an interface are lost (reset to 0) when the policy is modified.
- When a QoS policy attached to an interface is modified, there might not be any policy in effect on the interfaces in which the modified policy is used for a short period of time.
- The system does not support the show policy-map statistics for marking policies.
- An in-place modification of an ACL does not reset the policy-map statistics counter.

Note

- For QOS EXP-Egress marking applied on a Layer 3 interface on Cisco routers, there is a limit of two unique policy-maps per NPU. When the maximum limit for policy-maps is reached and you try to modify a policy-map which is shared between different interfaces, you may get an error.
- For QOS egress marking (CoS, DEI) applied on a Layer 2 interface or on L3 sub-interface, there is a limit of 13 unique policy-maps per NPU. When the maximum limit for policy-maps is reached and you try to modify a policy-map which is shared between different interfaces, you may get an error.

Verification

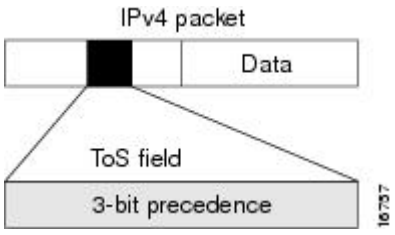
If unrecoverable errors occur during in-place policy modification, the policy is put into an inconsistent state on target interfaces. No new configuration is possible until the configuration session is unblocked. It is recommended to remove the policy from the interface, check the modified policy and then re-apply accordingly.

References for Modular QoS Service Packet Classification

Specification of the CoS for a Packet with IP Precedence

Use of IP precedence allows you to specify the CoS for a packet. You can create differentiated service by setting precedence levels on incoming traffic and using them in combination with the QoS queuing features. So that, each subsequent network element can provide service based on the determined policy. IP precedence is usually deployed as close to the edge of the network or administrative domain as possible. This allows the rest of the core or backbone to implement QoS based on precedence.

Figure 2: IPv4 Packet Type of Service Field



You can use the three precedence bits in the type-of-service (ToS) field of the IPv4 header for this purpose. Using the ToS bits, you can define up to eight classes of service. Other features configured throughout the network can then use these bits to determine how to treat the packet in regard to the ToS to grant it. These other QoS features can assign appropriate traffic-handling policies, including congestion management strategy and bandwidth allocation. For example, queuing features such as LLQ can use the IP precedence setting of the packet to prioritize traffic.

IP Precedence Bits Used to Classify Packets

Use the three IP precedence bits in the ToS field of the IP header to specify the CoS assignment for each packet. You can partition traffic into a maximum of eight classes and then use policy maps to define network policies in terms of congestion handling and bandwidth allocation for each class.

Each precedence corresponds to a name. IP precedence bit settings 6 and 7 are reserved for network control information, such as routing updates. These names are defined in RFC 791.

IP Precedence Value Settings

By default, the routers leave the IP precedence value untouched. This preserves the precedence value set in the header and allows all internal network devices to provide service based on the IP precedence setting. This policy follows the standard approach stipulating that network traffic should be sorted into various types of service at the edge of the network and that those types of service should be implemented in the core of the network. Routers in the core of the network can then use the precedence bits to determine the order of transmission, the likelihood of packet drop, and so on.

Because traffic coming into your network can have the precedence set by outside devices, we recommend that you reset the precedence for all traffic entering your network. By controlling IP precedence settings, you prohibit users that have already set the IP precedence from acquiring better service for their traffic simply by setting a high precedence for all of their packets.

The class-based unconditional packet marking and LLQ features can use the IP precedence bits.

IP Precedence Compared to IP DSCP Marking

If you need to mark packets in your network and all your devices support IP DSCP marking, use the IP DSCP marking to mark your packets because the IP DSCP markings provide more unconditional packet marking options. If marking by IP DSCP is undesirable, however, or if you are unsure if the devices in your network support IP DSCP values, use the IP precedence value to mark your packets. The IP precedence value is likely to be supported by all devices in the network.

You can set up to 8 different IP precedence markings and 64 different IP DSCP markings.

Conditional Marking of MPLS Experimental bits for L3VPN Traffic

The conditional marking of MPLS experimental bits is achieved for Layer 3 Virtual Private Network (L3VPN) traffic by applying a combination of ingress and egress policy-maps on the Provider Edge (PE) router. In the ingress policy-map, the qos-group or discard-class is set either based on the result of the policing action or implicitly. The egress policy-map matches on qos-group or discard-class and sets the mpls experiment bits to the corresponding value.

This feature is supported on both IPv4 and IPv6 traffic in the L3VPN network. Conditional marking can be used to mark the MPLS experimental bits differently for in-contract and out-of-contract packets. In-contract packets are the confirmed packets with the color green and discard-class set to 0. Out-of-contract packets are the packets which have exceeded the limit and have the color yellow and discard-class set to 1.

Conditional marking of MPLS experimental bits for L3VPN traffic is supported on both physical and bundle main interfaces as well as sub-interfaces.

Restrictions for Conditional Marking of MPLS Experimental bits on L3VPN

1. In the case of two PE routers connected back-to-back and the only label that the traffic between the routers have is the BGP label, then the explicit null label should be configured.
2. A maximum of three policy-maps which perform conditional marking of MPLS experimental bits can be configured per Network Processor Unit (NPU) of the Cisco NCS 540 Series Routers.
3. In the ingress policy-map if qos-group is being set for the incoming traffic packets, then setting of dscp and mpls experimental bits will not work.
4. Both the ingress and egress policy-maps must be applied in order to attain the expected behaviour. If either one of them is not applied then it may lead to undefined behaviour.
5. If the egress policy-map does not match on qos-group or discard-class and set the mpls experiment bits to the required value, then the mpls experimental bits will be set to a value of zero, by default.

Conditional Marking of MPLS Experimental bits for L2VPN Traffic

Conditional marking of MPLS EXP bits is supported on Virtual Private Wire Service (VPWS), Virtual Private LAN Service (VPLS) and Ethernet Virtual Private Network (EVPN) traffic in the L2VPN network.

The conditional marking of MPLS experimental bits is achieved for Layer 2 Virtual Private Network (L2VPN) traffic by applying a combination of ingress and egress policy-maps on the Provider Edge (PE) router. In the ingress policy-map, the qos-group or discard-class is set either based on the result of the policing action or implicitly. The egress policy-map matches on qos-group or on a combination of qos-group and discard-class and sets the mpls experiment bits to the corresponding value.

Conditional marking can be used to mark the MPLS experimental bits differently for in-contract and out-of-contract packets. In-contract packets are the confirmed packets with the color green and discard-class set to 0. Out-of-contract packets are the packets which have exceeded the limit and have the color yellow and discard-class set to 1.

Conditional marking of MPLS experimental bits for L2VPN traffic is supported on both physical and bundle main interfaces as well as sub-interfaces.

Restrictions for Conditional Marking of MPLS Experimental bits on L2VPN

1. Conditional marking of MPLS EXP bits is not supported for broadcast, unknown unicast and multicast (BUM) traffic.
2. In the case of two PE routers connected back-to-back and the only label that the traffic between the routers have is the BGP label, then the explicit null label should be configured.
3. A maximum of two policy-maps which perform conditional marking of MPLS experimental bits can be configured per Network Processor Unit (NPU) of the Cisco NCS 540 Series Routers. However, the same policy can be applied on multiple interfaces on the same NPU.

4. In the ingress policy-map if qos-group is being set for the incoming traffic packets, then setting of dscp and mpls experimental bits will not work.
5. Both the ingress and egress policy-maps must be applied in order to attain the expected behaviour. If either one of them is not applied then it may lead to undefined behaviour.
6. If the egress policy-map does not match on qos-group or discard-class and set the mpls experiment bits to the required value, then the mpls experimental bits will be set to a value of zero, by default.

QoS IP DSCP Preservation for SR-TE

Table 18: Feature History Table

Feature Name	Release Information	Feature Description
QoS IP DSCP Preservation for IPv6 SR-TE	Release 7.10.1	This release introduces the functionality to preserve IP DSCP markings for IPv6 SR-TE traffic and covers the following scenarios: • **For two or less than two topmost or imposition labels:** when you set the MPLS experimental bits (EXP) values (also called Traffic Class values), the IP DSCP markings are now preserved by default in the ingress policies when the MPLS labels are pushed into the packet. • **For more than three imposition labels:** you must enable this functionality to preserve IP DSCP markings. With preservation, traffic with IPv6 packets with DSCP marking for priority, flows as intended and there's no drop in traffic because of incorrect or missing labels. In previous releases, irrespective of the number of MPLS labels, when the EXP values were copied into the packet header during imposition, even the IP DSCP markings for IPv6 traffic were modified. This modification resulted in traffic drops at the next-hop routers in SR-TE tunnels. This feature introduces a new keyword, **v6uc-enable**, in the **hw-module profile mpls-ext-dscp-preserve** command.

Feature Name	Release Information	Feature Description
QoS IP DSCP Preservation for SR-TE	Release 7.7.1	In terms of preserving IP DSCP markings, this release covers two scenarios for SR-TE traffic: • **For two or less than two topmost or imposition labels:** when you set the MPLS experimental bits (EXP) values (also called Traffic Class values), the IP DSCP markings are now preserved by default in the ingress policies when the MPLS labels are pushed into the packet. • **For more than three imposition labels:** you must enable this functionality to preserve IP DSCP markings. With preservation, traffic with IP packets with DSCP marking for priority, flows as intended and there's no drop in traffic because of incorrect or missing labels. In previous releases, irrespective of the number of MPLS labels, when the EXP values were copied into the packet header during imposition, even the IP DSCP markings were modified. This modification resulted in traffic drops at the next-hop routers in SR-TE tunnels. This feature introduces the **hw-module profile mpls-ext-dscp-preserve** command.

A typical ingress scenario for SR-TE traffic involves:

1. Specifying DSCP values as match criteria for a class map.
2. Identifying specific three-bit experimental bits (EXP) field values (also called Traffic Class or TC values, as per RFC 5462) in the topmost MPLS label as match criteria for the same class map.

When the MPLS labels are pushed (also called imposition) into the packet, the EXP values are copied into the EXP field in the packet header. At this stage, for SR-TE traffic, even the DSCP values that you set initially are modified, which could cause traffic drops at the next-hop router.

It is this behavior that the QoS DSCP preservation for SR-TE functionality mitigates. To enable this functionality, configure the **hw-module profile mpls-ext-dscp-preserve** command. This configuration ensures that the IP DSCP values are preserved for SR-TE traffic.

Note Ensure that you restart the router for the configuration to take effect.

Here's a quick look at what the QoS DSCP preservation for SR-TE functionality achieves.

Table 19: QoS DSCP Preservation for SR-TE and TI-LFA Scenarios

Scenario	What was happening in previous releases?	What's happening now?
SR-TE cases	Irrespective of the number of MPLS labels (1, 2, … 12) for imposition, the push action modified IP DSCP values.	• IP DSCP values are preserved by default for two or less than two MPLS labels. • IP DSCP values are preserved for three or more labels only when you configure the **hw-module profile mpls-ext-dscp-preserve** command.
SR Topology-Independent Loop-Free Alternate (TI-LFA) cases	For three MPLS labels for imposition, the push action modified IP DSCP values.	Irrespective of the number of MPLS labels, IP DSCP values are preserved when you configure the **hw-module profile mpls-ext-dscp-preserve** command.

Guidelines and Limitations for QoS IP DSCP Preservation for SR-TE

- Disabling the propagation of IP Time-To-Live (TTL) to and from the MPLS header (using the **mpls ip-ttl-propagate** command) doesn't help preserve IP DSCP values for SR-TE traffic. This is because the IP TTL acts on the egress, while the DSCP preservation for SR-TE is for the ingress traffic.
- When you use the **on-demand steering** command to disable labeled services for SR-TE policies, your router supports 12 MPLS transport labels by default. With the QoS DSCP preservation for SR-TE functionality enabled, this default value changes to ten MPLS transport labels.
- By default, SR-TE (with steering services enabled) supports 2 service labels and 9 transport labels. This support changes to 2 service labels and 8 transport labels when you configure **hw-module profile mpls-ext-dscp-preserve**.
- This functionality doesn’t work for SR-TE traffic associated with Binding Single-ID (BSID) policy paths. This is because packets with BSID labels get terminated, and new SR-TE labels get pushed, modifying the IP DSCP values. Thus, for BSID traffic (labeled traffic), disabling the propagation of IP TTL (using **mpls ip-ttl-propagate**) or configuring the **hw-module profile mpls-ext-dscp-preserve** command to enable QoS DSCP preservation for SR-TE doesn't help in IP DSCP preservation.

- This functionality isn't applicable for routers that have line cards installed and operating in native and compatibility modes. For these routers, DSCP preservation is by default, irrespective of the number of labels for both SR TI-LFA and SR-TE cases.
- The **hw-module profile sr-policy v6-null-label-autopush** and **hw-module profile mpls-ext-dscp-preserve** commands are mutually exclusive.
- This feature is supported on the following variants of Cisco Network Convergence System 540 Series Routers:
  - N540-24Z8Q2C-M
  - N540X-ACC-SYS
  - N540-ACC-SYS
  - N540-28Z4C-SYS
  - N540X-12Z16G-SYS-A/D
  - N540-28Z4C-SYS-A/D
  - N540-12Z20G-SYS-A/D
  - N540X-16Z4G8Q2C-A/D

Configuration for QoS IP DSCP Preservation for SR-TE

Configuration Example

To configure QoS IP DSCP preservation for SR-TE:

1. Match the DSCP value for a class map. (See **match dscp** for command details.)

```
Router(config)#class-map
Router(config)#class-map match-any  exp_policy
Router(config-cmap)#match dscp cs2
Router(config-cmap)#end-class-map
Router(config)#
```

2. Set the MPLS EXP value for the same map. (See **set mpls experimental** for command details.)

```
Router(config)#policy-map exp_policy
Router(config-pmap)#class exp_policy
Router(config-pmap-c)#set mpls experimental imposition 7
Router(config-pmap-c)#set traffic-class 7
Router(config-pmap-c)#commit
```

3. Apply the policy on the ingress interface.

```
Router(config)#interface TenGigE0/0/0/0
Router(config-if)#service-policy input exp_policy
Router(config-if)#ipv4 address 192.0.2.1 255.255.255.0
Router(config-if)#commit
```

4. Nexthop reachability for ingress traffic is through SR-TE Tunnels. To configure SR-TE policies, see *Configure SR-TE Policies* in the *Segment Routing Configuration Guide for Cisco NCS 540 Series Routers*.

5. Configure the DSCP preservation for IPv4 or IPv6 SR-TE traffic and reload the chassis for the configuration to take effect.

```
Router#config
Router(config)#hw profile mpls-ext-dscp-preserve v4uc-enable
or
Router(config)#hw profile mpls-ext-dscp-preserve v6uc-enable
Router(config)#commit
Router(config)#exit
Router#reload location all
```

Running Configuration for IPv4 SR-TE Traffic

The following is the running configuration for DSCP preservation for IPv4 SR-TE traffic.

```
class-map
class-map match-any  exp_policy
 match dscp cs2
 end-class-map
!

policy-map
policy-map exp_policy
 class exp_policy
 set mpls experimental imposition 7
 set traffic-class 7

!

interface TenGigE0/0/0/0
service-policy input exp_policy
ipv4 address 192.0.2.1 255.255.255.0
!

 hw profile mpls-ext-dscp-preserve v4uc-enable
!
```

Running Configuration for IPv6 SR-TE Traffic

The following is the running configuration for DSCP preservation for IPv6 SR-TE traffic.

```
Class-map
Router#show run class-map
Tue Apr 18 20:30:51.077 UTC
class-map match-any dscp-preserve-cm
match dscp ipv6 0
 end-class-map
!

Policy-map
Router#show run policy-map
Tue Apr 18 20:31:08.077 UTC
policy-map dscp-preserve-pm
class dscp-preserve-cm
  set dscp 3
  set mpls experimental imposition 4
!
 class class-default
!
 end-policy-map
!
```

```
Policy-map attached to ingress interface:

interface HundredGigE0/0/0/10
service-policy input dscp-preserve-pm
ipv4 address 103.1.1.1 255.255.255.248
ipv6 address 2002::6701:101/125
!
```

Verification for IPv4 SR-TE Traffic

Verify that you have successfully configured DSCP preservation for IPv4 SR-TE traffic

Run the command **show controller fia diagshell 0 "config" location 0/0/cpu0 | i dscp** and in the output, verify that **custom_feature_dscp_preserve_with_extend_label** equals 1. The value is 1 only when the **hw profile mpls-ext-dscp-preserve v4uc-enable** command takes effect after router reload.

```
Router#show controller fia diagshell 0 "config" location 0/0/cpu0 | inc dscp_preserve
    custom_feature_dscp_preserve_with_extend_label_v6=1
```

Verification for IPv6 SR-TE Traffic

Verify that you have successfully configured DSCP preservation for IPv6 SR-TE traffic

Run the command **show controller fia diagshell 0 "config" location 0/0/cpu0 | inc dscp_preserve** and in the output, verify that **custom_feature_dscp_preserve_with_extend_label_v6** equals 1. The value is 1 only when the **hw profile mpls-ext-dscp-preserve v6uc-enable** command takes effect after router reload.

```
show controllers fia diagshell 0 "config show" location 0/1/CPU0 | include "dscp_preserve"
Sat Jun 24 01:25:13.883 UTC
    custom_feature_dscp_preserve_with_extend_label_v6=1
    custom_feature_dscp_preserve_outlif_bits_mask=3
    custom_feature_dscp_preserve_with_extend_label=1
```

Policy-map for conditional marking of incoming traffic

The incoming packets on the Power Edge router are classified based on the ingress policy-map and these actions are taken.

- Set qos-group
- Discard class or drop precedence is set implicitly or as a result of a policing action.
- Set traffic class
- Packets that violate the configured policer are dropped in the ingress processing itself.

Running Configuration:

```
class-map af11
   match cos 1
!

policy-map ingress
 class af11
  police rate percent 10 peak-rate percent 20
  !
  set qos-group 1
  set Traffic-class 3
 !
```

```
 class class-default
 !
 end-policy-map
!
```

Policy-map for conditional marking of outgoing MPLS traffic

The ingress packet undergoes MPLS encapsulation during the egress processing in the PE router which performs the label imposition. The MPLS experimental bits are marked on the basis of egress policy-map which performs the following actions:

- Match on qos-group or discard class or both
- Set the MPLS experimental bits based on the match criteria

Running Configuration:

```
class-map match-all qos-group2_0
   match qos-group 2
   match discard-class 0

policy-map egress-marking
 class qos-group2_0 # This class matches on qos-group 2 and discard-class 0
  set mpls experimental imposition 1
 !
 class class-default
 !
 end-policy-map
!
policy-map Egress-Queuing
 class Traffic-class3
  shape average 500 mbps
!
 class class-default
!
end-policy-map
!
```

Conditional Marking of MPLS Experimental Bits for EVPN-VPWS Single-Homing Services

Table 20: Feature History Table

Feature Name	Release Information	Feature Description

Conditional Marking of MPLS Experimental Bits for EVPN-VPWS Single-Homing Services	Release 7.3.1	This feature enables you to differentiate traffic in the MPLS forwarding domain and manage traffic from ingress PE to egress PE based on the MPLS EXP bit of the MPLS header. This feature is supported only for EVPN-VPWS single-homing services, and not supported for EVPN-VPWS multi-homing services.

The conditional marking of MPLS experimental bits is achieved for EVPN-VPWS single-homing services by applying a combination of ingress and egress policy-maps on the provider edge (PE) router. In the ingress policy-map, the qos-group or discard-class is set either based on the result of the policing action or implicitly. The egress policy-map matches on qos-group or on a combination of qos-group and discard-class and sets the MPLS experiment bits to the corresponding value.

Conditional marking can be used to mark the MPLS experimental bits differently for in-contract and out-of-contract packets. In-contract packets are the confirmed packets with the color green and discard-class set to 0. Out-of-contract packets are the packets that have exceeded the limit and have the color yellow and discard-class set to 1.

Conditional marking of MPLS experimental bits for EVPN-VPWS single-homing services are supported on both physical and bundle main interfaces as well as sub-interfaces.

MPLS EXP Marking for EVPN Multi-Homed Services

Table 21: Feature History Table

Feature Name	Release Information	Feature Description
MPLS EXP Marking for EVPN Multi-Homed Services	Release 7.6.1	You can now differentiate traffic in the MPLS forwarding domain and manage traffic from ingress PE to egress PE based on the MPLS EXP bit of the MPLS header for EVPN multi-home scenarios.

Configuration

- The ingress policing is applied on the UNI interface. It is with set qos-group and set traffic class.
- The marking policy is applied at the core facing NNI interface.
- MPLS EXP imposition is marked while packets egress from NNI Interface.

Running Configuration

```
interface TenGigE0/0/0/2.203 l2transport  => This is UNI
encapsulation dot1q 203
service-policy input pol50-100

interface TenGigE0/0/0/10  ===============> This is the core NNI
description *** CORE IF  ***
cdp
service-policy input in_mpls
service-policy output eg_mark
```

```
ipv4 address 192.18.44.18 255.255.255.0
ipv6 address 2005:18:44::18/48
lldp
  enable
!
monitor-session test ethernet direction tx-only port-level
!
load-interval 30
!
l2vpn
xconnect group 203
  p2p 203
   interface TenGigE0/0/0/2.203
   neighbor evpn evi 1 service 203

policy-map pol50-100
class class-default
  set traffic-class 2
  set qos-group 4
  police rate 50 mbps peak-rate 100 mbps
  !
!
end-policy-map
!

policy-map eg_mark
class qg4dc0
  set mpls experimental imposition 2
!
class qg4dc1
 set mpls experimental imposition 3
!
class class-default
!
end-policy-map
!
class-map match-all qg4dc0
match qos-group 4
match discard-class 0
end-class-map
!

class-map match-all qg4dc1
match qos-group 4
match discard-class 1
end-class-map
```

Verification

Verify that you have configured conditional marking of MPLS experimental bits for EVPN-VPWS single-homing services successfully.

```
Router#show qos int tenGigE 0/0/0/2.101 input
NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/2.101 ifh 0x41da  -- input policy
NPU Id:                        0
Total number of classes:       1
Interface Bandwidth:           10000000 kbps
Policy Name:                   pol50-100
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   class-default
New traffic class                        =   2
```

```
New qos group                             =   4

Policer Bucket ID                         =   0x18
Policer Stats Handle                      =   0x0
Policer committed rate                    =   49219 kbps (50 mbits/sec)
Policer peak rate                         =   98438 kbps (100 mbits/sec)
Policer conform burst                     =   62336 bytes (default)
Policer exceed burst                      =   187008 bytes (default)
------------------------------------------------------------------------------

Router#show qos int tenGigE 0/0/0/10 output
Tue Sep  1 04:18:27.508 UTC
NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/10 ifh 0xe0  -- output policy
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           10000000 kbps
Policy Name:                   eg_mark
VOQ Base:                      0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                              =   qg4dc0
New imposition exp                        =   2
Queue Max. BW.                            =   no max (default)
Queue Min. BW.                            =   0 kbps (default)
Inverse Weight / Weight                   =   0 / (BWR not configured)

Level1 Class                              =   qg4dc1
New imposition exp                        =   3
Queue Max. BW.                            =   no max (default)
Queue Min. BW.                            =   0 kbps (default)
Inverse Weight / Weight                   =   0 / (BWR not configured)

Level1 Class                              =   class-default
Queue Max. BW.                            =   no max (default)
Queue Min. BW.                            =   0 kbps (default)
Inverse Weight / Weight                   =   0 / (BWR not configured)
------------------------------------------------------------------------------
```

QPPB

QoS Policy Propagation via BGP (QPPB) is a mechanism that allows propagation of quality of service (QoS) policy and classification by the sending party that is based on the following:

- Access lists
- Community lists
- Autonomous system paths in the Border Gateway Protocol (BGP)

Thus, helps in classification that is based on the destination address instead of the source address.

QoS policies that differentiate between different types of traffic are defined for a single enterprise network. For instance, one enterprise may want to treat important web traffic, not-important web traffic, and all other data traffic as three different classes. And thereafter, use the different classes for the voice and video traffic.

Hence, QPPB is introduced to overcome the following problems:

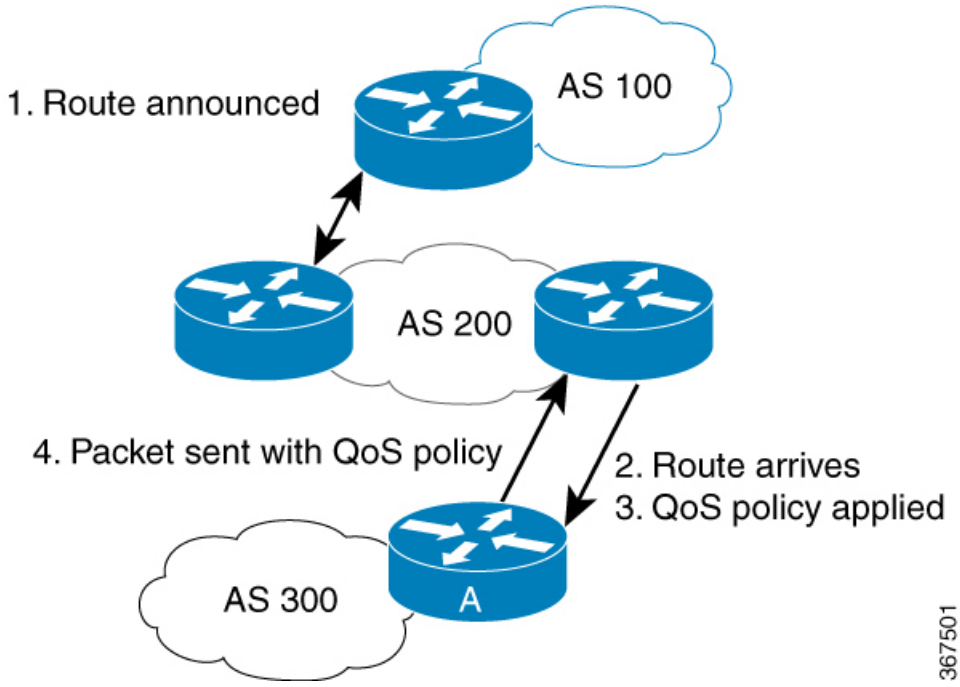
- The administrative challenges of classifying that is based on ACLs.
- The administrative problems of just listing the networks that need premium services.

QPPB allows marking of packets that are based on QoS group value associated with a Border Gateway Protocol (BGP) route.

Benefits of QPPB

- QPPB provides an IP prefix-based QoS capability.
- Traffic to IP addresses that have specific IP prefixes can be prioritized above other IP addresses.
- IP prefixes of interest are tagged through the control plane that uses common BGP route-map techniques, including the community attribute.
- Traffic to the tagged BGP prefixes is then classified and prioritized via the data forwarding plane by using the IOS-XR MQC (Modular QoS CLI) mechanisms, such as re-marking.
- QPPB provides the glue between the BGP control plane and the IP data forwarding plane in support of IP prefix-based QoS.
- BGP configuration within QPPB uses a table map to match specific prefixes learned through BGP neighbors, and then sets the router's local QoS Group variable maintained within the Forwarding Information Base (FIB) for those specific prefixes.
- The router supports a subset of full QPPB options - only IP destination prefix mode on input policy is supported.

Figure 3: Sample Scenario



Router A learns routes from AS 200 and AS 100. QoS policy is applied to any ingress interface of Router A to match the defined route maps with destination prefixes of incoming packets. Matching packets on Router A to AS 200 or AS 100 are sent with the appropriate QoS policy from Router A.

BGP maintains a scalable database of destination prefixes, QPPB, by using BGP table maps. BGP adds the ability to map a qos-group value to desired IP destinations. These qos-group values are used in QOS policies

applied locally on ingress interfaces. Whenever a packet bound for such destinations is encountered, the qos-group value matching that destination route looks up with work inside the policy classmap, and marks that packet for any configured policy actions.

Configuration Workflow

Use the following configuration workflow for QPPB:

- Define route policy.
- Put Route policy at table-policy attach point under BGP.
- Define classmaps and ingress policy to use the qos-groups that are used in table-policy.
- Enable ipv4/ipv6 QPPB configuration under the desired interfaces.
- Configure the QPPB hardware profile, *hw-module profile qos ipv6 short*.
- If you use ipv6 QPPB, you must reload that linecard. If you use only ipv4 QPPB, linecard reload is not mandatory.

Define route policy

A routing policy instructs the router to inspect routes, filter them, and potentially modify their attributes as they are accepted from a peer, advertised to a peer, or redistributed from one routing protocol to another.

The routing policy language (RPL) provides a language to express routing policy. You must set up destination prefixes either to match inline values or one of a set of values in a prefix set.

Example:

```
prefix-set prefix-list-v4
    70.1.1.1,
    70.2.1.0/24,
    70.2.2.0/24 ge 28,
    70.2.3.0/24 le 28
end-set
prefix-set prefix-list-v6
    2001:300::2,
    2003:200::3
end-set

route-policy qppb1
    if destination in (60.60.0.2) then
        set qos-group 5
    elseif destination in prefix-list-v4 then
        set qos-group 4
    else
        set qos-group 1
     pass
endif
end-policy
```

Put Route policy at table-policy attach point under BGP

The table-policy attach point permits the route policy to perform actions on each route as they are installed into the RIB routing table. QPPB uses this attachment point to intercept all routes as they are received from peers. Ultimately the RIB will update the FIB in the hardware forwarding plane to store destination prefix

routing entries, and in cases where table policy matches a destination prefix, the qos-group value is also stored with the destination prefix entry for use in the forwarding plane.

Example:

```
router bgp 900
    [vrf <name>]
    bgp router-id 22.22.22.22
    address-family ipv4 unicast
        table-policy qppb1
    address-family ipv6 unicast
        table-policy qppb2
    neighbor 30.2.2.1
        remote-as 500
        address-family ipv4 unicast
            route-policy pass in
            route-policy pass out
        address-family ipv6 unicast
            route-policy pass in
            route-policy pass out
```

Ingress interface QOS and ipv4/ipv6 bgp configuration

QPPB would be enabled per interface and individually for V4 and V6. An ingress policy would match on the qos groups marked by QPPB and take desired action.

If a packet is destined for a destination prefix on which BGP route policy has stored a qos-group, but it ingresses on an interface on which qppb is not enabled, it would not be remarked with qos-group.

Earlier, router supported matching on qos-group only in peering profile ‘hw-module profile qos ingress-model peering location <>’ . QPPB now permits classmaps to match qos-group in the default “non peering mode qos” as well. Also QPPB and hierarchical QOS policy profiles can work together if Hqos is used.

Example:

```
class-map match-any qos-group5
    match qos-group 5
    end-class-map

class-map match-any qos-group4
    match qos-group 4
    end-class-map

policy-map ingress-marker-po1
    class qos-group5
        set precedence 0
        set discard-class 0
        set traffic-class 1

    class qos-group4
        set precedence 1
        set discard-class 1
        set traffic-class 2
    class class-default

end-policy-map
```

Configuring QPPB on an Interface

1. RP/0/RP0/CPU0:router # configure

   Enters interface configuration mode and associates one or more interfaces to the VRF.

2.
```
RP/0/RP0/CPU0:router(config)# interface
type interface-path-id
```
Enters interface configuration mode and associates one or more interfaces to the VRF.

3. **ipv4 | ipv6 bgp policy propagation inputqos-groupdestination**

Example:

```
RP/0/RP0/CPU0:router(config-if)# ipv4 bgp policy propagation input qos-group destination
```

Enables QPPB on an interface

4. commit

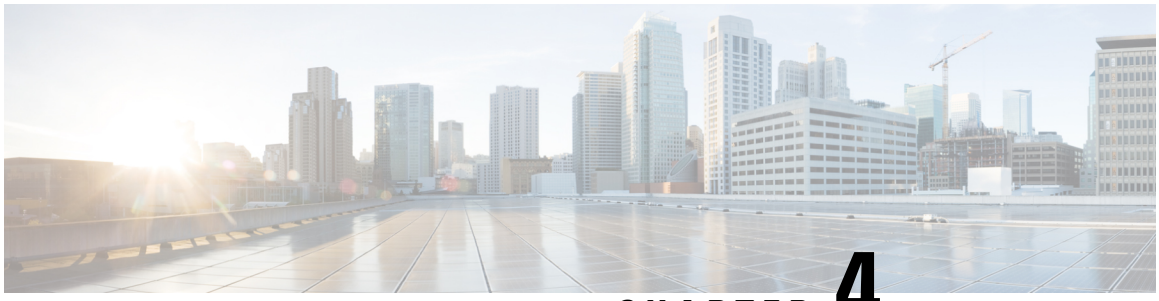
Egress Interface Configuration

The traffic-class set on ingress has no existence outside the device. Also, traffic-class is not a part of any packet header but is associated internal context data on relevant packets. It can be used as a match criteria in an egress policy to set up various fields on the outgoing packet or shape flows.

Restrictions:

- No IP precedence marking.
- No policing on egress policy.

```
class-map match-any level1
     match traffic-class 1
end-class-map

class-map match-any level2
    match traffic-class 2
end-class-map

policy-map output-po1
    class level1
        bandwidth percent 50
    class level2
        bandwidth percent 20
        queue-limit 50 ms
end-policy-map

interface hun 0/5/0/0
     ipv4 address 30.1.1.1/24
     ipv6 address 2001:da8:b0a:12f0::1/64
     service-policy output output-po1
```



CHAPTER 4

Configuring Modular QoS Congestion Avoidance

This chapter covers the following topics:

- Modular QoS Congestion Avoidance , on page 91
- Tail Drop and the FIFO Queue, on page 92
- Random Early Detection and TCP, on page 93
- Weighted Random Early Detection, on page 96
- Explicit Congestion Notification (ECN), on page 99

Modular QoS Congestion Avoidance

Congestion avoidance techniques monitor traffic flow to anticipate and avoid congestion at common network bottlenecks. Avoidance techniques are implemented before congestion occurs as compared with congestion management techniques that control congestion after it has occurred.

Note For traffic requiring header decapsulation, the size of the header that is being removed is still included for the egress queuing actions. To offset this header size (required to achieve line rate for small frame sizes), configure an egress user policy with user overhead accounting on the egress interface. This policy can be a dummy policy configuration as well (allowing full traffic rate), if a policy isn't already in use or required on the egress interface.

You can enable user overhead accounting using the optional configuration of **accounting user-defined <*overhead size in bytes*>** while attaching the service policy on the egress interface.

Congestion avoidance is achieved through packet dropping. The router supports these QoS congestion avoidance techniques:

- Tail Drop and the FIFO Queue, on page 92
- Random Early Detection and TCP, on page 93
- Weighted Random Early Detection, on page 96

Tail Drop and the FIFO Queue

Tail drop is a congestion avoidance technique that drops packets when an output queue is full until congestion is eliminated. Tail drop treats all traffic flow equally and does not differentiate between classes of service. It manages the packets that are unclassified, placed into a first-in, first-out (FIFO) queue, and forwarded at a rate determined by the available underlying link bandwidth.

Configure Tail Drop

Packets satisfying the match criteria for a class accumulate in the queue reserved for the class until they are serviced. The **queue-limit** command is used to define the maximum threshold for a class. When the maximum threshold is reached, the enqueued packets to the class queue result in tail drop (packet drop).

Restrictions

- When configuring the **queue-limit** command, you must configure one of the following commands: **priority**, **shape average**, **bandwidth** or **bandwidth remaining**, except for the default class.

Configuration Example

You have to accomplish the following to complete the tail drop configuration:

1. Creating (or modifying) a policy map that can be attached to one or more interfaces to specify a service policy
2. Associating the traffic class with the traffic policy
3. Specifying the maximum limit the queue can hold for a class policy configured in a policy map.
4. Specifying priority to a class of traffic belonging to a policy map.
5. (Optional) Specifying the bandwidth allocated for a class belonging to a policy map or specifying how to allocate leftover bandwidth to various classes.
6. Attaching a policy map to an output interface to be used as the service policy for that interface.

```
Router# configure
Router(config)# class-map qos-1
Router(config-cmap)# match traffic-class 1
Router(config-cmap)# commit
Router(config-pmap)# exit

Router(config)# policy-map test-qlimit-1
Router(config-pmap)# class qos-1
Router(config-pmap-c)# queue-limit 100 us
Router(config-pmap-c)# priority level 7
Router(config-pmap-c)# exit
Router(config-pmap)# exit

Router(config)# interface HundredGigE 0/6/0/18
Router(config-if)# service-policy output test-qlimit-1
Router(config-if)# commit
```

Running Configuration

```
class-map qos-1
 match traffic-class 1
commit

policy-map test-qlimit-1
 class qos-1
  queue-limit 100 us
  priority level 7
 !
 class class-default
 !
 end-policy-map
!
```

Verification

```
Router# show qos int hundredGigE 0/6/0/18 output

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/6/0/18 ifh 0x3000220  -- output policy
NPU Id:                        3
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
VOQ Base:                      11176
VOQ Stats Handle:              0x88550ea0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class (HP7)                       =   qos-1
Egressq Queue ID                         =   11177 (HP7 queue)
TailDrop Threshold                       =   1253376 bytes / 100 us (100 us)
WRED not configured for this class

Level1 Class                             =   class-default
Egressq Queue ID                         =   11176 (Default LP queue)
Queue Max. BW.                           =   101803495 kbps (default)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 (BWR not configured)
TailDrop Threshold                       =   1253376 bytes / 10 ms (default)
WRED not configured for this class
```

Related Topics

- Tail Drop and the FIFO Queue, on page 92

Associated Commands

- queue-limit

Random Early Detection and TCP

The Random Early Detection (RED) congestion avoidance technique takes advantage of the congestion control mechanism of TCP. By randomly dropping packets prior to periods of high congestion, RED tells the packet

source to decrease its transmission rate. Assuming the packet source is using TCP, it decreases its transmission rate until all packets reach their destination, indicating that the congestion is cleared. You can use RED as a way to cause TCP to slow transmission of packets. TCP not only pauses, but it also restarts quickly and adapts its transmission rate to the rate that the network can support.

RED distributes losses in time and maintains normally low queue depth while absorbing traffic bursts. When enabled on an interface, RED begins dropping packets when congestion occurs at a rate you select during configuration.

Configure Random Early Detection

The **random-detect** command with the **default** keyword must be used to enable random early detection (RED). The **random-detect default** command enables RED using system-defined default thresholds. These thresholds are derived dynamically based on router-specific queue characteristics and available buffer resources. Because the values are hardware-dependent, explicit minimum and maximum thresholds are not displayed or configurable when the default option is used.

Guidelines

If you configure the **random-detect default** command on any class including class-default, you must configure one of the following commands: **shape average**, **bandwidth**, and **bandwidth remaining**.

Configuration Example

You have to accomplish the following to complete the random early detection configuration:

1. Creating (or modifying) a policy map that can be attached to one or more interfaces to specify a service policy
2. Associating the traffic class with the traffic policy
3. Enabling RED with default minimum and maximum thresholds.
4. (Optional) Specifying the bandwidth allocated for a class belonging to a policy map or specifying how to allocate leftover bandwidth to various classes.
5. (Optional) Shaping traffic to the specified bit rate or a percentage of the available bandwidth.
6. Attaching a policy map to an output interface to be used as the service policy for that interface.

```
Router# configure
Router(config)# class-map qos-1
Router(config-cmap)# match traffic-class 1
Router(config-cmap)# commit
Router(config-pmap)# exit

Router# configure
Router(config)# policy-map test-wred-2
Router(config-pmap)# class qos-1
Router(config-pmap-c)# random-detect default
Router(config-pmap-c)# shape average percent 10
Router(config-pmap-c)# end-policy-map
Router(config)# commit
Router(config)# interface HundredGigE 0/6/0/18
Router(config-if)# service-policy output test-wred-2
```

```
Router(config-if)# commit
```

Running Configuration

```
class-map qos-1
 match traffic-class 1
commit

policy-map test-wred-2
 class qos-1
  random-detect default
  shape average percent 10
 !
 class class-default
 !
 end-policy-map
!

interface HundredGigE 0/6/0/18
 service-policy output test-wred-2
!
```

Verification

```
Router# show qos int hundredGigE 0/6/0/18 output

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/6/0/18 ifh 0x3000220  -- output policy
NPU Id:                        3
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
VOQ Base:                      11176
VOQ Stats Handle:              0x88550ea0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   qos-1
Egressq Queue ID                         =   11177 (LP queue)
Queue Max. BW.                           =   10082461 kbps (10 %)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 (BWR not configured)
Guaranteed service rate                  =   10000000 kbps
TailDrop Threshold                       =   12517376 bytes / 10 ms (default)

Default RED profile
WRED Min. Threshold                      =   12517376 bytes (10 ms)
WRED Max. Threshold                      =   12517376 bytes (10 ms)

Level1 Class                             =   class-default
Egressq Queue ID                         =   11176 (Default LP queue)
Queue Max. BW.                           =   101803495 kbps (default)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 (BWR not configured)
Guaranteed service rate                  =   50000000 kbps
TailDrop Threshold                       =   62652416 bytes / 10 ms (default)
WRED not configured for this class
```

Related Topics

- Random Early Detection and TCP, on page 93

Associated Commands

- random-detect

Weighted Random Early Detection

The Weighted Random Early Detection (WRED) drops packets selectively based on any specified criteria, like discard-class. WRED uses this matching criteria to determine how to treat different types of traffic.

You can configure WRED using the **random-detect** command and different discard-class values. The value can be range or a list of values that are valid for that field. You can also use minimum and maximum queue thresholds to determine the dropping point. Ensure that the WRED maximum threshold value is close to the queue limit. When the maximum threshold value is reached, packets start to get dropped.

You can also configure WRED threshold values per discard class. Such an approach helps differentiate when to drop packets among different discard classes, helping prioritize packets among discard classes.

When a packet arrives, the following actions occur:

- The average queue size is calculated.
- If the average queue size is less than the minimum queue threshold, the arriving packet is queued.
- If the average queue size is between the minimum queue threshold for that type of traffic and the maximum threshold for the interface, the packet is either dropped or queued, depending on the packet drop probability for that type of traffic.
- If the average queue size is greater than the maximum threshold, the packet is dropped.

Average Queue Size for WRED

The router automatically determines the parameters to use in the WRED calculations. The average queue size is based on the previous average and current size of the queue. The formula is:

average = (old_average * (1-2 -x)) + (current_queue_size * 2 -x)

where *x* is the exponential weight factor.

For high values of *x*, the previous average becomes more important. A large factor smooths out the peaks and lows in queue length. The average queue size is unlikely to change very quickly, avoiding a drastic change in size. The WRED process is slow to start dropping packets, but it may continue dropping packets for a time after the actual queue size has fallen below the minimum threshold. The slow-moving average accommodates temporary bursts in traffic.

Note

- The exponential weight factor, *x*, is fixed and is not user configurable.
- If the value of *x* gets too high, WRED does not react to congestion. Packets are sent or dropped as if WRED were not in effect.
- If the value of *x* gets too low, WRED overreacts to temporary traffic bursts and drops traffic unnecessarily.

For low values of *x*, the average queue size closely tracks the current queue size. The resulting average may fluctuate with changes in the traffic levels. In this case, the WRED process responds quickly to long queues. Once the queue falls below the minimum threshold, the process stops dropping packets.

Configure Weighted Random Early Detection

This configuration task is similar to that used for RED except that the **random-detect** command is not configured in RED.

Restrictions

- You cannot use the **random-detect** command in a class configured with the **priority** command, because WRED cannot be configured in a class that has been set for priority queueing (PQ).
- When configuring the **random-detect** command, you must configure one of the following commands: **shape average**, **bandwidth**, and **bandwidth remaining**.
- Do not enable Explicit Congestion Notification (ECN) and Weighted Random Early Detection (WRED) at the same time. So, do not use the following commands together:

```
*/ ECN Configuration */
Router(config-pmap-c)# random-detect ecn

*/ WRED Configuration */
Router(config-pmap-c)# random-detect discard-class 0 10 ms 500 ms
```

Configuration Example

You have to accomplish the following to complete the random early detection configuration:

1. Creating (or modifying) a policy map that can be attached to one or more interfaces to specify a service policy
2. Associating the traffic class with the traffic policy
3. Enabling WRED by specifying the match criteria (discard-class).
4. (Optional) Specifying the bandwidth allocated for a class belonging to a policy map or specifying how to allocate leftover bandwidth to various classes.
5. (Optional) Shaping traffic to the specified bit rate or a percentage of the available bandwidth.
6. (Optional) Changing queue limit to fine-tune the amount of buffers available for each queue.
7. Attaching a policy map to an output interface to be used as the service policy for that interface.

```
Router# configure
Router(config)# class-map qos-1
Router(config-cmap)# match traffic-class 1
Router(config-cmap)# commit
Router(config-pmap)# exit

Router# configure
Router(config)# policy-map test-wred-1
Router(config-pmap)# class qos-1
Router(config-pmap-c)# random-detect default
Router(config-pmap-c)# random-detect discard-class 0 10 ms 500 ms
Router(config-pmap-c)# shape average percent 10
Router(config-pmap-c)# commit

Router(config)# interface HundredGigE 0/6/0/18
Router(config-if)# service-policy output policy1
Router(config-if)# commit
```

Running Configuration

```
class-map qos-1
 match traffic-class 1
commit

policy-map test-wred-1
 class qos-1
  random-detect default
  random-detect discard-class 0 10 ms 500 ms
  shape average percent 10
 !
 class class-default
 !
 end-policy-map
!

interface HundredGigE 0/6/0/18
 service-policy output test-wred-1
!
```

Verification

```
Router# show qos int hundredGigE 0/0/0/20 output

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/0/0/20 ifh 0x38  -- output policy
NPU Id:                        0
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
Policy Name:                   test-wred-1
VOQ Base:                      1184
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   qos-1
Egressq Queue ID                         =   1185 (LP queue)
Queue Max. BW.                           =   10000152 kbps (10 %)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   1 / (BWR not configured)
Guaranteed service rate                  =   10000000 kbps
Peak burst                               =   36864 bytes (default)
```

```
TailDrop Threshold                      =   1250000896 bytes / 1000 ms (default)

WRED profile for Discard_Class 0
WRED Min. Threshold                     =   12499968 bytes (10 ms)
WRED Max. Threshold                     =   624999936 bytes (500 ms)

Default RED profile
WRED Min. Threshold                     =   7499776 bytes (6 ms)
WRED Max. Threshold                     =   12499968 bytes (10 ms)

WRED ECN                                =   Disabled

Level1 Class                            =   class-default
Egressq Queue ID                        =   1184 (Default LP queue)
Queue Max. BW.                          =   no max (default)
Queue Min. BW.                          =   0 kbps (default)
Inverse Weight / Weight                 =   1 / (BWR not configured)
Guaranteed service rate                 =   50000000 kbps
Peak burst                              =   36864 bytes (default)
TailDrop Threshold                      =   62499840 bytes / 10 ms (default)
WRED not configured for this class
```

Related Topics

- Weighted Random Early Detection, on page 96
- Configure Random Early Detection, on page 94

Associated Commands

- random-detect

Explicit Congestion Notification (ECN)

Weighted Random Early Detection (WRED) is implemented at the core routers of a network. Edge routers assign IP precedences to packets, as the packets enter the network. With WRED, core routers then use these precedences to determine how to treat different types of traffic. WRED provides separate thresholds and weights for different IP precedences, enabling the network to provide different qualities of service, in regard to packet dropping, for different types of traffic. Standard traffic may be dropped more frequently than premium traffic during periods of congestion.

ECN is an extension to WRED. ECN marks packets instead of dropping them when the average queue length exceeds a specific threshold value. When configured, ECN helps routers and end hosts to understand that the network is congested and slow down sending packets. However, if the number of packets in the queue is above the maximum threshold, packets are dropped based on the drop probability.

WRED starts dropping packets probabilistically before a queue becomes full, which means that it works during enqueuing. ECN marking with WRED (ECN-WRED) takes place as packets are about to be transmitted from the queue (during the dequeue process), when the router decides whether to forward, drop, or mark packets based on the congestion state of the queue. Also, WRED and ECN don’t work together.

RFC 3168, *The Addition of Explicit Congestion Notification (ECN) to IP*, states that with the addition of active queue management (for example, WRED) to the Internet infrastructure, routers are no longer limited to packet loss as an indication of congestion.

Note You cannot use this feature when you have set qos-group or mpls experimental along with a traffic class in the ingress policy.

Implementing ECN

Implementing ECN requires an ECN-specific field that has 2 bits—the ECN-capable Transport (ECT) bit and the CE (Congestion Experienced) bit—in the IP header. The ECT bit and the CE bit can be used to make four ECN field combinations of 00 to 11. The first number is the ECT bit and the second number is the CE bit.

Table 22: ECN Bit Setting

ECT Bit	CE Bit	Combination Indicates
0	0	Not-ECN-capable.
0	1	Endpoints of the transport protocol are ECN-capable.
1	0	Endpoints of the transport protocol are ECN-capable.
1	1	Congestion experienced.

The ECN field combination 00 indicates that a packet is not using ECN. The ECN field combinations 01 and 10—Called ECT(1) and ECT(0), respectively—are set by the data sender to indicate that the endpoints of the transport protocol are ECN-capable. Routers treat these two field combinations identically. Data senders can use either one or both of these two combinations. The ECN field combination 11 indicates congestion to the endpoints. Packets arriving a full queue of a router will be dropped.

Packet Handling When ECN Is Enabled

When the number of packets in the queue is below the minimum threshold, packets are transmitted.

If the number of packets in the queue is above the maximum threshold:

- all incoming packets are dropped when you enable only WRED.
- all incoming packets are marked as CE when you configure ECN on the router.

Note When the number of packets reaches the queue limit, all packets are dropped. This is the identical treatment that a packet receives when you enable WRED without ECN configured on the router.

Three different scenarios arise if the number of packets in the queue is between the minimum threshold and the maximum threshold:

- If the ECN field on the packet indicates that the endpoints are ECN-capable (that is, the ECT bit is set to 1 and the CE bit is set to 0, or the ECT bit is set to 0 and the CE bit is set to 1)—and the WRED algorithm determines that the packet should have been dropped based on the drop probability—the ECT and CE bits for the packet are changed to 1, and the packet is transmitted. This happens because ECN is enabled and the packet gets marked instead of dropped.

- If the ECN field on the packet indicates that neither endpoint is ECN-capable (that is, the ECT bit is set to 0 and the CE bit is set to 0), packet is dropped once the queue limit is reached.
- If the ECN field on the packet indicates that the network is experiencing congestion (that is, both the ECT bit and the CE bit are set to 1), the packet is transmitted. No further marking is required.

Note Applicable until Cisco IOS XR Release 7.11.1: When the incoming IP traffic with ECN bits set to 10 passes through the ingress qos-policy-map that has the class-map definition of `set DSCP/PREC <value>`, then the ECN bits in the IP header gets modified to 01.

Limitations

- ECN configuration is not supported per discard class but for all packets enqueued to a VOQ.
- Do not enable Explicit Congestion Notification (ECN) and Weighted Random Early Detection (WRED) at the same time. So, do not use the following commands together:

```
*/ ECN Configuration */
Router(config-pmap-c)# random-detect ecn

*/ WRED Configuration */
Router(config-pmap-c)# random-detect discard-class 0 10 ms 500 ms
```

Configuration Example

```
Router# configure
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth percent 50
Router(config-pmap-c)# random-detect 1000 packets 2000 packets
Router(config-pmap-c)# random-detect ecn
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# commit
```

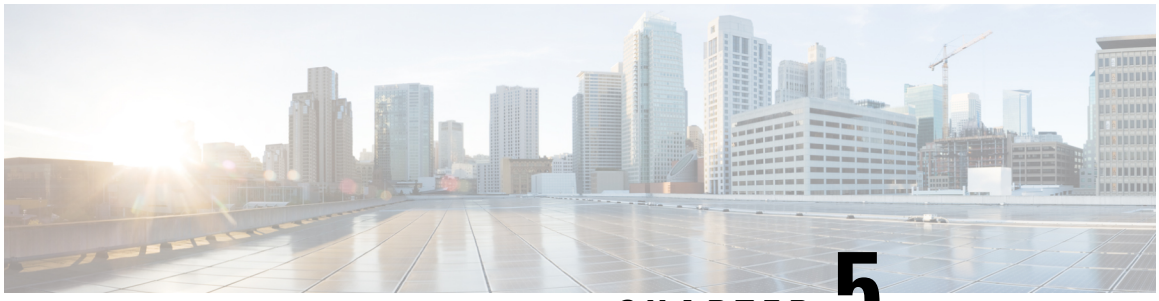
Verification

Use the **show policy-map interface** to verify the configuration.

```
Router# show policy-map interface tenGigE 0/0/0/6 output
TenGigE0/0/0/6 output: pm-out-queue

Class cm-tc-1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :            85528554/87581239296          4830672
    Transmitted         :            16240891/16630672384          966585
    Total Dropped       :            69287663/70950566912          3864087
  Queueing statistics
    Queue ID                              : 1113
    Taildropped(packets/bytes)            : 69287663/70950566912

    WRED profile for
    RED Transmitted (packets/bytes)             : N/A
    RED random drops(packets/bytes)             : N/A
    RED maxthreshold drops(packets/bytes)       : N/A
    RED ecn marked & transmitted(packets/bytes): N/A
```

```
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0                   0
    Transmitted         :                  0/0                   0
    Total Dropped       :                  0/0                   0
  Queueing statistics
    Queue ID                             : 1112
    Taildropped(packets/bytes)           : 0/0
```

Note No ECN-specific statistics are displayed in the show output for this command. ECN is enabled if all rows indicate **N/A**, as highlighted in the example.



CHAPTER 5

Configuring Modular QoS Congestion Management

This chapter covers the following topics:

- Congestion Management Overview, on page 103
- Best practices for minimizing packet drops during sustained maximum line-rate traffic, on page 104
- Class-based Weighted Fair Queueing, on page 104
- Low-Latency Queuing with Strict Priority Queuing, on page 107
- Overhead Accounting, on page 112
- Traffic Shaping, on page 114
- Burst Size for Port-Level Shaper, on page 117
- Traffic Policing, on page 123
- Shared Policer, on page 134
- Prioritize Traffic during Oversubscription, on page 138
- References for Modular QoS Congestion Management, on page 141

Congestion Management Overview

Congestion management features allow you to control congestion by determining the order in which a traffic flow (or packets) is sent out an interface based on priorities assigned to packets. Congestion management entails the creation of queues, assignment of packets to those queues based on the classification of the packet, and scheduling of the packets in a queue for transmission.

The types of traffic regulation mechanisms supported are:

- Class-based Weighted Fair Queueing, on page 104
- Low-Latency Queuing with Strict Priority Queuing, on page 107
- Traffic Shaping, on page 114
- Traffic Policing, on page 123

Best practices for minimizing packet drops during sustained maximum line-rate traffic

When operating Cisco NCS 540 routers at or near 100G line rate, packet drops may occur because of congestion in the egress traffic-management layer. Although it is not possible to eliminate packet drops at sustained wire-rate, you can apply these best practices to minimize loss and maintain optimal traffic flow:

- Shape egress traffic below full line rate: Configure traffic shaping slightly below the 100G line rate, for example, at around 98G, to maintain flows within internal thresholds and avoid Virtual Output Queue (VOQ) congestion. This approach aligns with Cisco's No Drop Rate (NDR) testing guidance.
- Tune egress QoS for burst handling: Increase queue limits moderately for Best-Effort (BE) or default traffic classes, within supported limits. Apply strict queue limits to priority traffic classes to protect their performance. Enable Weighted Random Early Detection (WRED) on BE traffic to smooth packet drops and prevent bursty tail-drop events.
- Avoid sustained oversubscription at 100% utilization: Design traffic profiles with a slight margin below full utilization, especially for mixed Maximum Transmission Unit (MTU) workloads or MPLS/L2VPN flows. While some oversubscription is supported, continuous 100% load will inevitably cause packet drops.
- Align certification and benchmarking expectations: Test and certify using the NDR methodology rather than assuming strict 100% line-rate operation. This provides a more realistic and achievable performance target.

Class-based Weighted Fair Queueing

Class-based Weighted Fair Queueing (CBWFQ) allows definition of traffic classes based on customer match criteria. With CBWFQ you can define traffic classes and assign guaranteed amount of minimum bandwidth to them. CBWFQ also allows for a strict priority queue for delay-sensitive traffic.

Bandwidth Remaining

The CBWFQ algorithm derives the weight for each class from the bandwidth remaining value allocated to the class. The **bandwidth remaining** option specifies a weight for the class to the CBWFQ . After the priority-queue is serviced, the leftover bandwidth is distributed as per bandwidth remaining ratio (BWRR) or percentage. If you do not configure this command for any class, the default value of the BWRR is considered as 1 (one). In the case of **bandwidth remaining percent**, the remaining bandwidth is equally distributed among other classes, to make it 100 percentage (100%).

Restrictions

- The **bandwidth remaining** command is supported only for egress policies.

Configuring Bandwidth Remaining

Supported Platforms: Cisco NCS 540 Series Routers.

This procedure configures the minimum bandwidth and bandwidth remaining on the router

Note The **bandwidth**, **bandwidth remaining**, **shaping**, **queue-limit** and wred commands may be configured together in the same class. But, **priority** cannot be configured along with **bandwidth**, **bandwidth remaining** and wred commands.

You can configure shape average command along with priority command.

Configuration Example

You have to accomplish the following to complete the bandwidth remaining configuration:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed
3. Allocating the leftover bandwidth for the class
4. Attaching the policy-map to an output interface

```
Router# configure
Router(config)#class-map qos-6
Router(config-cmap)#match traffic-class 4
Router(config-cmap)#exit
Router(config-cmap)#commit

Router(config)#class-map qos-5
Router(config-cmap)#match traffic-class 5
Router(config-cmap)#commit

Router(config)# policy-map test-bw-bw-rem
Router(config-pmap)# class qos-6
Router(config-pmap-c)# bandwidth percent 60
Router(config-pmap-c)# bandwidth remaining percent 60
Router(config-pmap)#class qos-5
Router(config-pmap-c)#bandwidth percent 20
Router(config-pmap-c)#bandwidth remaining percent 40
Router(config-pmap-c# exit
Router(config-pmap)# exit
Router(config)# interface HundredGigE 0/6/0/18
Router(config-if)# service-policy output test-bw-bw-rem
Router(config-if)# commit
```

Running Configuration

```
policy-map test-bw-bw-rem
 class qos-6
  bandwidth percent 60
  bandwidth remaining percent 60
 !
 class qos-5
  bandwidth percent 20
  bandwidth remaining percent 40
 !
 class class-default
 !
```

```
 end-policy-map
!

interface HundredGigE0/6/0/18
 service-policy output test-bw-bw-rem
!
```

Verification

```
Router# show qos interface HundredGigE 0/6/0/18 output

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/6/0/18 ifh 0x3000220  -- output policy
NPU Id:                        3
Total number of classes:       3
Interface Bandwidth:           100000000 kbps
VOQ Base:                      11176
VOQ Stats Handle:              0x88550ea0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   qos-6
Egressq Queue ID                         =   11182 (LP queue)
Queue Max. BW.                           =   100824615 kbps (default)
Queue Min. BW.                           =   60494769 kbps (60 %)
Inverse Weight / Weight                  =   2 (60%)
Guaranteed service rate                  =   71881188 kbps
TailDrop Threshold                       =   90177536 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class                             =   qos-5
Egressq Queue ID                         =   11181 (LP queue)
Queue Max. BW.                           =   100824615 kbps (default)
Queue Min. BW.                           =   20164923 kbps (20 %)
Inverse Weight / Weight                  =   3 (40%)
Guaranteed service rate                  =   27920792 kbps
TailDrop Threshold                       =   35127296 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class                             =   class-default
Egressq Queue ID                         =   11176 (Default LP queue)
Queue Max. BW.                           =   101803495 kbps (default)
Queue Min. BW.                           =   0 kbps (default)
Inverse Weight / Weight                  =   120 (BWR not configured)
Guaranteed service rate                  =   198019 kbps
TailDrop Threshold                       =   247808 bytes / 10 ms (default)
WRED not configured for this class
```

Related Topics

- Bandwidth Remaining, on page 104

Associated Commands

- bandwidth remaining

Low-Latency Queuing with Strict Priority Queuing

The Low-Latency Queuing (LLQ) feature brings strict priority queuing (PQ) to the CBWFQ scheduling mechanism. Priority queuing (PQ) in strict priority mode ensures that one type of traffic is sent, possibly at the expense of all others. For PQ, a low-priority queue can be detrimentally affected, and, in the worst case, never allowed to send its packets if a limited amount of bandwidth is available or the transmission rate of critical traffic is high.

Configuring Low Latency Queuing with Strict Priority queuing

Configuring low latency queuing (LLQ) with strict PQ allows delay-sensitive data such as voice to be de-queued and sent before the packets in other queues are de-queued.

Support information

Supported priority levels

- Priority levels 1 to 7 are supported, with 1 being the highest and 7 the lowest.
- The default Class of Service Queue (CoSQ) 0 has the lowest priority among all levels.

Profile-based priority support

- For non-H-QoS profiles, priority levels 1 to 7 are supported.
- For H-QoS profiles, only priority levels 1 to 4 are supported.
- Regardless of the profile type, class-default always maps to CoSQ 0, which has the lowest priority.

Queue assignment for priority Level 1

- Any one of the eight egress class-maps (queues) can have priority level 1 configured.

Guidelines

Queue configuration commands

- You can configure these commands along with the **priority** command:
  - **shape average**
  - **queue-limit**
  - **random-detect**

Priority queue oversubscription

- A PQ can oversubscribe bandwidth when other queues do not fully utilize the port bandwidth.
- Oversubscription is supported only for a single priority level.

Limitations

Egress policing

- Egress policing is not supported. Hence, in strict priority queuing scenarios, lower-priority queues may not be serviced if the priority queue consumes the available bandwidth.

Multiple priority level oversubscription

- Bandwidth over-subscription with multiple priority levels is not a supported configuration.
- Starting with Cisco IOS XR Release 7.0.1, if multiple priority levels are configured for oversubscription:
  - the configuration is accepted, and a warning message is displayed
  - fair queuing is applied across all configured priority levels and,
  - higher-priority traffic is not guaranteed precedence over lower-priority traffic.

Traffic Disruption Risk

- There can be minimal traffic disruption when priority level 1 configuration is applied to any of the 8 queues.

Configuration Example

You have to accomplish the following to complete the LLQ with strict priority queuing:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed.
3. Specifying priority to the traffic class
4. (Optional) Shaping the traffic to a specific bit rate
5. Attaching the policy-map to an output interface

```
Router# configure
Router(config)#class-map qos-1
Router(config-cmap)#match traffic-class 1
Router(config-cmap)#commit

Router(config)#class-map qos-2
Router(config-cmap)#match traffic-class 2
Router(config-cmap)#commit

Router(config)# policy-map test-priority-1
Router(config-pmap)# class qos1
Router(config-pmap-c)# priority level 7
Router(config-pmap-c)# shape average percent 2
Router(config-pmap-c)# class qos-2
Router(config-pmap-c)# priority level 6
Router(config-pmap-c)# shape average percent 1
Router(config-pmap-c)# commit
Router(config-pmap-c# exit
Router(config-pmap)# exit

Router(config)# interface HundredGigE 0/0/0/20
Router(config-if)# service-policy output test-priority-1
Router(config-if)# commit
```

Running Configuration

```
policy-map test-priority-1
 class qos-1
  priority level 7
  shape average percent 2
 !
 class qos-2
  priority level 6
  shape average percent 1
 !
 class class-default
 !
 end-policy-map
!

interface HundredGigE0/0/0/20
 service-policy output test-priority-1
!
```

Verification

```
Router#  show qos int hundredGigE 0/0/0/20 output

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/0/0/20 ifh 0x38  -- output policy
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           100000000 kbps
Policy Name:                   test-priority-1
VOQ Base:                      1184
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class (HP7)                          =   qos-1
Egressq Queue ID                            =   1185 (HP7 queue)
Queue Max. BW.                              =   2000000 kbps (2 %)
Guaranteed service rate                     =   2000000 kbps
Peak burst                                  =   36864 bytes (default)
TailDrop Threshold                          =   2499840 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class (HP6)                          =   qos-2
Egressq Queue ID                            =   1186 (HP6 queue)
Queue Max. BW.                              =   1000000 kbps (1 %)
Guaranteed service rate                     =   1000000 kbps
Peak burst                                  =   36864 bytes (default)
TailDrop Threshold                          =   1249792 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class                                =   class-default
Egressq Queue ID                            =   1184 (Default LP queue)
Queue Max. BW.                              =   no max (default)
Queue Min. BW.                              =   0 kbps (default)
Inverse Weight / Weight                     =   1 / (BWR not configured)
Guaranteed service rate                     =   97000000 kbps
Peak burst                                  =   36864 bytes (default)
TailDrop Threshold                          =   121249792 bytes / 10 ms (default)
WRED not configured for this class
```

Associated Commands

- priority

Configuring Low Latency Queuing for Specific QoS Flows

Table 23: Feature History Table

Feature Name	Release Information	Feature Description
Low Latency Queuing for Specific QoS Flows	Release 7.3.2	This feature allows you to configure low latency or priority for specific QoS flows at the class-level within a policy map. It enables delay-sensitive data to be prioritized and transmitted before other packets in other queues.

This feature allows you to configure low latency or priority for specific QoS flows at the class-level within a policy map. It enables delay-sensitive data to be prioritized and transmitted before other packets in other queues.

Low Latency Queuing (LLQ) is achieved by enabling low delay IPS profile for the designated Virtual Output Queuing (VOQ).

Configuring Low Latency Queuing for Specific QoS Flows

Ingress Class Map Configurations

```
class-map match-any inDSCP_CS6
match dscp cs6
end-class-map
!
class-map match-any inDSCP_CS7
match dscp cs7
end-class-map
!
```

Egress Class Map Configuration

```
class-map match-any outCM6
match traffic-class 6
end-class-map
!
class-map match-any outCM7
match traffic-class 7
end-class-map
!
```

Ingress Policy Map Configurations

```
policy-map Ingress
class inDSCP_CS6
set traffic-class 6
!
class inDSCP_CS7
set traffic-class 7
!
class class-default
```

```
!
end-policy-map
!
```

Egress Policy Map Configuration

```
policy-map Egress
class outCM6
priority level 2
low-latency-profile

!
class outCM7
priority level 1
low-latency-profile
!
```

Verification

```
Router# show controllers npu voq-usage interface all instance all location all
Sat Jul 24 23:33:07.038 UTC

--------------------------------------------------------------------
Node ID: 0/RP0/CPU0
Intf         Intf     NPU NPU  PP   Sys   VOQ   Flow   VOQ    Port
name         handle    #  core Port Port  base  base   port   speed
             (hex)                                    type
--------------------------------------------------------------------------
Hu0/0/0/24   3c000048   0   0    9     9   1032   5384 local   100G
Hu0/0/0/25   3c0000a8   0   0   21    21   1040   5392 local   100G

RP/0/RP0/CPU0:Lion#show  controllers fia   diagshell 0 "diag cosq voq id=1046 det=1 "
location 0/RP0/CPU0
Sat Jul 24 23:34:18.904 UTC

Node ID: 0/RP0/CPU0

R/S/I: 0/0/0

Core 0:
Basic info
        Q type: voq
        num cos: 8, cosq class: 6
        Base queue id: 1040, base queue gport: 0x24000410
Credit request type: BCM_COSQ_DELAY_TOLERANCE_100G_LOW_DELAY Adjusted for low delay 100Gb
ports
Watchdog enable in common status message mode
        Is queue in credit watchdog queue range:True
        Credit watchdog message time: 0
        Delete queue time: 512
        Backoff enter queue credit balance threshold: 98304
        Backoff exit queue credit balance threshold: 98304
        Backlog enter queue credit balance threshold: 7168
        Backlog exit queue credit balance threshold: 7168
        Empty queue satisfied credit balance threshold: 6720
        Max empty queue credit balance threshold: 11200
        Exceed max empty queue credit balance threshold: 1
        Off-To-Slow credit balance threshold: -11264
        Off-To-Normal credit balance threshold: -11264
        Slow-To-Normal credit balance threshold: -11264
        Normal-To-Slow credit balance threshold: -11264
        Delay Tolerance is OCB only
```

Limitations

- No support on logical interfaces and sub-interfaces.
- Device on which low latency is enabled, throughput may be degraded.
- We recommend that you enable only one high priority traffic class (TC) per port for low latency. Enabling multiple TC with low latency may result in burst.

Note Configurations are restricted on any TCs, but it is recommended that you enable LLQ on only one TC.

Overhead Accounting

Traffic shapers and policers use packet traffic descriptors to ensure adherence to the service level agreement in QoS. However, when traffic flows from one hop to another in a network, headers added or removed at interim hops affect the packet bytes being accounted for by QoS at each hop. When your end-user network measures the packet bytes to ensure they receive the payload as agreed, these additional header bytes cause a discrepancy.

QoS overhead accounting provides the flexibility to operators to decide which header bytes can be excluded by the traffic shaper and policer and which can be included, depending on the end user's requirements and device capabilities, to meet the committed payload in units of bytes.

For example, if the QoS commitment includes the additional header bytes, the overhead accounting feature allows your router to account for this overhead and reduces the traffic policing and shaping rates accordingly. This is also called a **positive accounting overhead**.

If however, the committed rate doesn't include the additional bytes, overhead accounting allows your router to adjust the core stream traffic such that the traffic policing and shaping rates are increased. This is also called a **negative accounting overhead**.

To summarize, QoS overhead accounting enables the router to account for packet overhead when shaping and policing traffic to a specific rate. This accounting ensures that the router runs QoS features on the actual bandwidth that the subscriber traffic consumes.

Any interface that supports QoS policies supports overhead accounting.

Note You can enable user overhead accounting using the optional configuration of **accounting user-defined** <***overhead size in bytes***> while attaching the service policy on the egress interface.

Guidelines and Restrictions

- Overhead accounting for ingress shaping is not supported.
- You can't program more than one compensation value per NPU or router, even if they're on different egress ports.
- You can configure the same egress compensation for different egress ports.

Configuring for Overhead Accounting

To configure overhead accounting, you must:

1. Create a policy map and configure QoS actions for that map.
2. Configure overhead accounting and attach the map to an egress interface.

```
/* create QoS policy */
Router#configure terminal
Router(config)#policy-map policer
Router(config-pmap)#class class-default
Router(config-pmap-c)#police rate percent 10
Router(config-pmap-c-police)#commit

/* configure account overhead value while attaching the QoS policy to an egress interface
*/
Router(config)#int hundredGigE 0/0/0/2
Router(config-if)#service-policy output policer account user-defined 12
Router(config-if)#commit
Router(config-if)#root
Router(config)#end
```

Running Configuration

```
Router#sh run int hundredGigE 0/0/0/2
interface HundredGigE0/0/0/2
service-policy output policer account user-defined 12
!
```

The following example shows how to **configure a negative overhead accounting value**:

```
Router#conf
Router(config)#int hundredGigE 0/0/0/2
Router(config-if)#service-policy output policer account user-defined -12
Router(config-if)#commit
```

To **modify an overhead accounting value,** you must:

1. Remove the existing QoS policy and re-add it.
2. Configure the new overhead accounting value.

```
Router#conf
Router(config)#int hundredGigE 0/0/0/2
Router(config-if)#no service-policy input policer
Router(config-if)#service-policy output policer account user-defined -20
Router(config-if)#commit
Router#sh run int hundredGigE 0/0/0/2
interface HundredGigE0/0/0/2
service-policy output policer account user-defined -20
!
```

Positive Accounting Use Case

If QoS commitment includes Preamble, Frame Delimiter & Interframe Gap and has the following configuration:

```
service-policy output <foo> account user-defined +20
```

For QoS purposes, your router treats this packet as a packet of size = Actual Packet size + 20. Hence, the effective policing and shaping is *reduced* to match the downstream interface.

Negative Accounting Use Case

If QoS commitment to your router does not include VLAN header information, and has the following configuration:

```
service-policy output <foo> account user-defined -4
```

For QoS purposes, your router treats this packet as a packet of size = Actual Packet size – 4. Hence, the effective policing and shaping is *increased* to match the downstream interface.

Associated Commands

service-policy (overhead accounting)

Traffic Shaping

Traffic shaping allows you to control the traffic flow exiting an interface to match its transmission to the speed of the remote target interface and ensure that the traffic conforms to policies contracted for it. Traffic adhering to a particular profile can be shaped to meet downstream requirements, hence eliminating bottlenecks in topologies with data-rate mismatches.

Configure VOQ-Level Traffic Shaping

The traffic shaping performed on outgoing interfaces is done at the Layer 1 level and includes the Layer 1 header in the rate calculation.

Guidelines

- You can configure **shape average** command along with **priority** command.
- When testing under 100% line-rate traffic, you may encounter packet loss due to clock drift. This occurs when the Ethernet generator clock runs faster than the device clock on the router. To mitigate this, either reduce the test traffic load from 100% to around 99% or, if the traffic generator permits, adjust the PPM (Parts Per Million) setting to better synchronize with the device clock. These adjustments help minimize packet loss and improve test reliability.

Configuration Example

You have to accomplish the following to complete the traffic shaping configuration:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed
3. Shaping the traffic to a specific bit rate and set peak burst size
4. Attaching the policy-map to an output interface

```
Router# configure
Router(config)#class-map c5
Router(config-cmap)#match traffic-class 5
Router(config-cmap)#commit

Router(config)# policy-map egress_policy1
Router(config-pmap)# class c5
```

```
Router(config-pmap-c)# shape average 40 percentpercent 50 1000
Router(config-pmap-c# exit
Router(config-pmap)# exit
Router(config)# interface HundredGigE 0/1/0/0
Router(config-if)# service-policy output egress_policy1
Router(config-if)# commit
```

Running Configuration

```
class-map c5
 match traffic-class 5
commit

policy-map egress_policy1
 class c5
  shape average percent 40
 !
 class class-default
 !
 end-policy-map
!

interface HundredGigE0/6/0/18
 service-policy output egress_policy1
!



class-map c5
 match traffic-class 5
commit

policy-map egress_policy1
 class c5
  shape average percent 50 1000
 !
 class class-default
 !
 end-policy-map
!

interface HundredGigE0/6/0/18
 service-policy output egress_policy1
!
```

Verification

```
Router#  show qos interface hundredGigE 0/6/0/18 output

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/6/0/18 ifh 0x3000220  -- output policy
NPU Id:                        3
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
VOQ Base:                      11176
VOQ Stats Handle:              0x88550ea0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   c5
Egressq Queue ID                         =   11177 (LP queue)
```

```
Queue Max. BW.                          =   40329846 kbps (40 %)
Queue Min. BW.                          =   0 kbps (default)
Inverse Weight / Weight                 =   1 (BWR not configured)
Guaranteed service rate                 =   40000000 kbps
TailDrop Threshold                      =   50069504 bytes / 10 ms (default)
WRED not configured for this class

Level1 Class                            =   class-default
Egressq Queue ID                        =   11176 (Default LP queue)
Queue Max. BW.                          =   101803495 kbps (default)
Queue Min. BW.                          =   0 kbps (default)
Inverse Weight / Weight                 =   1 (BWR not configured)
Guaranteed service rate                 =   50000000 kbps
TailDrop Threshold                      =   62652416 bytes / 10 ms (default)
WRED not configured for this class
```

```
Router# show qos interface tenGigE 0/0/0/0 output

Wed Jul 10 14:18:37.783 UTC
NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/0 ifh 0x120  -- output policy
NPU Id:                        0
Total number of classes:       1
Interface Bandwidth:           10000000 kbps
Policy Name:                   test
VOQ Base:                      1024
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                            =   class-default
Egressq Queue ID                        =   1024 (Default LP queue)
Queue Max. BW.                          =   5031499 kbps (50 %)
Queue Min. BW.                          =   0 kbps (default)
Inverse Weight / Weight                 =   1 / (BWR not configured)
Guaranteed service rate                 =   5000000 kbps
Peak burst                              =   2240 bytes (1000 bytes)
TailDrop Threshold                      =   6258688 bytes / 10 ms (default)
```

Related Topics

- Congestion Management Overview, on page 103

Associated Commands

- shape average

Burst Size for Port-Level Shaper

Table 24: Feature History Table

Feature Name	Release Information	Feature Description
Burst Size for Port-Level Shaper	Release 7.11.1	You can now achieve a predictable and accurate burst size at the link level by configuring port-level shaper burst size, thus ensuring better adherence to traffic SLAs. Also, with the port-level shaper burst size configured in the egress policy maps, the predictability in peak burst ensures that you can configure any next-hop low-capacity device to handle these bursts. Previously, you could configure burst sizes, which impacted traffic flow only at the Virtual Output Queue (VOQ) level but didn't control packet transmission at the link level.

In relation to a port-level shaper, the burst size refers to the maximum amount of data that can be sent through the port within a short period, exceeding the configured shaping rate. When traffic is shaped at the port level, the shaping algorithm smooths the traffic flow by limiting the average rate at which data is transmitted or received. However, there are scenarios where bursts of traffic may occur, such as during periods of high network activity or when multiple packets arrive simultaneously. In such scenarios, the burst size parameter represents the maximum burst of traffic that the port-level shaper allows before it starts to drop or delay packets.

This value influences the packet transmission and peak burst to the wire, irrespective of the shaper bursts configured on different VOQs. Thus, by providing tighter control of packet flow to the wire and next-hop devices, this functionality helps you achieve accurate and predictable peak bursts.

For example, if you configure the burst size to 100 KB, the shaper allows a burst of up to 100 KB of data to be transmitted within a short period, even if it exceeds the configured shaping rate. If the burst size exceeds 100 KB, the shaper takes action to enforce the shaping rate, such as dropping or delaying packets until the traffic falls within the specified limits. As a corollary, the expected port shaping rate may not be achieved if you configure a very low burst value.

The burst size is programmed on the egress port with:

- a default value of 32 KB for all routers

Burst Size for Port-Level Shaper: Why

Because...

Previously, you could configure burst size only for VOQ-level shaping, where the shaper bandwidth and burst sizes are programmed on the Fair Queue Elements (FQEs) in your router hardware. (FQE is a mechanism that allocates bandwidth fairly among different traffic flows or queues, and the FQEs are specific to ingress queues where egress policy maps are applied.)

and..

The VOQ-level shaper burst impacts credit flow for a particular VOQ. Configuring VOQ-level burst is a best-effort technique but may not translate to the intended peak burst at the link level.

Hence...

The burst size for the port level is port or egress queue-specific, and when you configure it, you have control over packet transmission at the link level, which means you achieve an accurate peak burst in the wire. This predictability in the burst ensures that low-end devices can absorb the burst according to their capability, and there's no unexpected drop in traffic.

Burst Size for Port-Level Shaper: Guidelines and Limitations

- You must configure the port burst along with the port shaper to activate this functionality.
- The burst size is configured on the port and egress queue level.
- The default burst values are:
  - 32 KB for all routers.
- **For Hierarchical QoS (H-QoS) and egress traffic management (ETM) models:**
  - Port shaper and burst on a particular physical interface are programmed when an egress policy map with only a **class-default** configuration and a configured shaper value is applied on that interface.
  - The shaper rate on the default class is calculated as the port shaper and the burst as the port burst.
  - For NCS 5700 line cards [Mode: Compatibility; Native] and NCS 5700 fixed port routers:
    - the port shaper is not supported if no port burst is configured. This limitation does not apply to other platforms.
    - Three-level H-QoS isn't supported. This means that you can't apply two-level egress H-QoS policies on the sub-interfaces and a port shaper policy on the main interface to achieve an aggregated port level SLA in a 1+2 H-QoS or three-level H-QoS model. See Overview of Hierarchical Modular QoS, on page 151 for details about H-QoS.
- **For non-H-QoS and non-ETM models:**
  - Port level burst on a particular interface is programmed when you configure it along with port shaper.
  - You must attach a two-level egress QoS policy map to the main interface. Here, the parent shaper is taken as the port shaper.

- The shaper rate on the parent policy configured for the default class is considered as port shaper and the burst as port burst.
- For NCS 5700 line cards [Mode: Compatibility; Native] and NCS 5700 fixed port routers, the minimum port shaper is 3 Gbps by default if no port burst is configured. For other platforms, the default minimum port shaper value is 1 Gbps. If the port burst is configured, there is no limitation on the minimum port shaper value.

- The port burst value configured on the egress interface doesn't restrict the burst sizes on the VOQs or child classes.
- The parent shaper burst size doesn't restrict the child shaper burst size. Unlike the child shaper bandwidth, a child shaper burst can be greater than the parent shaper burst.
- The actual burst size programmed on the ASIC could vary from the configured value due to a hardware approximation.

Configure Burst Size for Port-Level Shaper

Scenario 1: For Cisco NCS 5700 Series Routers, NCS 5700 line cards, and NCS 5500 Series Routers, in non-H-QoS and non-ETM mode: Assign Port Burst Value

In this scenario, you specifically assign a burst value (say, 2000 bytes) along with the shaper rate (say, 2 Gbps) on a parent policy configured to the default class. These parent shaper and burst values are taken as the port shaper and burst.

1. Create a two-level egress QoS policy map by configuring the parent policy with name, for example, **port-shaper-non-hqos**.
2. Configure the **port-shaper-non-hqos** policy for the default class. There should be no other class other than the default class in the policy map.
3. To the parent policy, apply a child policy named, for example, **egress-child**, using the **service-policy** command.
4. Configure a burst value of 2000 bytes and shaper rate of 2 Gbps for the parent (**port-shaper-non-hqos**) policy.
5. Using the **class** command, specify a traffic class (say **traffic-class-1**) for the child policy (**egress-child**).
6. Using the **shape average** command, configure the shape average for **traffic-class-1** to say, 25%.
7. Attach the parent policy map to the output interface to be used as the service policy for that interface.

This configuration applies the parent shaper and burst values that are taken as the port-level shaper and burst values.

```
Router(config)# policy-map port-shaper-non-hqos
Router(config-pmap)# class class-default
Router(config-pmap-c)# service-policy egress-child
Router(config-pmap-c)# shape average 2 gbps 2000 bytes
Router(config-pmap-c)# exit
Router(config)#policy-map egress-child
Router(config-pmap)# class traffic-class-1
Router(config-pmap-c)# shape average percent 25
Router(config-pmap-c)# exit
```

```
Router(config)# interface tenGigE 0/0/0/0
Router(config-if)# service-policy output port-shaper-non-hqos
Router(config-if)# commit
```

Running Configuration

```
policy-map port-shaper-non-hqos
  class class-default
   service-policy egress-child
   shape average 2 gbps 2000 bytes
 !

 !
policy-map egress-child
 class traffic-class-1
  shape average percent 25
 !
 !

interface tenGigE 0/0/0/0
 service-policy output port-shaper-non-hqos
!
```

Verification

Run the **show qos interface** command to confirm the Peak burst value (2000 bytes) you configured for the default class.

```
Router#show qos interface tenGigE 0/0/0/0 output
NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/0 ifh 0x120  -- output policy
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           10000000 kbps
Policy Name:                   port-shaper-non-hqos
SPI Id:                        0x0
VOQ Base:                      1024
PFC enabled:                   0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                              =   class-default
Queue Max. BW.                            =   2020026 kbps (2 gbits/sec)
Queue Min. BW.                            =   2020026 kbps (default)
Inverse Weight / Weight                   =   1 / (BWR not configured)
Peak burst                             =   2000 bytes (2000 bytes)

   Level2 Class                           =   traffic-class-1
   Egressq Queue ID                       =   1025 (LP queue)
   Queue Max. BW.                         =   505417 kbps (25 %)
   Queue Min. BW.                         =   0 kbps (default)
   Inverse Weight / Weight                =   1 / (BWR not configured)
   Guaranteed service rate                =   500000 kbps
   Peak burst                             =   33600 bytes (default)
   TailDrop Threshold                     =   626688 bytes / 10 ms (default)
   WRED not configured for this class

   Level2 Class                           =   class-default
   Egressq Queue ID                       =   1024 (Default LP queue)
   Queue Max. BW.                         =   no max (default)
   Queue Min. BW.                         =   0 kbps (default)
   Inverse Weight / Weight                =   1 / (BWR not configured)
   Guaranteed service rate                =   1000000 kbps
   Peak burst                             =   33600 bytes (default)
   TailDrop Threshold                     =   1253376 bytes / 10 ms (default)
   WRED not configured for this class
```

Scenario 2: For Cisco NCS 5700 Series Routers, NCS 5700 line cards, and NCS 5500 Series Routers, in non-H-QoS mode: Don't Assign Port Burst Value

In this scenario, you assign only the port shaper value (say, 2 Gbps) to the default class. In such cases, the burst size for port level shaper isn't activated, and the parent class has no port burst value.

1. Create a parent policy map named, for example, **port-shaper-non-hqos**.
2. Configure the **port-shaper-non-hqos** policy for the default class.
3. Apply a child policy-map named, for example, **egress-child** to the default class.
4. Shape the traffic in the default class to an average rate of 2 Gbps, limiting the egress traffic to a specific bandwidth.
5. Attach the parent policy map to the output interface to be used as the service policy for that interface.

```
Router(config)#policy-map port-shaper-non-hqos
Router(config-pmap)#class class-default
Router(config-pmap-c)#service-policy egress-child
Router(config-pmap-c)#shape average 2 gbps
Router(config-pmap-c)#exit
Router(config-pmap)#exit
Router(config)# interface tenGigE 0/0/0/0
Router(config-if)# service-policy output port-shaper-non-hqos
Router(config-if)#commit
```

Running Configuration

```
policy-map port-shaper-non-hqos
 class class-default
  service-policy egress-child
  shape average 2 gbps
 !
!
interface tenGigE 0/0/0/0
 service-policy output port-shaper-non-hqos
!
```

Verification

When you run the **show qos interface** command, you see no port burst value assigned to the parent class.

```
Router#show qos interface tenGigE 0/0/0/0 output
Sun Sep 10 20:17:16.053 UTC
NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/0 ifh 0xa0  -- output policy
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           10000000 kbps
Policy Name:                   port-shaper-non-hqos
SPI Id:                        0x0
VOQ Base:                      1024
PFC enabled:                   0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   class-default
Queue Max. BW.                           =   2100096 kbps (2 gbits/sec)
Queue Min. BW.                           =   2100096 kbps (default)
Inverse Weight / Weight                  =   1 / (BWR not configured)

   Level2 Class                             =   traffic-class-1
```

```
  Egressq Queue ID                          =   1025 (LP queue)
  Queue Max. BW.                            =   505861 kbps (25 %)
  Queue Min. BW.                            =   0 kbps (default)
  Inverse Weight / Weight                   =   1 / (BWR not configured)
  Guaranteed service rate                   =   500000 kbps
  Peak burst                                =   33600 bytes (default)
  TailDrop Threshold                        =   626688 bytes / 10 ms (default)
  WRED not configured for this class

  Level2 Class                              =   class-default
  Egressq Queue ID                          =   1024 (Default LP queue)
  Queue Max. BW.                            =   1011732 kbps (50 %)
  Queue Min. BW.                            =   0 kbps (default)
  Inverse Weight / Weight                   =   1 / (BWR not configured)
  Guaranteed service rate                   =   1000000 kbps
  Peak burst                                =   33600 bytes (default)
  TailDrop Threshold                        =   1253376 bytes / 10 ms (default)
  WRED not configured for this class
```

Scenario 3: For NCS 5700 Series Routers and NCS 5700 line cards, in H-QoS and ETM-enabled mode: Assign Port Burst Value

In this scenario, you specifically assign a burst value (say, 2000 bytes) along with the shaper rate (say, 2 Gbps) on the default class.) See Burst Size for Port-Level Shaper: Guidelines and Limitations, on page 118 for limitations that apply to this scenario.

1. Create a policy map named, for example, **port-shaper-hqos**.
2. Configure the **port-shaper-hqos** policy for the default class. The **class-default** class matches all traffic that does not match any other specific class. There should be no other class other than the default class in the policy map.
3. Configure the traffic shaping rate and the burst value for the **class-default** class.
4. Apply the policy-map named **port-shaper-hqos** as the output service policy for the interface.

```
Router (config)# policy-map port-shaper-hqos
Router (config-pmap)# class class-default
Router (config-pmap-c)# shape average 2 gbps 2000 bytes
Router (config-pmap-c)#exit
Router (config-pmap)#exit
Router(config)# interface tenGigE 0/0/0/0
Router(config-if)# service-policy output port-shaper-hqos
Router(config-if)#commit
```

Running Configuration

```
policy-map port-shaper-hqos
 class class-default
  shape average 2 gbps 2000 bytes
 !
!
interface tenGigE 0/0/0/0
 service-policy output port-shaper-hqos
!
```

Verification

Run the **show qos interface** to confirm the peak burst value (2000 bytes) you configured for the default class. Because there's no three-level H-QoS shaping support on NCS 5700 line cards [Mode: Compatibility; Native] and NCS 5700 fixed port routers, the output displays only the default class details (and not other child levels) for which you configured the shaper and burst values.

```
Router#show qos interface tenGigE 0/0/0/0 output
NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/0 ifh 0xa0  -- output policy
NPU Id:                        0
Total number of classes:       1
Interface Bandwidth:           10000000 kbps
Policy Name:                   port-shaper-hqos
SPI Id:                        0x0
VOQ Base:                      1024
PFC enabled:                   0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                                =   class-default
Egressq Queue ID                            =   1024 (Default LP queue)
Queue Max. BW.                              =   2024355 kbps (2 gbits/sec)
Queue Min. BW.                              =   0 kbps (default)
Inverse Weight / Weight                     =   60 / (BWR not configured)
Guaranteed service rate                     =   2000000 kbps
Peak burst                                  =   2000 bytes (2000 bytes)
TailDrop Threshold                          =   2506752 bytes / 10 ms (default)
WRED not configured for this class
```

Traffic Policing

Traffic policing allows you to control the maximum rate of traffic sent or received on an interface and to partition a network into multiple priority levels or class of service (CoS).Traffic policing manages the maximum rate of traffic through a token bucket algorithm. The token bucket algorithm uses user-configured values to determine the maximum rate of traffic allowed on an interface at a given moment in time. The token bucket algorithm is affected by all traffic entering or leaving the interface (depending on where the traffic policy with traffic policing is configured) and is useful in managing network bandwidth in cases where several large packets are sent in the same traffic stream. By default, the configured bandwidth value takes into account the Layer 2 encapsulation that is applied to traffic leaving the interface.

Traffic policing also provides a certain amount of bandwidth management by allowing you to set the burst size (Bc) for the committed information rate (CIR). See, Committed Bursts and Excess Bursts, on page 124.

The router supports the following traffic policing mode(s):

- Single-Rate Two-Color (SR2C) in color-blind mode. See Single-Rate Policer, on page 124.
- Single-Rate Three-Color (SR3C) in color-blind mode.
- Two-Rate Three-Color (2R3C) in color-blind mode. See Two-Rate Policer, on page 128 .

Restrictions

- Traffic policing is supported only in ingress direction, and only color-blind mode is supported.
- The policing rate accuracy may vary up to +/-2% from the configured policer value.
- Ensure that you don’t configure a policer and match criteria for **discard-class** in the same class. Even though the configuration is allowed, the policer doesn’t work and allows all traffic without dropping packets.
- Policer marking is not supported.

- Policers are configured in the interface at the core level and “show qos int <>” value is displayed at the NPU level.

  For policers configured in a bundle interface where bundle members are from the same NPU but different cores (NPU cores), each member sends the traffic up to the core level policer configuration, but “show qos int <>” displays the NPU level policer output.

- Example:

  For bundle interface with two 10GE members (same NPU, but one interface from core0, one interface from core1) 2R3C policer applied on bundle interface (1G confirm rate , 1G exceed rate – total 2G policer rate) will be shown on the “show qos int <>” output):

  Interface in core0 – 500 Mbps confirm rate, 500 Mbps exceed rate

  Interface in core1 – 500 Mbps confirm rate, 500 Mbps exceed rate

  For traffic in one out of two interfaces, the policed rate will be 1Gbps. For traffic on two interfaces, policed rate will be 2Gbps.

Committed Bursts and Excess Bursts

Unlike a traffic shaper, a traffic policer does not buffer excess packets and transmit them later. Instead, the policer executes a “send or do not send” policy without buffering. Policing uses normal or committed burst (bc) values and excess burst values (be) to ensure that the router reaches the configured committed information rate (CIR). Policing decides if a packet conforms or exceeds the CIR based on the burst values you configure. Burst parameters are based on a generic buffering rule for routers, which recommends that you configure buffering to be equal to the round-trip time bit-rate to accommodate the outstanding TCP windows of all connections in times of congestion. During periods of congestion, proper configuration of the excess burst parameter enables the policer to drop packets less aggressively.

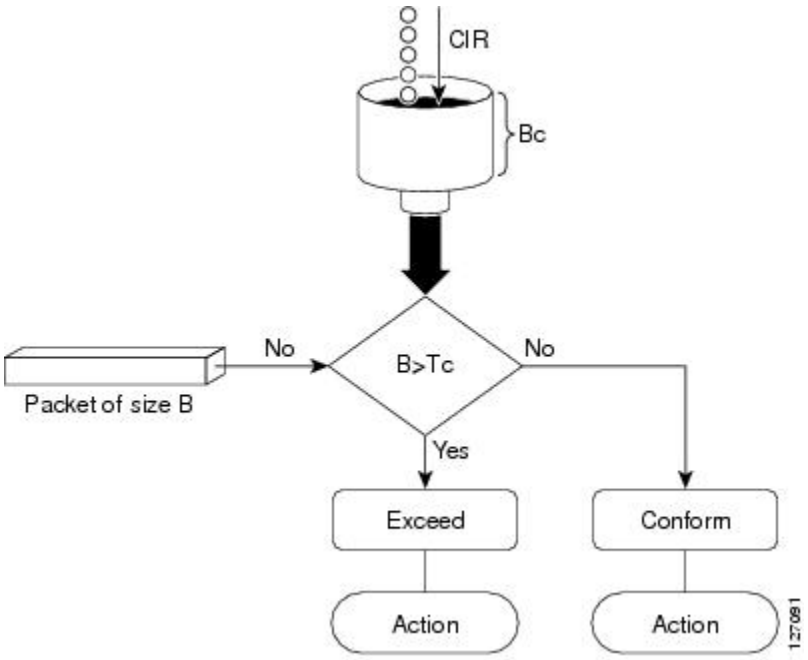
For more details, see Committed Bursts, on page 141 and Excess Bursts, on page 141.

Single-Rate Policer

Single-Rate Two-Color Policer

A single-rate two-color (SR2C) policer provides one token bucket with two actions for each packet: a conform action and an exceed action.

Figure 4: Workflow of Single-Rate Two-Color Policer



Based on the committed information rate (CIR) value, the token bucket is updated at every refresh time interval. The Tc token bucket can contain up to the Bc value, which can be a certain number of bytes or a period of time. If a packet of size B is greater than the Tc token bucket, then the packet exceeds the CIR value and a default action is performed. If a packet of size B is less than the Tc token bucket, then the packet conforms and a different default action is performed.

Single-Rate Three-Color Policer

A single-rate three-color (SR3C) policer provides one token bucket with three actions for each packet: a conform action, an exceed action and a violate action. The packet is marked based on the CIR value and the two associated burst size - committed burst size (CBS) and excess burst size (EBS). If a packet does not exceed the CBS, it is marked as conformed packet. The packet is marked as exceeded if it exceeds CBS, but not the EBS. If it exceeds the EBS as well, it is marked as violate packet.

Configure Traffic Policing (Single-Rate Two-Color)

Traffic policing is often configured on interfaces at the edge of a network to limit the rate of traffic entering or leaving the network. The default conform action for single-rate two color policer is to transmit the packet and the default exceed action is to drop the packet. Users cannot modify these default actions.

Configuration Example

You have to accomplish the following to complete the traffic policing configuration:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed
3. (Optional) Specifying the marking action
4. Specifying the policy rate for the traffic

5. Attaching the policy-map to an input interface

```
Router# configure
Router(config)# policy-map test-police-1
Router(config-pmap)# class  ipv6-6
Router(config-pmap-c)# set dscp cs2 (optional)
Router(config-pmap-c)# set qos-group 7 (optional)
Router(config-pmap-c)# police rate percent 20 burst 10000 bytes
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# interface HundredGigE 0/6/0/18
Router(config-if)# service-policy input test-police-1
Router(config-if)# commit
```

Running Configuration

```
class-map match-any ipv6-6
 match precedence 3
 end-class-map
!

policy-map test-police-1
 class ipv6-6
  set dscp cs2
  set qos-group 7
  police rate percent 20 burst 10000 bytes
  !
 !
 class class-default
 !
 end-policy-map
!

interface HundredGigE0/6/0/18
 service-policy input test-police-1
 service-policy output test-priority-1
!
```

Verification

```
Router# show qos interface hundredGigE 0/6/0/18 input

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/6/0/18 ifh 0x3000220  -- input policy
NPU Id:                        3
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   ipv6-6
New dscp                                 =   16
New qos group                            =   7

Policer Bucket ID                        =   0x102a0
Policer Stats Handle                     =   0x8a8090c0
Policer committed rate                   =   19980000 kbps (20 %)
```

```
Policer conform burst                    =   9856 bytes (10000 bytes)

Level1 Class                             =   class-default

Default Policer Bucket ID                =   0x102a1
Default Policer Stats Handle             =   0x8a808e78
Policer not configured for this class
```

Related Topics

- Traffic Policing, on page 123

Associated Commands

- police rate

Configure Traffic Policing (Single-Rate Three-Color)

The default conform action and exceed actions for single-rate three-color policer are to transmit the packet and the default violate action is to drop the packet. User cannot modify these default actions.

Configuration Example

You have to accomplish the following to complete the traffic policing configuration:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed
3. (Optional) Specifying the marking action
4. Configuring the policy rate for the traffic along with the peak-burst values
5. Attaching the policy-map to an input interface

```
Router# configure
Router(config)# policy-map test-police-1R3C
Router(config-pmap)# class ipv4-5
Router(config-pmap-c)# set qos-group 2 (optional)
Router(config-pmap-c)# police rate percent 20 burst 100000 bytes peak-burst 190000 bytes
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# interface HundredGigE 0/6/0/18
Router(config-if)# service-policy input test-police-1R3C
Router(config-if)# commit
```

Running Configuration

```
class-map match-any ipv4-5
 match precedence 3
 end-class-map
!

policy-map test-police-1R3C
```

```
 class ipv4-5
  set qos-group 7
  police rate percent 20 burst 100000 bytes peak-burst 190000 bytes
  !
 !
 class class-default
 !
 end-policy-map
!

interface HundredGigE0/6/0/18
 service-policy input test-police-1R3C
 service-policy output test-priority-1
!
```

Verification

```
Router# show qos interface hundredGigE 0/6/0/18 input

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/6/0/18 ifh 0x3000220  -- input policy
NPU Id:                        3
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                                =   ipv4-5
New qos group                               =   2

Policer Bucket ID                           =   0x102a1
Policer Stats Handle                        =   0x8a8090c0
Policer committed rate                      =   19980000 kbps (20 %)
Policer conform burst                       =   99584 bytes (100000 bytes)
Policer exceed burst                        =   188672 bytes (190000 bytes)

Level1 Class                                =   class-default

Default Policer Bucket ID                   =   0x102a1
Default Policer Stats Handle                =   0x8a808e78
Policer not configured for this class
```

Related Topics

- Traffic Policing, on page 123

Associated Commands

- police rate

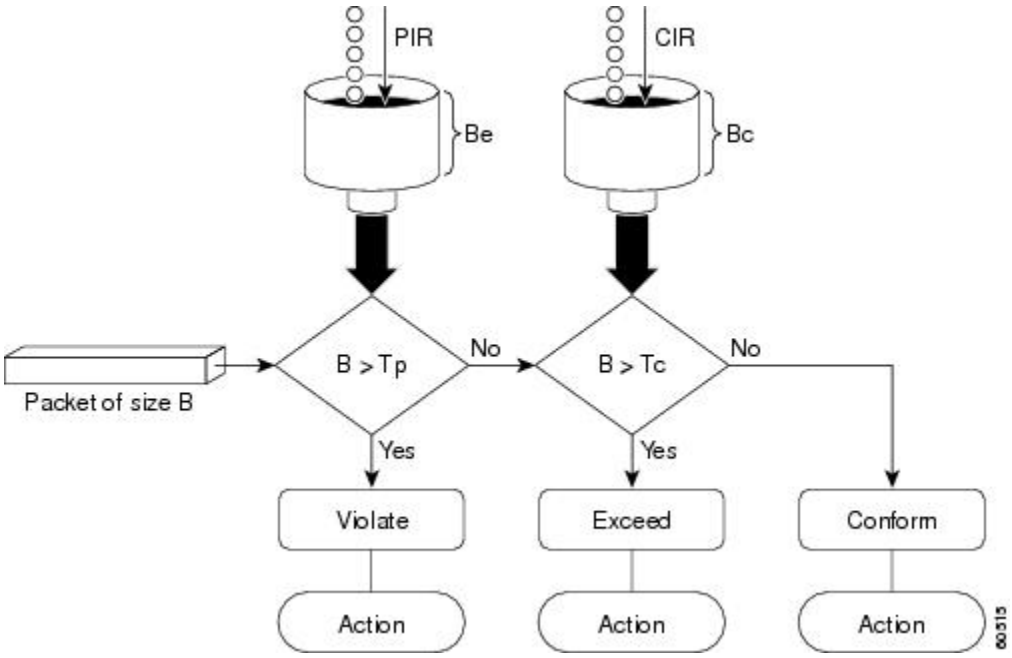
Two-Rate Policer

The two-rate policer manages the maximum rate of traffic by using two token buckets: the committed token bucket and the peak token bucket. The dual-token bucket algorithm uses user-configured values to determine the maximum rate of traffic allowed on a queue at a given moment. In this way, the two-rate policer can meter traffic at two independent rates: the committed information rate (CIR) and the peak information rate (PIR).

The dual-token bucket algorithm provides users with three actions for each packet—a conform action, an exceed action, and an optional violate action. Traffic entering a queue with the two-rate policer configured is placed into one of these categories. The actions are pre-determined for each category. The default conform and exceed actions are to transmit the packet, and the default violate action is to drop the packet.

This figure shows how the two-rate policer marks a packet and assigns a corresponding action to the packet.

Figure 5: Marking Packets and Assigning Actions—Two-Rate Policer



Also, see Two-Rate Policer Details, on page 142.

The router supports Two-Rate Three-Color (2R3C) policer.

Configure Traffic Policing (Two-Rate Three-Color)

The default conform and exceed actions for two-rate three-color (2R3C) policer are to transmit the packet and the default violate action is to drop the packet. Users cannot modify these default actions.

Configuration Example

You have to accomplish the following to complete the two-rate three-color traffic policing configuration:

1. Creating or modifying a policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created or changed
3. Specifying the packet marking
4. Configuring two rate traffic policing
5. Attaching the policy-map to an input interface

```
Router# configure
Router(config)# policy-map policy1
```

```
Router(config-pmap)# class ipv4-7
Router(config-pmap-c)# set qos-group 4
Router(config-pmap-c)# police rate percent 20 burst 100000 bytes peak-rate percent 50
peak-burst 200000 bytes
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# interface HundredGigE 0/6/0/18
Router(config-if)# service-policy input policy1
Router(config-if)# commit
```

Running Configuration

```
policy-map policy1
 class ipv4-7
  set qos-group 4
  police rate percent 20 burst 100000 bytes peak-rate percent 50 peak-burst 200000 bytes
  !
 !

interface HundredGigE 0/6/0/18
 service-policy input policy1
!
```

Verification

```
Router# show policy-map interface HundredGigE 0/6/0/18

NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/6/0/18 ifh 0x3000220  -- input policy
NPU Id:                        3
Total number of classes:       8
Interface Bandwidth:           100000000 kbps
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   ipv4-4
- - -
- - -
Level1 Class                             =   ipv4-7
New qos group                            =   4

Policer Bucket ID                        =   0x102a3
Policer Stats Handle                     =   0x8a8089e8
Policer committed rate                   =   19980000 kbps (20 %)
Policer peak rate                        =   49860000 kbps (50 %)
Policer conform burst                    =   99584 bytes (100000 bytes)
Policer exceed burst                     =   199168 bytes (200000 bytes)

Level1 Class                             =   class-default

Policer Bucket ID                        =   0x102a7
Policer Stats Handle                     =   0x8a7c8510
Policer committed rate                   =   29880000 kbps (30 %)
Policer conform burst                    =   4194304 bytes (default)
```

Important Notes

- From Cisco IOS XR Release 6.6.25 onwards:
  - a committed information rate of 0 kbps is supported, which is the only exception to the minimum rate of 22 kbps for both commit and excess rates.
  - a policer on a bundle interface also allows absolute rates apart from the already supported units of percent, per-thousand and per-million.
- A policer is programmed per NPU core on a bundle interface. So, all members on a bundle interface from the same core share the policer.

Related Topics

- Two-Rate Policer, on page 128

Associated Commands

- police rate

Packets-Per-Second-Based Policer

Table 25: Feature History Table

Feature Name	Release Information	Feature Description
Packets-Per-Second-Based Policer	Release 7.4.1	Prior to this functionality, when configuring policers, the only available option for policer rates was bit-rate measured in units of bits per second (bps). With this release, you can configure policer rates in units of packets per second (pps) as well. pps-based policer is critical in fending off malicious attacks—when attackers target your specific resources with a vast amount of traffic that contain higher number of packets, but move at a slower packet rate. Protection from such attacks is possible because pps-based policers ensure that regardless of the packet size and rate, the policer only accepts a fixed number of packets per second. This functionality modifies the **police rate** command.

- **Policer rates so far**—You used the **police rate** command to configure policers based on two parameters:
  - bit-rates (default unit: bits per second or bps)

- Burst size (default unit: bytes)

- **packets-per-second (pps)-based policer**—With this additional functionality, you can use the **police rate** command to configure policers in units of packets per second (pps). The pps configuration option is available as **police rate <*pps-value*> pps**. When you configure the pps option, ensure that you configure burst size in packets. (See **Restrictions and guidelines**.) Thus, the parameters for pps-based policer are:
  - packets per second (pps)
  - burst size (packets)

- **Why pps-based-policer**—Networks face newer types of attacks, and these days malicious operators don't necessarily employ aggressive tactics that involve overwhelming your bandwidth with large amount of traffic to cause distributed denial of service (DDoS). Now, some attackers go the 'softer' route, where they send smaller packet sizes at slower traffic rates. During such malicious network activity, a bandwidth-based policer can still aggregate up to many packets to be processed if the packet size is small. Attackers tend to use this behavior to bypass bandwidth-based policers to exploit vulnerabilities or try to hit performance limitations by increasing the packet rates.

  Packets-per-second-based policers ensure that regardless of the packet size and traffic rate, the policer only accepts a fixed number of packets per second.

  pps-based-policer support cheat-sheet—Here's a quick look at some key support areas and their details for pps-based policer.

Support	Details
Classification and marking support	Same as that for bps-based-policer
Units	Equivalent kbps values display for QoS programming and statistics.
H-QoS	Support for parent and child policers
Bursts	Support for confirm burst (bc) and exceed burst (be) values in units of packets. The default value is in multiple of 128 bytes equivalent to 10 milliseconds.
Minimum pps value	For better granularity, recommended minimum value is 100 pps.

- **Restriction and guidelines**
  - This functionality is **applicable only for ingress.**
  - When using a pps-based policer, **ensure that you configure the burst-size value in number of packets as well.** This is because a policer burst rate determines whether a specific number of packets out of contract would be subject to the next action (that is, exceed or violate).
  - **Within a QoS policy, configure the parent and child policies policers to either bps or pps.** Else, the configuration displays an error when you try attaching the policy to an interface.
  - **For single-level policy maps:** under the same policy map, you can configure one class map with bps-based policer and the other class map with a pps-based policer.

- **For two-level hierarchical policy maps:**
  - The parent and child-level policy maps must use the same unit-based policer. That is, both must have either pps-based or bps-based policers.
  - If you configure the child-level policy map with pps-based policer, ensure that the parent policy-map class default has a pps-based policer.

- **Configure pps-based policer**—To configure pps-based policer, you must:
  1. Configure a class map.
  2. Create a service policy for the map and configure the pps values.
  3. Attach the service policy to an interface.

```
/*Configure a class map*/

Router(config)#class-map prec1
Router(config-cmap)#match precedence 1
Router(config-cmap)# exit
Router(config)# commit

/*Create a service policy map*/

Router(config)# policy-map policy1
Router(config-pmap)# class prec1
Router(config-pmap-c)#police rate 1000 pps burst 300 packets
Router(config-pmap-c-police)#exit
Router(config-pmap-c)#exit
Router(config-pmap)#exit
Router(config)# commit

/*Attach the service policy to an interface*/
Router#int hundredGigE 0/7/0/2
Router(config-if)service-policy input policy1
Router(config-if)#exit
Router(config)#commit
```

Running Configuration

```
class-map match-any prec1
match precedence 1
 end-class-map
!
policy-map policy1
class prec1
  police rate 1000 pps burst 300 packets
  !
 !
 class class-default
!
 end-policy-map
!
```

Verification

```
Router#show qos int hundredGigE 0/7/0/2 input
NOTE:- Configured values are displayed within parentheses
Interface HundredGigE0/7/0/2 ifh 0xe000088  -- input policy
```

```
NPU Id:                        0
Total number of classes:       2
Interface Bandwidth:           100000000 kbps
Policy Name:                   policy1
SPI Id:                        0x0
Accounting Type:               Layer2 (Include Layer 2 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                                =   prec1

Policer Bucket ID                           =   0x9
Policer Stats Handle                        =   0x0
Policer committed rate                      =   998 kbps (1000 packets/sec)
Policer conform burst                       =   37632 bytes (300 packets)

Level1 Class                                =   class-default

Default Policer Bucket ID                   =   0x8
Default Policer Stats Handle                =   0x0
Policer not configured for this class
```

Associated Commands

police rate

Shared Policer

The classification of the incoming packet occurs only once. However, based on the different classification criteria, the shared policer feature allows sharing of the policer bucket amongst two or more classes in a QoS policy map . That is, the same token bucket is used for a traffic flow matching against any of the classes sharing the policer.

For example, let us say a policer of 10 Mbps is shared among two classes C1 and C2. This feature ensures that both C1 and C2 get traffic flow assigned on First Come First Serve (FCFS) basis. Also that, if C2 does not have any traffic, C1 uses all of the 10 Mbps for transmission.

This feature includes two components:

- Policer Bucket Shared
- Policer Bucket Referred

Policer Bucket Shared

The policer bucket shared feature defines and shares a policer bucket instance among multiple classes.

Here is a sample configuration that defines and shares policer bucket instance sp1 :

```
policy-map parent
        class long-distance
          police bucket shared sp1 rate 1 mbps
```

In this configuration, a policy-map for class long-distance traffic type is created to police at 1Mbps rate and the policer bucket is shared.

Policer Bucket Referred

The policer bucket referred feature refers a defined policer bucket instance. Shared policer is not supported across policy levels. This means for example, that parent and child policy cannot share a common bucket.

Here is a sample configuration that refers shared policer bucket instance sp1 :

```
policy-map voip-child
        class long-distance-voip
         police bucket referred sp1
```

In this configuration, a policy-map for class long-distance-voip traffic type is created and the shared policer bucket sp1 is referred.

Shared Policer Statistics

Currently, individual class statistics are not available as a default option for shared policer. You can access statistics in the following modes.

Aggregate Mode

In this mode the policer bucket is shared among two or more classes. However, statistics are not available for every individual class. You can view the aggregate statistics that combine the numbers for all the classes sharing the policer bucket.

Per-Class Mode

In this mode the policer bucket is shared among two or more classes, and you can also view individual class statistics. However, when this mode is active, the Policy-Based Tunnel Selection (PBTS) mechanism is disabled. To enable the per-class mode, you must configure the **hw-module profile qos shared-policer-per-class-stats** command.

Restrictions and Guidelines

The following restrictions and guidelines apply while configuring the shared policer feature.

- When shared policer is enabled in per-class mode, Policy-Based Tunnel Selection (PBTS) mechanism is disabled. In other words, shared policer-per-class-mode and PBTS are mutually exclusive features.
- Shared policer is not supported across policy levels. This means, for example, that parent and child policies cannot share a common policer bucket.
- Shared policer is not supported in ingress peering mode.
- Shared policer is supported within classes of the same policy. However, cross-policy bucket sharing is not supported.
- There are no limitations on the number of classes that can share policer.
- There are no changes in policer scale numbers in the aggregate and per-class modes.
- All the existing policer types (1R2C, 1R3C and 2R3C) are supported.
- You must reload the affected line card to enable the per-class-stats mode.

Configuring Shared Policer

To configure shared policer, you must:

1. Create a class map to be used for matching packets to the specified class.
2. Create a policy map to be used for matching packets to the specified class.
3. Specify a class name.
4. Define and share a policer bucket.
5. Specify a class name.
6. Refer a shared policer bucket.

```
RP/0/RSP0/CPU0:ios(config)#class-map match-any c1
RP/0/RSP0/CPU0:ios(config-cmap)#match precedence 1
RP/0/RSP0/CPU0:ios(config-cmap)#end-class-map
RP/0/RSP0/CPU0:ios(config)#class-map match-any c2
RP/0/RSP0/CPU0:ios(config-cmap)#match precedence 2
RP/0/RSP0/CPU0:ios(config-cmap)#end-class-map
RP/0/RSP0/CPU0:ios(config)#policy-map s-pol
RP/0/RSP0/CPU0:ios(config-pmap)#class c1
RP/0/RSP0/CPU0:ios(config-pmap-c)#police bucket shared 1 rate 10 mbps
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#exit
RP/0/RSP0/CPU0:ios(config-pmap-c)#exit
RP/0/RSP0/CPU0:ios(config-pmap)#class c2
RP/0/RSP0/CPU0:ios(config-pmap-c)#police bucket referred 1
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#class class-default
RP/0/RSP0/CPU0:ios(config-pmap-c)#exit
RP/0/RSP0/CPU0:ios(config-pmap)#exit
RP/0/RSP0/CPU0:ios(config)#interface HundredGigE 0/6/0/18
RP/0/RSP0/CPU0:ios(config-if)#service-policy input s-pol
RP/0/RSP0/CPU0:ios(config-if)#commit
```

Running Configuration

```
class-map match-any c1
 match precedence 1
 end-class-map

class-map match-any c2
 match precedence 2
 end-class-map

policy-map s-pol
 class c1
  police bucket shared 1 rate 10 mbps
  !
 !
 class c2
  police bucket referred 1
  !
 !
 class class-default
 !
 end-policy-map
!

interface HundredGigE0/6/0/18
 service-policy input s-pol
!
```

Verification

In Aggregate Mode

```
RP/0/RP0/CPU0:ios#sh policy-map interface tenGigE 0/0/0/0 input
Fri Nov 15 12:55:56.817 UTC

TenGigE0/0/0/0 input: s-pol

Class c1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          1784530245/228419871360         8640780
    Transmitted         :             2067504/264640512            10011
    Total Dropped       :          1782462741/228155230848         8630769
  Policing statistics                (packets/bytes)     (rate - kbps)
    Policed(conform)    :             2067504/264640512            10011
    Policed(exceed)     :          1782462741/228155230848         8630769
    Policed(violate)    :                   0/0                    0
    Policed and dropped :          1782462741/228155230848
Class c2
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                   0/0                    0
    Transmitted         :                   0/0                    0
    Total Dropped       :                   0/0                    0
  Policing statistics                (packets/bytes)     (rate - kbps)

Policed(conform)    :                  0/0                    0
    Policed(exceed)     :                   0/0                    0
    Policed(violate)    :                   0/0                    0
    Policed and dropped :                   0/0
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                   0/0                    0
    Transmitted         :                   0/0                    0
    Total Dropped       :                   0/0                    0
Policy Bag Stats time: 1573822531986  [Local Time: 11/15/19 12:55:31.986]
```

In Per-Class Mode

```
RP/0/RP0/CPU0:ios#sh policy-map interface tenGigE 0/0/0/0 input
Fri Nov 15 15:18:18.319 UTC

TenGigE0/0/0/0 input: s-pol

Class c1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          1005369276/128687267328         4320337
    Transmitted         :             1163300/148902400            5013
    Total Dropped       :          1004205976/128538364928         4315324
  Policing statistics                (packets/bytes)     (rate - kbps)
    Policed(conform)    :             1163300/148902400            5013
    Policed(exceed)     :          1004205976/128538364928         4315324
    Policed(violate)    :                   0/0                    0
    Policed and dropped :          1004205976/128538364928
Class c2
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          1005341342/128683691776         4320335
    Transmitted         :             1166269/149282432            4997
    Total Dropped       :          1004175073/128534409344         4315338
  Policing statistics                (packets/bytes)     (rate - kbps)
    Policed(conform)    :             1166269/149282432            4997
    Policed(exceed)     :          1004175073/128534409344         4315338
    Policed(violate)    :                   0/0                    0
    Policed and dropped :          1004175073/128534409344
```

```
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :              49159/6292352              0
    Transmitted         :              49159/6292352              0
    Total Dropped       :                  0/0                    0
Policy Bag Stats time: 1573831087338  [Local Time: 11/15/19 15:18:07.338]
```

Related Commands hw-module profile qos shared-policer-per-class-stats

Prioritize Traffic during Oversubscription

Table 26: Feature History Table

Feature Name	Release Information	Feature Description
Prioritize Traffic during Oversubscription	Release 7.7.1	If the ASIC packet processing capacity is overloaded with continuous flow of packets, it's called oversubscription and it results in packet drops at the interface level. By enabling prioritization of packets during oversubscription, you can protect the high priority traffic. This feature introduces the hw-module oversubscription command.

An application-specific integrated circuit (ASIC) processes the traffic in the router. When the ingress packet enters the network interface in the ASIC, there's a small buffer to converge and process the packets. Independent of the traffic size, there's only one processing point in the ASIC to process the packets called Packet Processing Unit (PPU).

This PPU capacity is about the number of packets being processed per second by the ASIC, based on its packet processing power. The packet processing capacity is independent of the packet size, either it's 100 bytes or 5000 bytes, it processes only the header part, which is a fixed size, independent of the packet size.

Oversubscription occurs when the packet processing capacity is affected, and unable to process packets due to overloading of packets at PPU. But there's still some buffer available, which accommodates the logic of allowing traffic, based on their priority.

The network interface or ports in ASIC are grouped as port groups or quads. Depending upon the ASIC, supported port types include 1G, 10G, 25G, 40G, 50G, 100G, 200G, and 400G. The priority of packets are based on Class of Service (CoS) values.

The CoS value range is 0–7, where 6 and 7 are high priority by default. You can configure priority between 0–5. For example, consider interface X, with priority of 5, and if interface X belongs to a particular quad, then all the other interfaces belonging to that quad is also defined with the same priority of 5. So, if the CoS value is defined on an interface with high priority, then all the interface belonging to that quad becomes prioritized.

For untagged packets (without any VLAN tags) the interface priority is based on the interface name and it is considered with default CoS values as 6 and 7. Control plane packets always go through the highest priority.

Configuring Traffic Priorities on Tagged Interface

Perform this task to configure the traffic priorities using CoS values on tagged interface.

1. Creating egress policy-map that can be attached to one or more interfaces
2. Specifying the traffic class whose policy has to be created
3. Specifying priority to the traffic class
4. Creating H-QoS profile with ingress Layer 2 Access Control List (L2 ACL)
5. Specifying priority to the interface using oversubscription

Configuration Example

```
Router#configure
Router(config)#policy-map egress
Router(config-pmap)#class qos1
Router(config-pmap-c)#priority level 1
Router(config-pmap-c)#low-latency-profile
Router(config-pmap-c)#class class-default
Router(config-pmap-c)#end-policy-map
Router(config)#commit
Router(config)#end

Router#configure
Router(config)#hw-module profile qos hqos-enable
Router(config)#hw-module oversubscription
Router(config-oversubscription)#prioritize cos 1 interface tenGigE 0/0/0/0
Router(config-oversubscription)#commit
Wed Jul  6 11:07:45.608 UTC

NOTE: This CLI Config affects all interfaces in Quad
Check 'show oversubscription'for quad details
```

Running Configuration

```
policy-map egress
 class qos-1
  priority level 1
  low-latency-profile
 !
 class class-default
 !
  end-policy-map
!

hw-module profile qos hqos-enable
hw-module oversubscription
  prioritize cos 1 interface tenGigE 0/0/0/0

Wed Jul  6 11:07:45.608 UTC
```

```
NOTE: This CLI Config affects all interfaces in Quad
Check 'show oversubscription'for quad details
```

Verification

Verify the traffic priority configuration for both tagged interfaces, using the **show oversubscription** command.

```
Router#show oversubscription
Wed Jul  6 11:09:03.255 UTC
Interface              Quad       Prioritization
TenGigE0/0/0/0            4         cos [ 1 6 7]
TenGigE0/0/0/3            4         cos [ 1 6 7]
TenGigE0/0/0/2            4         cos [ 1 6 7]
TenGigE0/0/0/1            4         cos [ 1 6 7]
```

Configuring Traffic Priorities on Untagged Interface

Perform this task to configure the traffic priorities using CoS values on tagged interface.

1. Creating H-QoS profile with ingress Layer 2 Access Control List (L2 ACL)
2. Specifying priority to the interface using oversubscription

Configuration Example

```
Router#configure
Router(config)#hw-module profile qos hqos-enable
Router(config)#hw-module oversubscription
Router(config-oversubscription)#prioritize untagged interface tenGigE 0/0/0/10
Router(config-oversubscription)#commit
Wed Jul  6 11:07:45.608 UTC

NOTE: This CLI Config affects all interfaces in Quad
Check 'show oversubscription'for quad details
```

Running Configuration

```
hw-module oversubscription
hw-module profile qos hqos-enable
  prioritize untagged interface tenGigE 0/0/0/10

Wed Jul  6 11:07:45.608 UTC

NOTE: This CLI Config affects all interfaces in Quad
Check 'show oversubscription'for quad details
```

Verification

Verify the traffic priority configuration for untagged interfaces, using the **show oversubscription** command.

```
Router#show oversubscription
Wed Jul  6 11:09:03.255 UTC
Interface              Quad       Prioritization
TenGigE0/0/0/10           6         cos [untagged 6 7]
TenGigE0/0/0/11           6         cos [untagged 6 7]
TenGigE0/0/0/9            6         cos [untagged 6 7]
TenGigE0/0/0/8            6         cos [untagged 6 7]
```

Limitations

- Only up to 20% of the high priority traffic is allowed per quad.
- Oversubscription configuration is supported only on ports with bandwidth greater than 10G.
- Oversubscription is supported only for packets with VLAN header and control frames.
- You can enable oversubscription only at quad levels, based on the port speed in each quad.

References for Modular QoS Congestion Management

Committed Bursts

The committed burst (bc) parameter of the police command implements the first, conforming (green) token bucket that the router uses to meter traffic. The bc parameter sets the size of this token bucket. Initially, the token bucket is full and the token count is equal to the committed burst size (CBS). Thereafter, the meter updates the token counts the number of times per second indicated by the committed information rate (CIR).

The following describes how the meter uses the conforming token bucket to send packets:

- If sufficient tokens are in the conforming token bucket when a packet arrives, the meter marks the packet green and decrements the conforming token count by the number of bytes of the packet.
- If there are insufficient tokens available in the conforming token bucket, the meter allows the traffic flow to borrow the tokens needed to send the packet. The meter checks the exceeding token bucket for the number of bytes of the packet. If the exceeding token bucket has a sufficient number of tokens available, the meter marks the packet.

  Green and decrements the conforming token count down to the minimum value of 0.

  Yellow, borrows the remaining tokens needed from the exceeding token bucket, and decrements the exceeding token count by the number of tokens borrowed down to the minimum value of 0.
- If an insufficient number of tokens is available, the meter marks the packet red and does not decrement either of the conforming or exceeding token counts.

Note When the meter marks a packet with a specific color, there must be a sufficient number of tokens of that color to accommodate the entire packet. Therefore, the volume of green packets is never smaller than the committed information rate (CIR) and committed burst size (CBS). Tokens of a given color are always used on packets of that color.

Excess Bursts

The excess burst (be) parameter of the police command implements the second, exceeding (yellow) token bucket that the router uses to meter traffic. The exceeding token bucket is initially full and the token count is

equal to the excess burst size (EBS). Thereafter, the meter updates the token counts the number of times per second indicated by the committed information rate (CIR).

The following describes how the meter uses the exceeding token bucket to send packets:

- When the first token bucket (the conforming bucket) meets the committed burst size (CBS), the meter allows the traffic flow to borrow the tokens needed from the exceeding token bucket. The meter marks the packet yellow and then decrements the exceeding token bucket by the number of bytes of the packet.
- If the exceeding token bucket does not have the required tokens to borrow, the meter marks the packet red and does not decrement the conforming or the exceeding token bucket. Instead, the meter performs the exceed-action configured in the police command (for example, the policer drops the packets).

Two-Rate Policer Details

The committed token bucket can hold bytes up to the size of the committed burst (bc) before overflowing. This token bucket holds the tokens that determine whether a packet conforms to or exceeds the CIR as the following describes:

- A traffic stream is conforming when the average number of bytes over time does not cause the committed token bucket to overflow. When this occurs, the token bucket algorithm marks the traffic stream green.
- A traffic stream is exceeding when it causes the committed token bucket to overflow into the peak token bucket. When this occurs, the token bucket algorithm marks the traffic stream yellow. The peak token bucket is filled as long as the traffic exceeds the police rate.

The peak token bucket can hold bytes up to the size of the peak burst (be) before overflowing. This token bucket holds the tokens that determine whether a packet violates the PIR. A traffic stream is violating when it causes the peak token bucket to overflow. When this occurs, the token bucket algorithm marks the traffic stream red.

For example, if a data stream with a rate of 250 kbps arrives at the two-rate policer, and the CIR is 100 kbps and the PIR is 200 kbps, the policer marks the packet in the following way:

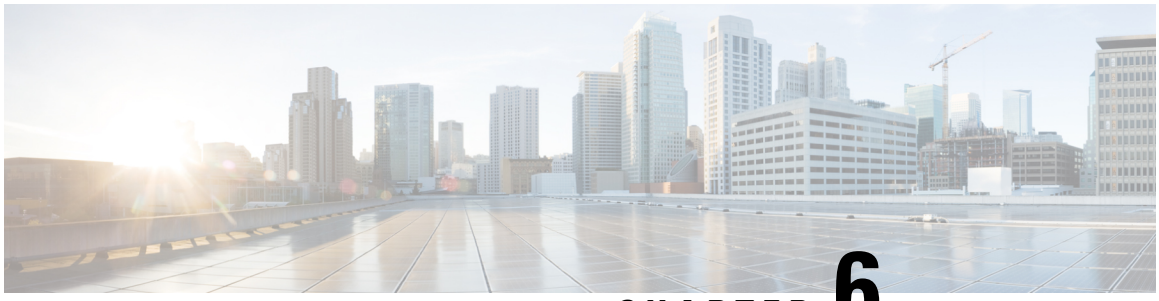
- 100 kbps conforms to the rate
- 100 kbps exceeds the rate
- 50 kbps violates the rate

The router updates the tokens for both the committed and peak token buckets in the following way:

- The router updates the committed token bucket at the CIR value each time a packet arrives at the interface. The committed token bucket can contain up to the committed burst (bc) value.
- The router updates the peak token bucket at the PIR value each time a packet arrives at the interface. The peak token bucket can contain up to the peak burst (be) value.
- When an arriving packet conforms to the CIR, the router takes the conform action on the packet and decrements both the committed and peak token buckets by the number of bytes of the packet.
- When an arriving packet exceeds the CIR, the router takes the exceed action on the packet, decrements the committed token bucket by the number of bytes of the packet, and decrements the peak token bucket by the number of overflow bytes of the packet.

- When an arriving packet exceeds the PIR, the router takes the violate action on the packet, but does not decrement the peak token bucket.

See Two-Rate Policer, on page 128.



CHAPTER 6

Configuring Modular QoS on Link Bundles

This chapter covers the following topics:

- QoS on Link Bundles, on page 145

QoS on Link Bundles

A bundle is a group of one or more ports that are aggregated together and treated as a single link. The router supports Ethernet interfaces and VLAN interfaces (bundle sub-interfaces) bundles. All QoS features currently supported on physical interfaces, are also supported on all link bundle interfaces. Applying QoS on bundle members is not supported.

Restrictions for Link Bundles

- Only Ethernet link bundling is supported.
- A bundle interface can only contain physical interface.
- All links within a single bundle must be configured either to run 802.3ad (LACP) or Etherchannel (non-LACP). Mixed links within a single bundle are not supported.
- MAC accounting is not supported on Ethernet link bundles.
- Maximum number of links supported in each link bundle is 64.
- The maximum number of link bundles supported is 128.
- On NCS-57C3-MOD-SYS routers, configuring an egress QoS service policy on a Bundle-Ether interface with a shaper burst size smaller than the interface MTU can cause the bundle interface to go down and stop forwarding traffic.

  In this condition, router reloads, line card reloads, and removal or reapplication of the service policy do not restore the bundle interface. Service is restored only after configuring the shaper burst size to a value greater than the bundle interface MTU.

Load Balancing

Load balancing function is a forwarding mechanism to distribute traffic over multiple links based on Layer 3 routing information in the router. Per-destination load balancing isonly supported on the router, where the router is allowed to distribute packets over one of the links in the bundle. When the per-destination load

balancing is enabled, all packets for a certain source-destination pair goes through the same link, though there are multiple links available. In other words, per-destination load balancing can ensure that packets for a certain source-destination pair could arrive in order.

Layer 3 Load Balancing on Link Bundles

Layer 3 load balancing for link bundles is done on Ethernet Flow Points (EFPs) and is based on the IPv4 source and destination addresses in the packet. When Layer 3 service-specific load balancing is configured, all egress bundles are load balanced based on the IPv4 source and destination addresses. When packets do not have IPv4 addresses, default load-balancing (based on the MAC SA/DA fields in the packet header) is used.

Configure QoS on Link Bundles

QoS is configured on link bundles in the same way that it is configured on individual interfaces.

Guidelines

- When a QoS policy is applied on a bundle in the egress direction, it's also applied at each member interface.
- When a QoS policy is applied on a bundle (ingress direction), it's replicated at each NPU core.
- If a QoS policy is not applied to a bundle interface, both the ingress and egress traffic use the default queue of the per link member port.
- The shape rate that is specified in the bundle policy-map is not an aggregate for all bundle members. The shape rate applied to the bundle depends on the load balancing of the links. For example, if a policy map with a shape rate of 10 Mbps is applied to a bundle with two member links, and if the traffic is always load-balanced to the same member link, then an overall rate of 10 Mbps applies to the bundle. However, if the traffic is load-balanced evenly between the two links, the overall shape rate for the bundle becomes 20 Mbps.
- If a member is deleted from a bundle, the total bundle statistics changes because the statistics that belongs to the detached link is lost.
- The QoS policy that is applied on bundle is inherited to all its member links and the reference bandwidth that is used to calculate shaper/bandwidth is applied as per the physical member interface bandwidth, and not the bundle as a whole.

Configuration Example

You have to accomplish the following to complete the QoS configuration on link bundles:

Note The policy works only if it is applied on the ingress direction. The egress is supported on COS, DEI and MPLS exp marking. So the below policy may not work when it is applied on egress.

1. Creating a class-map
2. Creating a policy-map and specifying the respective class-map
3. Specifying the action type for the traffic

Refer Attach a Traffic Policy to an Interface, on page 20 for details on step 1, 2 and 3.

4. Creating a link bundle
5. Applying traffic policy to the link bundle

```
/* Configure Ether-Bundle and apply traffic policy */
Router(config)# interface Bundle-Ether 12000
Router(config-if)# mtu 9100
Router(config-if)# service-policy input ingress
Router(config-if)# service-policy output egress
Router(config-if)# ipv4 address 100.12.0.0 255.255.255.254
Router(config-if)# bundle maximum-active links 64
Router(config-if)# commit
```

Running Configuration

This example shows how a traffic policy is applied on an Ethernet link bundle. The policy is applied to all interfaces that are members of the Ethernet link bundle.

```
/* Policy-map */

policy-map ingress
 class inet4-classifier-af1
  set qos-group 1
 !
 class inet4-classifier-af2
  set qos-group 2
 !
 class inet4-classifier-af3
  set qos-group 3
 !
 class inet4-classifier-af4
  set qos-group 4
 !
 class inet4-classifier-be1
  set qos-group 5
 !
 class inet4-classifier-nc1
  set qos-group 6
 !
 class class-default
 !
 end-policy-map
!

/* Ether Bundle */
interface Bundle-Ether12000
 mtu 9100
 service-policy input ingress
 service-policy output egress
 ipv4 address 100.12.0.0 255.255.255.254
 load-interval 30
 flow ipv4 monitor FMM-V4 sampler SM ingress
 flow ipv6 monitor FMM-V6 sampler SM ingress
 flow mpls monitor FMM-MPLS sampler SM ingress
 ipv4 access-group IPV4ACL_101 ingress
 ipv6 access-group IPV6ACL_101 ingress
!
```

Verification

- Verify that the bundle status is UP.

```
router# show bundle bundle-ether 1200
Wed Dec 16 19:55:49.974 PST

Bundle-Ether12000
  Status:                                  Up
  Local links <active/standby/configured>:   35 / 0 / 35
  Local bandwidth <effective/available>:     3500000000 (3500000000) kbps
  MAC address (source):                    ea3b.745f.c4b0 (Chassis pool)
  Inter-chassis link:                      No
  Minimum active links / bandwidth:        1 / 1 kbps
  Maximum active links:                    64
  Wait while timer:                        2000 ms
  Load balancing:                          Default
  LACP:                                    Operational
    Flap suppression timer:                Off
    Cisco extensions:                      Disabled
    Non-revertive:                         Disabled
  mLACP:                                   Not configured
  IPv4 BFD:                                Not configured

  Port                  Device           State        Port ID         B/W, kbps
  --------------------  ---------------  -----------  --------------  ----------
  Hu0/4/0/0             Local            Active       0x8000, 0x0009   100000000
      Link is Active
  Hu0/4/0/1             Local            Active       0x8000, 0x000a   100000000
      Link is Active
- - -
- - -
  Hu0/4/0/35            Local            Active       0x8000, 0x002b   100000000
      Link is Active
```

- Verify the bundle statistics:

```
router# show policy-map interface bundle-ether 12000

Bundle-Ether12000 input: ingress

Class inet4-classifier-af1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          4647401962/21236124455654       26403040
    Transmitted         :          4647401962/21236124455654       26403040
    Total Dropped       :                   0/0                    0
Class inet4-classifier-af2
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          4502980177/20576584333939       25571493
    Transmitted         :          4502980177/20576584333939       25571493
    Total Dropped       :                   0/0                    0
Class inet4-classifier-af3
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          4647404125/21236213667880       26389086
    Transmitted         :          4647404125/21236213667880       26389086
    Total Dropped       :                   0/0                    0
Class inet4-classifier-af4
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          9291188840/42456120548683       52771168
    Transmitted         :          9291188840/42456120548683       52771168
    Total Dropped       :                   0/0                    0
Class inet4-classifier-be1
```

```
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          4647413429/21235847852686       26393414
    Transmitted         :          4647413429/21235847852686       26393414
    Total Dropped       :                   0/0                    0
Class inet4-classifier-nc1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          9294887621/42473100149807       52778258
    Transmitted         :          9294887621/42473100149807       52778258
    Total Dropped       :                   0/0                    0

Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                   0/0                    0
    Transmitted         :                   0/0                    0
    Total Dropped       :                   0/0                    0

Bundle-Ether12000 output: egress

Class c1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :         16665494532/75878118942463       8760591
    Transmitted         :         16655834643/75834136022017       8760591
    Total Dropped       :             9659889/43982920446          0
  Queueing statistics
    Queue ID                              : None (Bundle)
    Taildropped(packets/bytes)            : 9659889/43982920446
Class c2
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :         16665421959/75877849543188       8718687
    Transmitted         :         16665421959/75877849543188       8718687
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                              : None (Bundle)
    Taildropped(packets/bytes)            : 0/0
Class c3
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :         16665247833/75877509455458       8703470
    Transmitted         :         16665187414/75877234624197       8703470
    Total Dropped       :               60419/274831261            0
  Queueing statistics
    Queue ID                              : None (Bundle)
    Taildropped(packets/bytes)            : 60419/274831261
Class c4
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :         33330896131/151755393012945      17470745
    Transmitted         :         33330745421/151754709368565      17470745
    Total Dropped       :              150710/683644380            0
  Queueing statistics
    Queue ID                              : None (Bundle)
    Taildropped(packets/bytes)            : 150710/683644380
Class c5
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :         16878910340/76849791869834       8833394
    Transmitted         :         16878849464/76849514633309       8833394
    Total Dropped       :               60876/277236525            0
  Queueing statistics
    Queue ID                              : None (Bundle)
    Taildropped(packets/bytes)            : 60876/277236525
Class c6
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :         33330898844/151756094112925      17456785
    Transmitted         :         33330752668/151755427708382      17456785
    Total Dropped       :              146176/666404543            0
  Queueing statistics
```

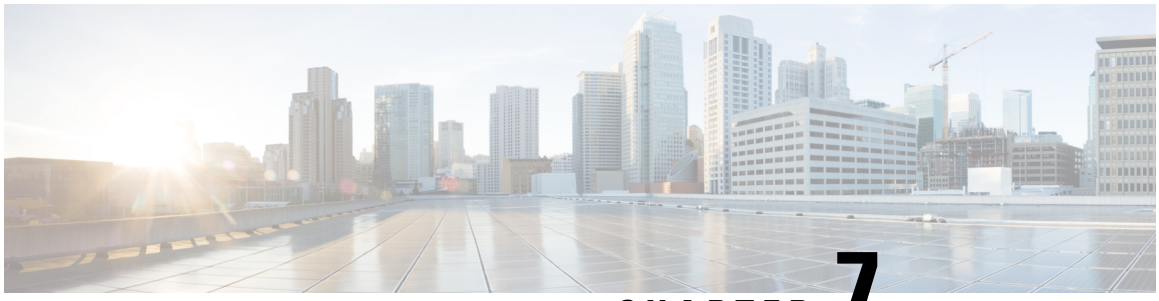
```
    Queue ID                                : None (Bundle)
    Taildropped(packets/bytes)              : 146176/666404543
Class c7
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :              244106/79922040             74
    Transmitted         :              244106/79922040             74
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                                : None (Bundle)
    Taildropped(packets/bytes)              : 0/0
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :        267075066180/1215993441123215     139917482
    Transmitted         :        267075066180/1215993441123215     139917482
    Total Dropped       :                   0/0                    0
  Queueing statistics
    Queue ID                                : None (Bundle)
    Taildropped(packets/bytes)              : 0/0
```

Related Topics

- QoS on Link Bundles, on page 145

Associated Commands

- bundle maximu-active links
- interface Bundle-Ether



CHAPTER 7

Configuring Hierarchical Modular QoS

Hierarchical QoS (H-QoS) is a QoS model that enables you to specify QoS behavior at multiple levels of hierarchy. This chapter provides information about this feature and the different steps involved in configuring it.

This chapter covers the following topics:

- Overview of Hierarchical Modular QoS, on page 151
- Restrictions, Limitations, and Requirements for Configuring H-QoS, on page 152
- Configuring Hierarchical Queuing, on page 153
- Bandwidth Remaining Ratio at Parent Level, on page 159
- Conform Aware Hierarchical Policy Overview, on page 162

Overview of Hierarchical Modular QoS

Table 27: Feature History Table

Feature Name	Release Information	Feature Description
Scale Limits for H-QoS on Cisco NCS 540-24Q8L2DD-SYS Routers	Release 7.6.1	With this release, 750 egress interfaces and subinterfaces are available for configuring QoS policies in the H-QoS model on Cisco NCS 540-24Q8L2DD-SYS routers. You can now achieve higher egress policy-map scales because of the increased number of VOQs available per subinterface. In earlier releases, 250 egress interfaces and subinterfaces were available for configuring QoS policies in the H-QoS model.

Hierarchical QoS (H-QoS) allows you to specify QoS behavior at multiple policy levels, which provides a high degree of granularity in traffic management.

H-QoS is applied on the router interface using nested traffic policies. The first level of traffic policy, the parent traffic policy, is used for controlling the traffic at the main interface or sub-interface level. The second level of traffic policy, the child traffic policy, is used for more control over a specific traffic stream or class. The child traffic policy, is a previously defined traffic policy, that is referenced within the parent traffic policy using the **service-policy** command.

Two-level H-QoS is supported on both ingress and egress directions on all line cards and on physical or bundle main interfaces and sub-interfaces.

Three-level Hierarchical QoS (H-QoS) enables enforcement of class/service, group/ Ethernet Flow Point (EFP), and port level SLAs. You can apply regular two-level egress H-QoS policies on the sub-interfaces to achieve class and EFP SLAs at child and parent levels. In addition, you can apply a port shaper policy on the main interface to achieve an aggregated port level SLA in a 1+2 H-QoS or three-level H-QoS model.

An important point to note is that before Release 6.6.25 (where the three-level H-QoS capability was introduced), when you applied class-default shaper on a main interface, it was enforced *only* on the traffic going through the main interface. With three-level HQoS, a class default shaper that is applied on the main interface is considered as a port shaper and enforced on *all* traffic going out of that physical port. The advantage of three-level H-QoS is that the parent shaper on the sub-interfaces is allowed to oversubscribe, thus enabling best effort sharing of the aggregate port shaper at the third level.

Restrictions, Limitations, and Requirements for Configuring H-QoS

There are some key restrictions, limitations, and requirements for configuring H-QoS on your router to ensure optimal performance and compliance.

Parent Traffic Policy Restrictions

- The parent traffic policy only supports the traffic class of type class-default.
- The parent traffic policy only supports the class-action shape, and no other queuing action can be configured in it.
- For congestion avoidance and management, the traffic shaper in the parent traffic policy calculates the queue limit and drop priority

Child Policy Requirements

- While configuring on the router, it is mandatory that the priority class must have a traffic shaper in the child traffic policy.
- The sum of the bandwidth of the child policies must be less than the parent policy's traffic shaper.

H-QoS Profile Limitations

- H-QoS profile and ingress peering profile don't work simultaneously. Hence, features requiring a peering profile also do not work with the H-QoS profile enabled.
- The PBTS feature does not work when the H-QoS profile is enabled. This is due to TCAM limitations.
- A maximum of 896 bundle subinterfaces are supported in the system, even if there are no QoS policies applied. This is due to an internal LAG_ID resource consumption in H-QoS profile mode for bundle subinterfaces with or without QoS policies being applied.
- A maximum of 4 priority levels are supported in H-QoS profile mode, unlike the default mode where 7-priority levels are supported. The restriction also applies to physical and bundle main interface policies where 7-level priorities were previously used in non-H-QoS profile mode.

Bandwidth Configuration Limitations

- Bandwidth and Bandwidth remaining configurations are not supported simultaneously within the same policy-map. If a class has bandwidth (CIR), other classes must also have only bandwidth configuration. If a class-map has bandwidth remaining percent/ratio (EIR), other classes should also have only the bandwidth remaining configuration. Shaping is applied on any class.
- In H-QoS mode, if multiple queues are configured with BRR and there is high congestion in the LP (low-priority) queues, one BRR queue receives more credits than the others. The deviation is proportional to the congestion in the LP queues.
- The granularity of bandwidth or the bandwidth remaining ratio (BRR) is 1:64, compared to 1:4096 in non-H-QoS mode. Consequently, there may be differences in bandwidth performance accuracy depending on the values used.

Priority Classes Requirements

Priority classes must be rate-limited using a shaping configuration. The effective shaper value is used as the priority bandwidth reservation. The total priority bandwidth reservations across all subinterfaces and main interfaces must not exceed the network interface (NIF) port speed to prevent oversubscription of priority traffic. Non-priority class rates and parent shaping can be oversubscribed.

Multicast Traffic Restriction

Filtering egress IPv4 and IPv6 multicast traffic is not supported when H-QoS is configured on the router.

Three-Level H-QoS Restrictions

- There is no support for bandwidth action at the EFP parent level. All EFP and subinterface policies get a fair share of the port shaper.
- Three-level H-QoS does not apply to ingress policies or egress marking policies.
- Executing clear qos counters on the main interface clears only the main interface policy statistics. Use the all option to clear all sub-interface statistics, or alternatively, clear the sub-interface policy statistics individually.
- Main interface policy statistics do not reflect the subinterface packet or byte counters, although the port shaper is enforced on all logical ports for a given physical interface. The subinterface policy-map statistics reflect the transmitted and dropped packet or byte count post-port shaper enforcement.

Configuring Hierarchical Queuing

Before you configure H-QoS, you must enable the H-QoS profile on the router. After enabling H-QoS profile, reload the router, as shown in the following configuration.

```
admin
hw-module location all reload
Router# configure
Router(config)# hw-module profile qos hqos-enable
Router(config)# commit
Router# admin
sysadmin-vm:0_RP0# hw-module location all reload
```

The steps that are involved in configuring hierarchical queuing are as follows:

1. Configure a class-map.
2. Configure a child traffic policy using the class-map that was configured in the previous step.
3. Configure a parent traffic policy and add the child traffic policy in it.

The parent traffic policy is the H-QoS traffic policy and it can be applied on physical or bundle main interfaces and sub-interfaces.

Configuration Example

Configuration of a class-map is as follows:

```
Router# configure
Router(config)# class-map match-any tc2
Router(config-cmap)# match traffic-class 1
Router(config-cmap)# end-class-map
Router(config)# commit
```

Configuration of a child traffic policy is as follows:

```
Router# configure
Router(config)# policy-map child
Router(config-pmap)# class tc2
Router(config-pmap-c)# shape average percent 20
Router(config-pmap-c)# exit
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average percent 1
Router(config-pmap)# end-policy-map
Router(config)# commit
```

Configuration of a parent traffic policy is as follows:

```
Router# configure
Router(config)# policy-map parent
Router(config-pmap)# class class-default
Router(config-pmap-c)# service-policy child
Router(config-pmap-c)# shape average percent 50
Router(config-pmap)# end-policy-map
Router(config)# commit
```

Running Configuration

```
/* Configuration of a Class-map */
class-map match-any tc2
 match traffic-class 1
 end-class-map
!
/* Configuration of a Child Traffic Policy */
policy-map child
 class tc2
  shape average percent 20
 !
 class class-default
  shape average percent 1
 !
 end-policy-map
```

```
!
/* Configuration of a Parent Traffic Policy */
policy-map parent
 class class-default
  service-policy child
  shape average percent 50
 !
 end-policy-map
!
```

Applying the Parent Traffic Policy on a Main Interface

```
Router# configure
Router(config)# Interface TenGigE 0/0/0/10
Router(config-int)# service-policy output parent
Router(config-int)# commit
```

Applying the Parent Traffic Policy on a Sub-interface

```
Router# configure
Router(config)# Interface TenGigE 0/0/0/10.1
Router(config-int)# service-policy output parent
Router(config-int)# commit
```

Verification

Verify if the H-QoS traffic policy is applied correctly on the interface using the commands **show qos interface** *interface-name* **output**. In the following example, the **Level1 Class** gives information about the class-map that is associated with the parent traffic policy and the **Level2 Class** gives information about the class-maps that are associated with the child traffic policy.

```
RP/0/RP0/CPU0:ios#show qos interface ten0/0/0/10 output

NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/10 ifh 0x1e0  -- output policy
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           10000000 kbps
VOQ Base:                      1136
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                                =   class-default
Queue Max. BW.                              =   no max (50 %)
Queue Min. BW.                              =   0 kbps (default)
Inverse Weight / Weight                     =   0 / (BWR not configured)
   Level2 Class                                 =   tc2
   Egressq Queue ID                             =   1138 (LP queue)
   Queue Max. BW.                               =   1020015 kbps (20 %)
   Queue Min. BW.                               =   0 kbps (default)
   Inverse Weight / Weight                      =   1 / (BWR not configured)
   Guaranteed service rate                      =   1000000 kbps
   TailDrop Threshold                           =   1253376 bytes / 10 ms (default)
   WRED not configured for this class
   Level2 Class                                 =   class-default
   Egressq Queue ID                             =   1136 (Default LP queue)
   Queue Max. BW.                               =   50625 kbps (1 %)
   Queue Min. BW.                               =   0 kbps (default)
   Inverse Weight / Weight                      =   1 / (BWR not configured)
   Guaranteed service rate                      =   50000 kbps
```

```
   TailDrop Threshold                         =   62720 bytes / 10 ms (default)
   WRED not configured for this class
```

The statistics for the packets that have matched the different traffic classes of the parent and child traffic policies can be viewed using the command **show policy-map interface** *interface-name* **output**. Also, this command also shows the number of packets that are transmitted or dropped when the specified action is applied on the packets that have matched the respective traffic class.

```
Router# show policy-map interface ten0/0/0/10 output

TenGigE0/0/0/10 output: parent
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          2313578823/296138089344         8494665
    Transmitted         :           232805738/29799134464          854465
    Total Dropped       :          2080773085/266338954880         7640200
  Policy child Class tc2
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :          2313578823/296138089344         8494665
      Transmitted         :           232805738/29799134464          854465
      Total Dropped       :          2080773085/266338954880         7640200
    Queueing statistics
      Queue ID                              : 1138

      Taildropped(packets/bytes)            : 2080773085/266338954880
  Policy child Class class-default
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :                    0/0                   0
      Transmitted         :                    0/0                   0
      Total Dropped       :                    0/0                   0
    Queueing statistics
      Queue ID                              : 1136

      Taildropped(packets/bytes)            : 0/0
```

When using hierarchical policers, there is no independent set of hardware counters to store the parent policer statistics. Instead, parent policer statistics are manipulated in the software to be the sum of all child policers under the same policy-map.

This is shown in the following example where two streams of traffic, with CoS value of 1 and 2 are sent at a speed of 3.5 Gbps each.

```
/*Hierarchical Policy Map Configuration*/
==================================================
Router# show running-config policy-map Hingress
policy-map Hingress
 class class-default
  service-policy ingress
  police rate 5 gbps peak-rate 9 gbps
  !
 !
 end-policy-map
!
 /*Ingress Policy Map Configuration*/
====================================
Router#show running-config policy-map ingress
policy-map ingress
 class cos1
  set traffic-class 1
  police rate 5 gbps
  !
 !
 class cos2
```

```
  set traffic-class 2
  police rate 5 gbps
  !
 !
 class class-default
 !
 end-policy-map
!
 /*Policy Map applied at TenGigE0/0/0/6.100 Interface*/
=======================================================
Router#show policy-map interface tenGigE 0/0/0/6.100 input

TenGigE0/0/0/6.100 input: Hingress

Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :           856717937/109659895936         6683676
    Transmitted         :           856717937/109659895936         6683676
    Total Dropped       :                   0/0                    0
  Policing statistics                (packets/bytes)     (rate - kbps)
    Policed(conform)    :           856717937/109659895936         6683674
    Policed(exceed)     :                   0/0                    0
    Policed(violate)    :                   0/0                    0
    Policed and dropped :                   0/0

  Policy ingress Class cos1
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :           437826303/56041766784          3341838
      Transmitted         :           437826303/56041766784          3341838
      Total Dropped       :                   0/0                    0
    Policing statistics                (packets/bytes)     (rate - kbps)
      Policed(conform)    :           437826303/56041766784          3341838
      Policed(exceed)     :                   0/0                    0
      Policed(violate)    :                   0/0                    0
      Policed and dropped :                   0/0
      Policed and dropped(parent policer)  : 0/0

  Policy ingress Class cos2
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :           418891634/53618129152          3341838
      Transmitted         :           418891634/53618129152          3341838
      Total Dropped       :                   0/0                    0
    Policing statistics                (packets/bytes)     (rate - kbps)
      Policed(conform)    :           418891634/53618129152          3341838
      Policed(exceed)     :                   0/0                    0
      Policed(violate)    :                   0/0                    0
      Policed and dropped :                   0/0
      Policed and dropped(parent policer)  : 0/0

  Policy ingress Class class-default
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :                   0/0                    0
      Transmitted         :                   0/0                    0
      Total Dropped       :                   0/0                    0
Policy Bag Stats time: 0
Policy Bag Stats time: 0
```

Configuration Example for Three-Level H-QoS

To configure three-level H-QoS:

1. Configure the port shaper or EFP group shaper.

2. Configure the two-level H-QoS policy with an EFP parent shaper and class or service level actions
3. Enable the port or EFP group shaper on the main interface to become the root policy.
4. Enable the two-level H-QoS policy on each EFP instance, thus providing a three-level hierarchy of service, EFP and EFP group or port SLAs.

A sample configuration for three-level H-QoS is as follows.

```
policy-map port_shaper
 class class-default
  shape average 6 gbps
 !
 end-policy-map
!

policy-map efp_policy
 class class-default
  service-policy efp_policy_child
  shape average 4 gbps
 !
 end-policy-map

!
policy-map efp_policy_child
 class tc1
  shape average 50 mbps
  priority level 1
 !
 class tc2
  bandwidth percent 50
 !
 class tc3
  bandwidth percent 30
 !
 class class-default
 !
 end-policy-map
!

interface TenGigE0/5/0/4
 service-policy output port_shaper
!

interface TenGigE0/5/0/4.1
 service-policy output efp_policy
 encapsulation dot1q 11
!

interface TenGigE0/5/0/4.2
 service-policy output efp_policy
 encapsulation dot1q 12
!
```

Verification

Run the **show policy-map interface** command in XR EXEC mode to display the packet/byte count and rate post port shaper enforcement on each of the sub-interface/EFP policies.

Bandwidth Remaining Ratio at Parent Level

Table 28: Feature History Table

Feature Name	Release Information	Feature Description
Bandwidth Remaining Ratio at Parent Level	Release 7.10.1	You can now provide preference to a specific user assigned to a subinterface. With this feature, the bandwidth remaining ratio (BWRR) is applied at the parent traffic policy level to control the low-priority traffic at the subinterface level. During congestion, the subinterface with the higher BWRR gets more bandwidth allocation that allows traffic to flow. Earlier, the BWRR in H-QoS was supported only at the child policy level. Supported on the following Cisco NCS 540 variants: • N540-ACC-SYS • N540X-ACC-SYS • N540-24Z8Q2C-SYS

Hierarchical QoS (H-QoS) allows you to specify QoS behavior at multiple policy levels, and starting from Cisco IOS XR Release 7.10.1 you can allocate weights corresponding to the bandwidth remaining ratio (BWRR) at the parent traffic policy level. The bandwidth remaining option specifies a weight at parent policy level for the subinterface. After the priority-queue is serviced, the leftover bandwidth is distributed as per BWRR. For more information on BWRR, see Bandwidth Remaining.

You can configure the BWRR on parent traffic policy on the subinterfaces belonging to the same physical interface. You can configure the parent policy of the subinterfaces, with different weights. Subinterfaces with more weight get more bandwidth.

For example, consider the use case where three subinterfaces belonging to same physical interface (1G) configured with H-QoS policy. All three subinterface policies are configured with 500 Mbps parent shaper each. When all three try to use the assigned bandwidth, it requires 1.5G in total bandwidth exceeding 1G and it results in congestion. If on these three subinterfaces the BWRR weights are configured as 20, 40, and 60 each, then the subinterface with higher BWRR allows more traffic to flow.

Restrictions and Limitations

- Weight sharing is applied only on low-priority traffic. For more information on sharing of weights, see Shared Policer.

Configuring BWRR at H-QoS

Before you configure H-QoS, you must enable the H-QoS profile on the router. After enabling H-QoS profile, reload the router, for more information see Configuring Hierarchical Queuing.

In the following example you'll create three traffic policies with BWRR as 20, 40, and 60, respectively. These policies will be applied to a 10G subinterface, with BWRR applied to each parent 1, 2, and 3. The subinterface parent 3, has more weightage of 60 than the other two parent subinterfaces.

Configuration of a parent traffic policy 1 is as follows:

```
Router# configure
Router(config)# policy-map hqos_bw_remaining_ratio_parent_1
Router(config-pmap)# class class-default
Router(config-pmap-c)# service-policy hqos_child_1
Router(config-pmap-c)# bandwidth remaining ratio 20
Router(config-pmap)# exit
Router(config)# commit
```

Configuration of a parent traffic policy 2 is as follows:

```
Router# configure
Router(config)# policy-map hqos_bw_remaining_ratio_parent_2
Router(config-pmap)# class class-default
Router(config-pmap-c)# service-policy hqos_child_2
Router(config-pmap-c)# bandwidth remaining ratio 40
Router(config-pmap)# exit
Router(config)# commit
```

Configuration of a parent traffic policy 3 is as follows:

```
Router# configure
Router(config)# policy-map hqos_bw_remaining_ratio_parent_3
Router(config-pmap)# class class-default
Router(config-pmap-c)# service-policy hqos_child_3
Router(config-pmap-c)# bandwidth remaining ratio 60
Router(config-pmap)# exit
Router(config)# commit
```

In the following example, the policy maps are assigned to the subinterfaces.

```
Router# configure
Router(config)# interface TenGigE0/0/0/20
Router(config-if)# service-policy output shaper_verification_parent
Router(config-if)# interface TenGigE0/0/0/20.103
Router(config-subif)# service-policy output hqos_bw_remaining_ratio_parent_1
Router(config-subif)# interface TenGigE0/0/0/20.1
Router(config-subif)# service-policy output hqos_bw_remaining_ratio_parent_2
Router(config-subif)# interface TenGigE0/0/0/20.101
Router(config-subif)# service-policy output hqos_bw_remaining_ratio_parent_3
```

Running Configuration

```
/* Configuration of a Parent Traffic Policy 1*/
hqos_bw_remaining_ratio_parent_1
 class class-default
  service-policy hqos_child_1
  bandwidth remaining ratio 20
 !
 end-policy-map
!
/* Configuration of a Parent Traffic Policy 2*/
hqos_bw_remaining_ratio_parent_2
 class class-default
  service-policy hqos_child_2
```

```
  bandwidth remaining ratio 40
 !
 end-policy-map
!
/* Configuration of a Parent Traffic Policy 3*/
hqos_bw_remaining_ratio_parent_3
 class class-default
  service-policy hqos_child_3
  bandwidth remaining ratio 60
 !
 end-policy-map
!
```

Verification

Verify the H-QoS traffic policy that is applied on the interface using the commands **show qos interface** interface-name **output**.

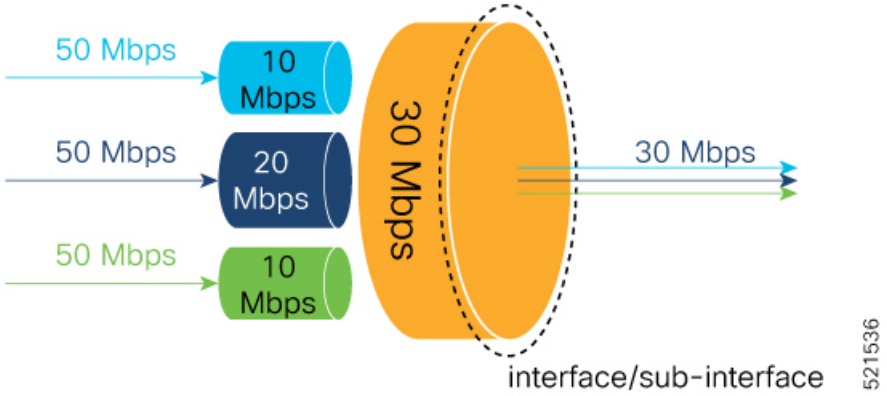
The following example shows the output of all three parent traffic policies, but the subinterface **policy-map hqos_bw_remaining_ratio_parent_3** gets more bandwidth allocation that allows traffic to flow.

```
RP/0/RP0/CPU0:ios#show qos interface TenGigE0/0/0/20.103 output
NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/20.103 ifh 0x8152  -- output policy
NPU Id:                        0
Total number of classes:       9
Interface Bandwidth:           10000000 kbps
Policy Name:                   hqos_bw_remaining_ratio_parent_1
SPI Id:                        0x0
VOQ Base:                      1320
PFC enabled:                   0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class = class-default
Queue Max. BW. = no max (default)
Queue Min. BW. = 0 kbps (default)
Inverse Weight / Weight = 1 / (20)

RP/0/RP0/CPU0:ios#show qos interface TenGigE0/0/0/20.1 output
NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/20.1 ifh 0x8152  -- output policy
NPU Id:                        0
Total number of classes:       9
Interface Bandwidth:           10000000 kbps
Policy Name:                   hqos_bw_remaining_ratio_parent_2
SPI Id:                        0x0
VOQ Base:                      1320
PFC enabled:                   0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class = class-default
Queue Max. BW. = no max (default)
Queue Min. BW. = 0 kbps (default)
Inverse Weight / Weight = 1 / (40)

RP/0/RP0/CPU0:ios#show qos interface ten0/0/0/20.101 output
NOTE:- Configured values are displayed within parentheses
Interface TenGigE0/0/0/20.101 ifh 0x81a0 -- output policy
NPU Id:                          0
Total number of classes:         9
Interface Bandwidth:             10000000 kbps
Policy Name:                   hqos_bw_remaining_ratio_parent_3 /* BWRR with more weightage
```

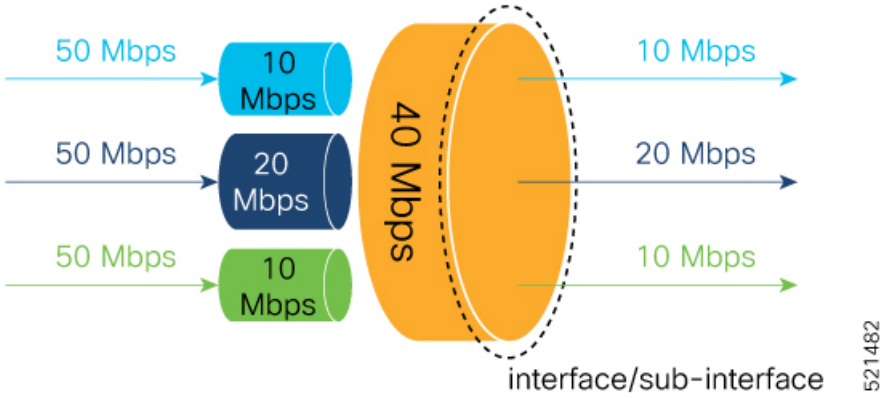
```
 */
SPI Id:                          0x0
VOQ Base:                        1296
PFC enabled:                     0
Accounting Type:                 Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class = class-default
Queue Max. BW. = no max (default)
Queue Min. BW. = 0 kbps (default)
Inverse Weight / Weight = 1 / (60) /* BWRR with more weightage */
```

Conform Aware Hierarchical Policy Overview

Hierarchical QoS (H-QoS), while allowing for granular and multi-level management of traffic, does not allow for conform traffic from a child-level policy to a parent-level policy to get priority. This means that in case of excess traffic, the parent policer drops conform traffic packets from the child level as well.



The conform-aware hierarchical policy feature enables the parent-level policy to prioritize conform traffic from child-level policy over exceed and violate traffic.



Here is how it works: the child-level policer initially marks its packets as red, yellow, or green. Packets are marked based on the committed information rate (CIR) value and the two associated burst sizes - committed burst size (CBS) and excess burst size (EBS). If a packet does not exceed the CBS, it is marked as conformed packet (green). The packet is marked as exceeded if it exceeds CBS, but not the EBS (yellow). If it exceeds the EBS as well, it is marked as violate packet (red).

When the packets arrive at the parent level policer (which is color aware), the policer prioritizes the packets marked green over the packets marked yellow. After all the conform traffic (green) is transmitted and there are tokens available still, the yellow packets are transmitted next, instead of being marked as violate traffic (red).

To enable the conform-aware hierarchical policy feature run the command.

```
hw-module profile qos conform-aware-policer
```

Configuring Conform Aware Hierarchy Policy

To enable and configure shared policer:

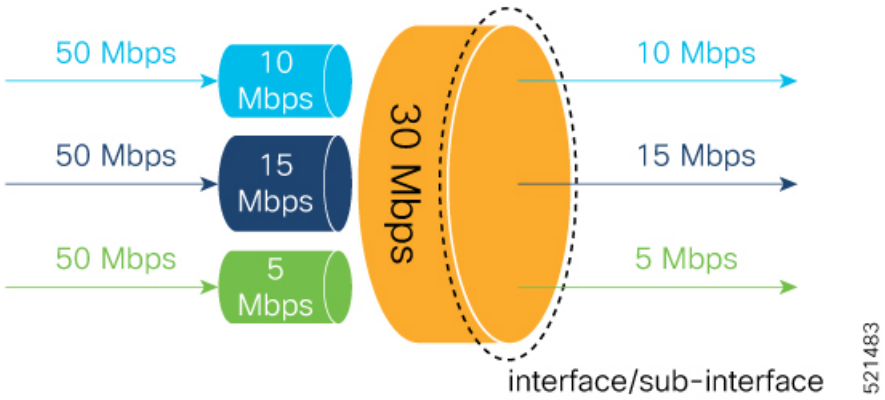
1. Run the **hw-module profile qos conform-aware-policer** command.
2. Reload the affected line card.
3. Configure class maps to be used for matching packets to the class specified.
4. Create a child policy map.
5. Configure traffic policing for the classes in the child policy map.
6. Attach the child policy-map to the parent's class-default class.
7. Configure traffic policing for the parent policy map.

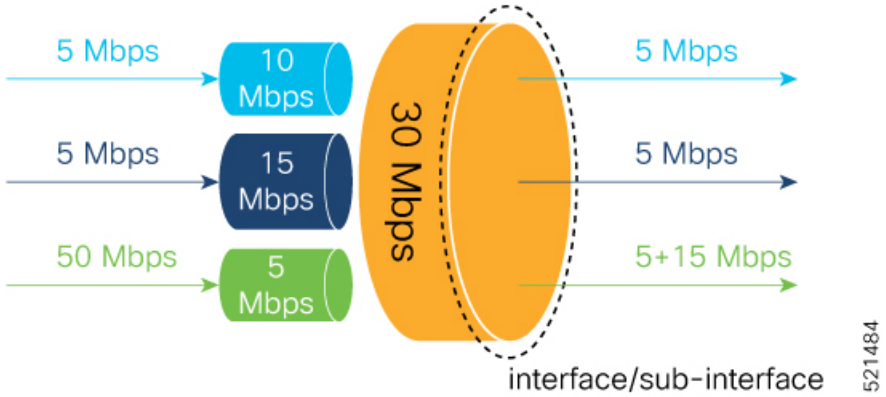
```
RP/0/RP0/CPU0:ios(config)#hw-module profile qos conform-aware-policer
/* To activate this profile, you must manually reload the chassis or all line cards */
RP/0/RP0/CPU0:ios(config)#exit
Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:yes
RP/0/RP0/CPU0:router# reload location 0/0/CPU0
RP/0/RSP0/CPU0:ios(config)# policy-map CHILD-INGRESS
RP/0/RSP0/CPU0:ios(config-pmap)# class REAL-TIME
RP/0/RSP0/CPU0:ios(config-pmap-c)# set traffic-class 5
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#police rate 20 mbps
RP/0/RSP0/CPU0:ios(config-pmap)# class NET-CONTROL
RP/0/RSP0/CPU0:ios(config-pmap-c)# set traffic-class 4
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#police rate 5 mbps
RP/0/RSP0/CPU0:ios(config-pmap)# class DATA1
RP/0/RSP0/CPU0:ios(config-pmap-c)# set traffic-class 3
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#police rate 5 mbps
RP/0/RSP0/CPU0:ios(config-cmap)# set qos-group 3
RP/0/RSP0/CPU0:ios(config-cmap)# class DATA2
RP/0/RSP0/CPU0:ios(config-pmap-c)# set traffic-class 2
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#police rate 5 mbps
RP/0/RSP0/CPU0:ios(config-cmap)# set qos-group 3
RP/0/RSP0/CPU0:ios(config-pmap-c)#class class-default
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#police rate 10 mbps
RP/0/RSP0/CPU0:ios(config-pmap)#end-policy-map
RP/0/RSP0/CPU0:ios(config)# policy-map PARENT-INGRESS
RP/0/RSP0/CPU0:ios(config-pmap)#class class-default
RP/0/RSP0/CPU0:ios(config-cmap-c)# service-policy CHILD-INGRESS
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#police rate 100 mbps
RP/0/RSP0/CPU0:ios(config-pmap)#end-policy-map
RP/0/RSP0/CPU0:ios(config-pmap)#
RP/0/RSP0/CPU0:ios(config-pmap)# interface GigabitEthernet0/0/0/22.13
RP/0/RSP0/CPU0:ios(config-pmap)# service-policy in parent-policer
RP/0/RSP0/CPU0:ios (config-subif)#end
Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:yes
```

Running Configuration

```
RP/0/RP0/CPU0:ios#show run interface GigabitEthernet0/0/0/22.13
Tue Feb 23 20:45:08.889 GMT
interface GigabitEthernet0/0/0/22.13
 description Testing interface
 service-policy input parent-policer
 vrf customer1
 ipv4 address 172.16.1.1 255.255.255.252

policy-map PARENT-INGRESS
  class class-default
    service-policy CHILD-INGRESS
    police rate 100 mbps
  !
end-policy-map
!
policy-map CHILD-INGRESS
  class REAL-TIME
    set traffic-class 5
    police rate 20 mbps
  !
  class NET-CONTROL
    set traffic-class 4
    police rate 5 mbps
  !
  class DATA1
    set traffic-class 3
    police rate 5 mbps
    set qos-group 3
  !
  class DATA2
    set traffic-class 2
    police rate 5 mbps
    set qos-group 3
 !
  class class-default
    police rate 10 mbps
!
end-policy-map
```

Figure 6: Parent Conform Aware-Ingress Policing






```
RP/0/RSP0/CPU0:ios(config)#policy-map parent-policer
RP/0/RSP0/CPU0:ios(config-pmap)#class class-default
RP/0/RSP0/CPU0:ios(config-pmap-c)#service-policy child-policer
RP/0/RSP0/CPU0:ios(config-pmap-c)#police rate 30 mbps peak-rate 30 mbps
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#end-policy-map
RP/0/RSP0/CPU0:ios(config)# policy-map child-policer
RP/0/RSP0/CPU0:ios(config-pmap)# class cos3
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#police rate 10 mbps peak-rate 30 mbps
RP/0/RSP0/CPU0:ios(config-pmap)# class cos4
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#police rate 15 mbps peak-rate 30 mbps
RP/0/RSP0/CPU0:ios(config-pmap-c)#class class-default
RP/0/RSP0/CPU0:ios(config-pmap-c-police)#police rate 5 mbps peak-rate 30 mbps
RP/0/RSP0/CPU0:ios(config-pmap)#end-policy-map
```

Running Configuration

```
RP/0/RP0/CPU0:ios#show run policy-map parent-policer
Tue Feb 23 19:37:57.500 GMT
policy-map parent-policer
 class class-default
  service-policy child-policer
  police rate 30 mbps peak-rate 30 mbps
  !
 end-policy-map

RP/0/RP0/CPU0:ios#show run policy-map child-policer
Tue Feb 23 19:38:35.472 GMT
policy-map child-policer
 class cos3
  police rate 10 mbps peak-rate 30 mbps
  !
 class cos4
  police rate 15 mbps peak-rate 30 mbps
  !
 class class-default
  police rate 5 mbps peak-rate 30 mbps
  !
 end-policy-map
```

Verification

Run the **show policy-map interface** command in XR EXEC mode to confirm that the committed information rate (CIR) is prioritized over the peak information rate (PIR).

In the example below, **Policed (conform)** or CIR from each class is prioritized over **Policed (exceed)** or PIR.

```
RP/0/RP0/CPU0:ios# show policy-map interface GigabitEthernet0/0/0/22.13 input

GigabitEthernet0/0/0/22.13 input: parent-policer

Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :          217797200/111512166400         2344847
    Transmitted         :            8314388/4256966656           88089
    Total Dropped       :          209482812/107255199744         2256758

  Policing statistics                (packets/bytes)     (rate - kbps)
    Policed(conform)    :            6602174/3380313088           69926
    Policed(exceed)     :            1712214/876653568            18165
    Policed(violate)    :          209482812/107255199744         2256782
    Policed and dropped :          209482812/107255199744

  Policy child-policer Class cos3
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :          54449300/27878041600           586215
      Transmitted         :           3246813/1662368256            34399
      Total Dropped       :          51202487/26215673344           551816
    Policing statistics                (packets/bytes)     (rate - kbps)
      Policed(conform)    :           2818471/1443057152            29851
      Policed(exceed)     :            428342/219311104             4547
      Policed(violate)    :          51202487/26215673344           551816
      Policed and dropped :          51202487/26215673344
      Policed and dropped(parent policer)  : 0/0

  Policy child-policer Class cos4
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :          54449300/27878041600           586213
      Transmitted         :           2319731/1187702272            24577
     Total Dropped       :          52129569/26690339328           561636
    Policing statistics                (packets/bytes)     (rate - kbps)
      Policed(conform)    :           1891851/968627712             20037
      Policed(exceed)     :            427880/219074560             4540
      Policed(violate)    :          52129569/26690339328           561636
      Policed and dropped :          52129569/26690339328
      Policed and dropped(parent policer)  : 0/0

  Policy child-policer Class class-default
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :          108898600/55756083200          1172419
      Transmitted         :           2747844/1406896128            29113
      Total Dropped       :          106150756/54349187072          1143306
    Policing statistics                (packets/bytes)     (rate - kbps)
      Policed(conform)    :           1891852/968628224             20036
      Policed(exceed)     :            855992/438267904             9076
      Policed(violate)    :          106150756/54349187072          1143306
      Policed and dropped :          106150756/54349187072
      Policed and dropped(parent policer)  : 0/0
```

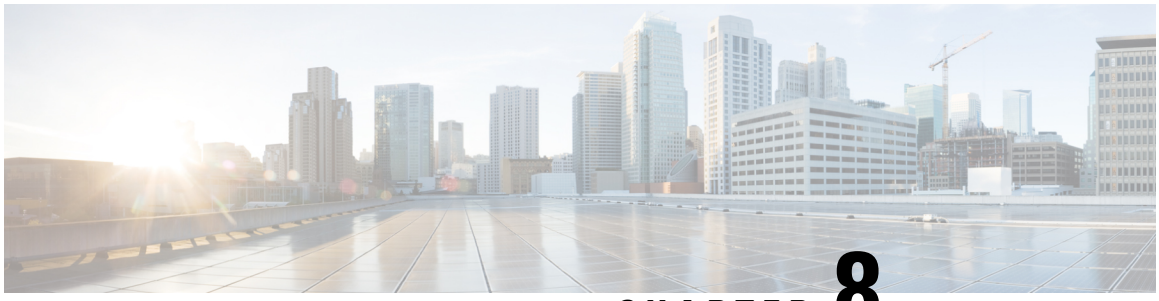
Related Commands hw-module profile qos conform-aware-policer

Conform Aware Hierarchical Policy Restrictions

The parent-policy traffic police rate must be greater than or equal to the sum of child conform rates.

- The Confirm-Aware hierarchical policer function allows the child level policy to color the traffic as either green, yellow, or red for handling at the parent-level policer.

- All classes in the child-level of the QoS policy need to have a policer configured for their packets to be handled correctly by the parent policer.



CHAPTER 8

QoS for Bridge-Group Virtual Interfaces

Integrated Routing and Bridging (IRB) provides the ability to route between a bridge group and a routed domain with the help of Bridge-Group Virtual Interface (BVI).

The BVI is a virtual interface within the router that acts like a normal routed interface that does not support bridging, but represents the comparable bridge group to routed interfaces within the router. The interface number of the BVI is the number of the bridge group that the virtual interface represents. The number is the link between the BVI and the bridge group.

For more information on IRB/ BVI, please refer the *Interface and Hardware Component Configuration Guide for Cisco NCS 540 Series Routers*

- Information on Qos on BVI, on page 169
- Restrictions on BVI , on page 169
- Classification and Marking , on page 170
- Configuring QoS on BVI , on page 171
- Verifying QoS on BVI, on page 172

Information on Qos on BVI

A BVI integrates Layer2 domain with Layer3 domain by creating a virtual interface in between them. The traffic flow supported for QoS is from the bridged to routed interface.

- A BVI can have bridge domain members from different linecards or NPU or Core.
- A BVI service-policy is applied on all linecards and within each linecard it is replicated per NPU. If any policers are configured in the qos policy map is applied to the BVI interface, then the policer is configured on each core of the NPU, and is shared among all the interfaces on that NPU core.

Restrictions on BVI

- Egress QoS on BVI is not supported for Layer2 flows for routed to bridge domain traffic.
- QoS on BVI is not supported for:
  - Queuing (priority, bandwidth, bandwidth remaining, sShaping, queue-limit, random-detect)
  - Shared policy instance

- Overhead accounting
- Percentage policer at the parent level

- Conditional EXP marking is only supported for Layer3 VPN when BVI is the AC interface or the packets have destination MAC address as BVI interface.

  Conditional EXP marking is not supported for EVPN (Layer2 traffic – non-BVI destination MAC traffic).

- Inheriting a policy map must be from either from a BVI interface or main interface.

  The policy map on a bridge domain sub-interface is inherited by the main interface and BVI interface. If the policy is applied on both the interfaces, then the policy is inherited on the last applied interface. This configuration is not supported and should be avoided to undesired or unknown behavior.

- A policy on a BVI interface should be applied after the BVI is added to the bridge domain as a routed interface. If the policy is applied before, then the policy has no effect.
- The show QoS interface command is not supported for the BVI interface.

Classification and Marking

The following features are supported:

- Classification
- Policing (level 1 and level 2)
- Ingress marking

Table 29: Classification and Marking

	Classification	Marking	Direction	Comment
Qos-group	No	Yes	Ingress	Used for Egress marking policy
Discard-class	No	Yes	Ingress	Used for VOQ selection and Egress queuing policy
Precedence	Yes	Yes	Ingress	
DSCP	Yes	Yes	Ingress	
Vlan	No	No		
CoS	Yes	No	Ingress	Only for Layer2 flows (bridge member).
Dei	Yes	No	Ingress	Only for Layer2 flows (bridge member).

	Classification	Marking	Direction	Comment
EXP	No	Yes	Ingress	

Configuring QoS on BVI

```
Router# show running-config interface bvI 1

interface BVI1
 service-policy input bvi-ingress
 ipv4 address 1.1.1.1 255.255.255.0
!

Router# show running-config policy-map bvi-ingress

policy-map bvi-ingress
 class prec1
  set traffic-class 1
  set mpls experimental imposition 3
  police rate 100 mbps
  !
 !
 class dscp_af22
  set traffic-class 2
  set mpls experimental imposition 4
  police rate 200 mbps
  !
 !
 class class-default
 !
 end-policy-map
!

Router# show qos interface bvI 1 input

NOTE:- Configured values are displayed within parentheses
Interface BVI1 ifh 0x800464c  -- input policy
NPU Id:                        0
Total number of classes:       3
Interface Bandwidth:           104857600 kbps
Policy Name:                   bvi-ingress
SPI Id:                        0x0
Accounting Type:               Layer1 (Include Layer 1 encapsulation and above)
------------------------------------------------------------------------------
Level1 Class                             =   prec1
New traffic class                        =   1
New imposition exp                       =   3

Policer Bucket ID                        =   0x6
Policer Stats Handle                     =   0x0
Policer committed rate                   =   99844 kbps (100 mbits/sec)
Policer conform burst                    =   124672 bytes (default)

Level1 Class                             =   dscp_af22
New traffic class                        =   2
New imposition exp                       =   4

Policer Bucket ID                        =   0x5
Policer Stats Handle                     =   0x0
```

```
Policer committed rate                    =   199688 kbps (200 mbits/sec)
Policer conform burst                     =   249472 bytes (default)

Level1 Class                              =   class-default

Default Policer Bucket ID                 =   0x4
Default Policer Stats Handle              =   0x0
Policer not configured for this class
```

Verifying QoS on BVI

Use the **show policy-map interface input** command to collect statistics from all linecards.

```
Router# show policy-map interface bvI 1 input

BVI1 input: bvi-ingress

Class prec1
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0                   0
    Transmitted         :                  0/0                   0
    Total Dropped       :                  0/0                   0
Class dscp_af22
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0                   0
    Transmitted         :                  0/0                   0
    Total Dropped       :                  0/0                   0
Class class-default
  Classification statistics          (packets/bytes)     (rate - kbps)
    Matched             :                  0/0                   0
    Transmitted         :                  0/0                   0
    Total Dropped       :                  0/0                   0
```