Modular QoS Configuration Guide for Cisco NCS 5500 Series Routers, IOS XR Release 6.3.x

First Published: 2017-09-15

Last Modified: 2018-03-15

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

CONTENTS

PREFACE

Preface vii

Changes to This Document vii

Communications, Services, and Additional Information vii

CHAPTER 1

New and Changed QoS Features 1

New and Changed QoS Features 1

CHAPTER 2

Configuring Modular QoS Service Packet Classification 3

Packet Classification Overview 3

Traffic Class Elements 4

Default Traffic Class 5

Create a Traffic Class 5

Traffic Policy Elements 7

Create a Traffic Policy 8

Attach a Traffic Policy to an Interface 9

Packet Marking 11

QoS Re-marking of IP Packets in Egress Direction 12

QoS Re-marking of Ethernet Packets in Egress Direction 13

QoS L2 Re-marking of Ethernet Packets in Egress Direction 13

Bundle Traffic Policies 14

Configuring QoS Groups with an ACL 14

QoS Egress Marking and Queuing Using Dual Policy-Map 17

Restrictions 19

In-Place Policy Modification 21

References for Modular QoS Service Packet Classification 22

Specification of the CoS for a Packet with IP Precedence 22

IP Precedence Bits Used to Classify Packets 23
IP Precedence Value Settings 23
IP Precedence Compared to IP DSCP Marking 23
Usage of QoS-group and Queue Selection 23
Conditional Marking of MPLS Experimental bits for L3VPN Traffic 24
Policy-map for conditional marking of incoming IPv4 and IPv6 traffic 25
Policy-map for conditional marking of outgoing MPLS traffic 25

CHAPTER 3 **Configuring Modular QoS Congestion Avoidance** 27
Modular QoS Congestion Avoidance 27
Tail Drop and the FIFO Queue 27
Configure Tail Drop 27
Random Early Detection and TCP 29
Configure Random Early Detection 29
Weighted Random Early Detection 31
Average Queue Size for WRED 32
Configure Weighted Random Early Detection 32

CHAPTER 4 **Configuring Modular QoS Congestion Management** 35
Congestion Management Overview 35
Class-based Weighted Fair Queueing 35
Bandwidth Remaining 36
Configure Minimum Bandwidth and Bandwidth Remaining 36
Low-Latency Queueing with Strict Priority Queueing 38
Configure Low Latency Queueing with Strict Priority Queueing 38
Traffic Shaping 40
Configure Traffic Shaping 40
Traffic Policing 42
Committed Bursts and Excess Bursts 43
Single-Rate Policer 43
Configure Traffic Policing (Single-Rate Two-Color) 44
Configure Traffic Policing (Single-Rate Three-Color) 46
Two-Rate Policer 47
Configure Traffic Policing (Two-Rate Three-Color) 48

Per-thousand and Per-million Units 50
References for Modular QoS Congestion Management 50
Committed Bursts 50
Excess Bursts 51
Two-Rate Policer Details 51

CHAPTER 5 **Configuring Modular QoS on Link Bundles** 53
QoS on Link Bundles 53
Load Balancing 53
Configure QoS on Link Bundles 54

CHAPTER 6 **Configuring Hierarchical Modular QoS** 59
Overview of Hierarchical Modular QoS 59
Restrictions for Configuring H-QoS 59
Configuring Hierarchical Queuing 60

Preface

This product has reached end-of-life status. For more information, see the End-of-Life and End-of-Sale Notices.

This preface contains these sections:

- Changes to This Document, on page vii
- Communications, Services, and Additional Information, on page vii

Changes to This Document

Table 1: Changes to This Document

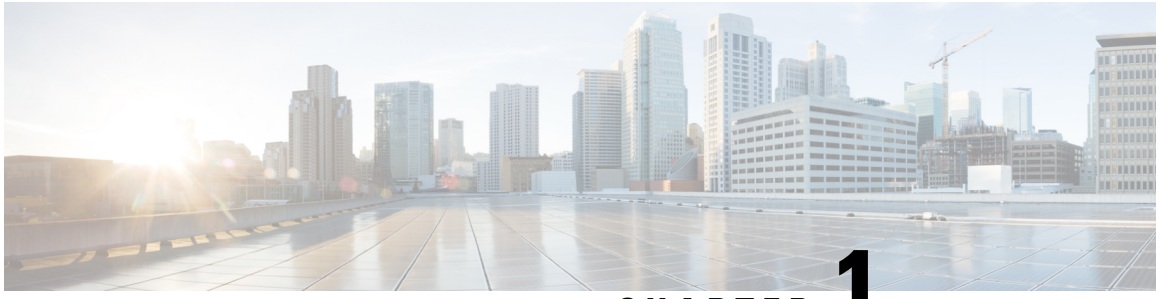
Date	Summary
September 2017	Initial release of this document.
March 2018	Republished for Release 6.3.2.

Communications, Services, and Additional Information

- To receive timely, relevant information from Cisco, sign up at Cisco Profile Manager.
- To get the business impact you're looking for with the technologies that matter, visit Cisco Services.
- To submit a service request, visit Cisco Support.
- To discover and browse secure, validated enterprise-class apps, products, solutions and services, visit Cisco Marketplace.
- To obtain general networking, training, and certification titles, visit Cisco Press.
- To find warranty information for a specific product or product family, access Cisco Warranty Finder.

Cisco Bug Search Tool

Cisco Bug Search Tool (BST) is a web-based tool that acts as a gateway to the Cisco bug tracking system that maintains a comprehensive list of defects and vulnerabilities in Cisco products and software. BST provides you with detailed defect information about your products and software.



CHAPTER 1

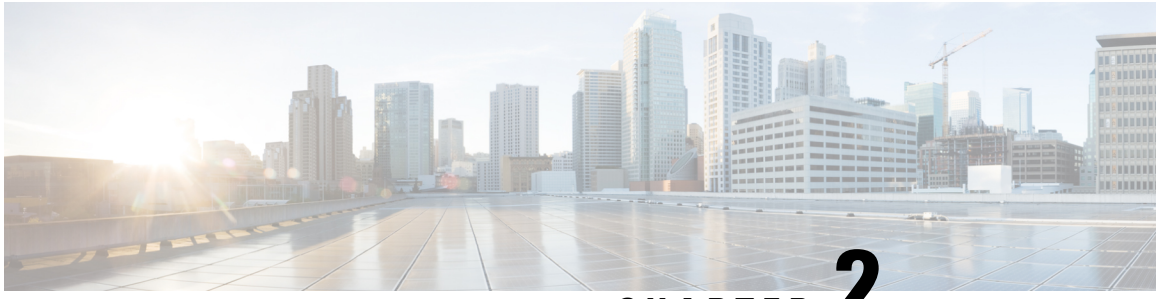
New and Changed QoS Features

- New and Changed QoS Features, on page 1

New and Changed QoS Features

Table 2: QoS Features Added or Modified in IOS XR Release 6.3.x

Feature	Description	Changed in Release	Where Documented
Hierarchical Modular QoS (H-QoS)	H-QoS is a QoS model that enables you to specify QoS behavior at multiple levels of hierarchy.	Release 6.3.1	Configuring Hierarchical Modular QoS, on page 59 chapter
Conditional Marking of MPLS Experimental bits for L3VPN traffic	This features allows the L3VPN traffic packets to be marked for differential treatment.	Release 6.3.2	Configuring Modular QoS Service Packet Classification, on page 3 chapter



CHAPTER 2

Configuring Modular QoS Service Packet Classification

Table 3: Feature History

Release	Modification
Release 6.2.2	Added support for QoS re-marking of IP packets in egress direction.
Release 6.3.2	Added support for conditional marking of MPLS Experimental bits for L3VPN traffic.

This chapter covers these topics:

- Packet Classification Overview, on page 3
- Traffic Class Elements, on page 4
- Traffic Policy Elements, on page 7
- Configuring QoS Groups with an ACL, on page 14
- QoS Egress Marking and Queuing Using Dual Policy-Map, on page 17
- Restrictions, on page 19
- In-Place Policy Modification, on page 21
- References for Modular QoS Service Packet Classification, on page 22

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) is used to define 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.

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 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 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
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)	7	No	No	Egress
CoS	(0,7)	8	No	Yes	Ingress
DEI	(0,1)	1	No	No	Ingress
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 qos-group** *x* configured in the ingress, you must have a corresponding **match qos-group** *x* in the egress, in order to see the statistics in the egress side. Also, see Usage of QoS-group and Queue Selection, on page 23.

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 **qos-group** (1-7) is supported. Match **qos-group 0** cannot be configured. The class-default in the egress policy maps to **qos-group 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

- Users 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.
- Use the **not** keyword with the **match** command to perform a match based on the values of a field that are not specified.
- All **match** commands specified in this configuration task are considered optional, but you must configure at least one match criterion for a class.
- 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.
- For the **match access-group** command, QoS classification based on the packet length or TTL (time to live) field in the IPv4 and IPv6 headers is not supported.

- 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.
- The **match qos-group**, **traffic-class**, and **discard-class** are supported only in egress direction, and these are the only match criteria supported in egress direction.
- The egress default class implicitly matches **qos-group** 0.
- 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.
- 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 does not take effect for multicast traffic in the high priority (HP) traffic classes. It only applies to unicast traffic.
- 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 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.
- Also, see Usage of QoS-group and Queue Selection, on page 23.

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 4.)

```
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 10.

Also see, Verification, on page 10.

Related Topics

- Traffic Class Elements, on page 4

• Traffic Policy Elements, on page 7

Associated Commands

- class-map
- match access-group
- match dscp
- match mpls experimental topmost
- match precedence
- match qos-group

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 11)
police	ingress
priority	egress (level 1 to level 7)
queue-limit	egress
shape	egress

Supported Action Types	Direction supported on Interfaces
wred	egress

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.

To configure a traffic class, see Create a Traffic Class, on page 5.

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 9.

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 7)

```
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 10.

See, Verification, on page 10.

Related Topics

- Traffic Policy Elements, on page 7
- Traffic Class Elements, on page 4

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

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 Create a Traffic Class, on page 5)
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 8)
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 test-shape-1
Router(config-int)# commit
```

Running Configuration

```
/* Class-map configuration */

class-map match-any traffic-class-1
 match traffic-class 1
 end-class-map
!
- - -
- - -

/* Traffic policy configuration */
policy-map test-shape-1
 class traffic-class-1
  shape average percent 40
 !
 class class-default
 !
 end-policy-map
!
- - -
- - -

/* Attaching traffic policy to an interface in egress direction */
interface HundredGigE0/6/0/18
 service-policy output test-shape-1
 !
```

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 7
- Traffic Class Elements, on page 4

Associated Commands

- service-policy

Packet Marking

The packet marking feature provides users with a means to differentiate packets based on the designated markings. The router supports egress packet marking. match on **discard-class** on egress, if configured, can be used for a marking policy only.

The router also supports L2 ingress marking.

For ingress marking:

Ingress traffic— For the ingress pop operation, re-marking the customer VLAN tag (CoS, DEI) is not supported.

Egress traffic— 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:

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

Limitation

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

Supported Packet Marking Operations

This table shows the supported packet marking operations.

Supported Mark Types	Range	Support for Unconditional Marking	Support for Conditional Marking
set cos	0-7	ingress	No
set dei	0-1	ingress	No
set discard-class	0-3	ingress	No
set dscp	0-63	ingress	No
set mpls experimental topmost	0-7	ingress	No

Supported Mark Types	Range	Support for Unconditional Marking	Support for Conditional Marking
set precedence	0-7	ingress	No
set QoS-group	0-7	ingress	No

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.

- 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 QoS group identifier on MPLS packets, use the **set qos-group** command in policy map class configuration mode.

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

- Use QoS unconditional packet marking to assign packets to set the priority value of IEEE 802.1p/ Inter-Switch Link (ISL) packets. The router uses the CoS value to determine how to prioritize packets for transmission and can use this marking to perform Layer 2-to-Layer 3 mapping. To set the Layer 2 CoS value of an outgoing packet, use the **set cos** command in policy map configuration mode.

- Use QoS unconditional packet marking to mark a packet based on the drop eligible indicator value (DEI) bit on 802.1ad frames. To set the DEI value, use the **set dei** command to set the drop eligible indicator value (DEI) in policy map class configuration mode.

Note

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

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
!
```

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 are supported.

Note Egress marking is not supported on BVI interfaces.

For details, see Configure QoS on Link Bundles, on page 54.

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).

Note You cannot configure QoS group ACLs on NC57 line cards because you cannot configure the command, **hw-module profile qos ingress-model peering**, on these line cards.

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.

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(config-if)# ipv4 access-group qos-acl ingress compress level 3

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.

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

Refer to the below table for Ingress QoS Scale limitation.

Table 4: Ingress QoS Scale Limitation

QoS Mode	Class-Map Size	Maximum number of Interfaces with Ingress QoS Applied	
		Per Core	Per NPU
Normal	4	1023	2046
Normal	8	511	1022
Normal	16	255	510
Normal	32	127	254
Enhanced	4	871	1742
Enhanced	8	435	870
Enhanced	16	217	434
Enhanced	32	108	216

Example: For Default Configuration, which is Normal (2 counter mode) QoS Mode & 32 Class Map-Size, you can configure 127 interfaces with Ingress Policy per core.

Other restrictions to follow:

- If you have a **set traffic class** statement explicitly configured in ingress service policy, it is 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 egress class-default, traffic class should be set to 0 on ingress.
- If you have a **set traffic class** configured in Ingress service policy, and no corresponding **match traffic class** on egress, the traffic will not go to class default and the stats for this traffic flow will not be seen in **show policy-map interface <> output** command.
- If you do not 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 ingress service policy, it is 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 ingress service policy and do not have a corresponding **match discard-class** on egress, the traffic will not hit the class-default and the stats for this flow will not be accounted in **show policy-map interface <> output** command.
- The system does not 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

- **explicit set discard-class** statement is not supported.
- This feature is supported only on L3 interfaces and is limited to 1000 L3 interfaces per system.
- **set mpls exp topmost** statement is not supported within QoS in peering mode.
- **access group** statement is not supported.
- (Only in Release 6.2.x and Release 6.3.x) **set mpls exp imposition** statement is not supported on ingress interface.

Restrictions for QoS on BVI

- The system does not support egress QoS policy on BVI.
- 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 cannot be applied on the L2 interfaces. Similarly, if a QoS policy is attached to any of the L2 interfaces, any QoS policy cannot be applied on the BVI, which is part of the same bridge-domain.

Restrictions for TCAM

- The creation of 250 ingress unique policy-maps is supported. However, you may be able to create up to 254 unique policy maps after which the error message “Out of ACLID resource” may display. However, you must avoid creating more than 250 ingress unique policy maps because the additional map sizes are reserved for internal purposes.
- The 250 policy-maps scale is based on the internal TCAM space available for each PID. The available TCAM space differs for every PID, and is dependent upon TCAM bank sharing.

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 L3 interface, there is a limit of 3 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 L2 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

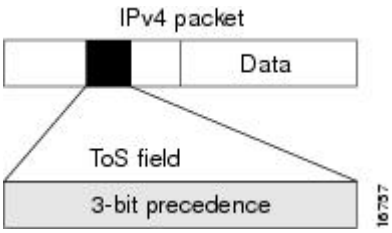
If unrecoverable errors occur during in-place policy modification, the policy is put into an inconsistent state on target interfaces. No new configuration is possible until the configuration session is unblocked. It is recommended to remove the policy from the interface, check the modified policy and 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 1: 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.

Usage of QoS-group and Queue Selection

The router supports up to 8 CoSQs for each egress interface, in the range of 0 through 7, with 0 being the default CoSQ. The **qos-group** value is used to select a CoSQ and eventually a virtual output queue (VOQ).

In order to designate the traffic class to a certain CoSQ other than CoSQ 0, in the ingress policy-map, you must explicitly configure **set qos-group** *x* command in the class-map, where 'x' is the CoSQ value.

In the egress policy-map, a class-map with a corresponding **match qos-group** *x* allows further QoS actions to be applied to the traffic class.

For example,

```
class-map prec1
  match prec 1

policy-map test-ingress
  class prec1
    set qos-group 1
    police rate percent 50

class-map qg1
  match qos-group 1

policy-map test-egress
  class qg1
    set cos 1
```

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 2 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 3 policy-maps which perform conditional marking of MPLS experimental bits can be configured per Network Processor Unit (NPU) of the Cisco NCS 5500 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.

Note The DSCP value of the packet is preserved only for L3VPN networks when the packets are destined to the directly connected routes on the PE routers. To preserve the DSCP value for packets to destinations beyond the egress PE routers for L3VPN, you should use the **label mode per-vrf** command under the VRF at the PE routers. Similarly for MPLS networks, the default behaviour is uniform mode where the penultimate hop copies the MPLS EXP bit to IP DSCP and IP DSCP value is not preserved beyond the egress PE router. To preserve IP DSCP value, you should use the **mpls ip-ttl-propogate disable** command at the penultimate hop.

Note DSCP value preservation is not supported on NC55-24DD and NC55-18DD-SE line cards for Release 7.0.2.

Policy-map for conditional marking of incoming IPv4 and IPv6 traffic

The incoming packets are classified based on the ingress policy-map and the following actions are done.

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

Running Configuration:

```
policy-map ingress
 class af11
  police rate percent 10 peak-rate percent 20
  !
  set qos-group 1
 !
 class af22
  police rate percent 30 peak-rate percent 50
  !
  set qos-group 2
 !
 class af32
  set qos-group 3
  police rate percent 30 peak-rate percent 60
  !
 !
 class class-default
 !
 end-policy-map
!
```

Policy-map for conditional marking of outgoing MPLS traffic

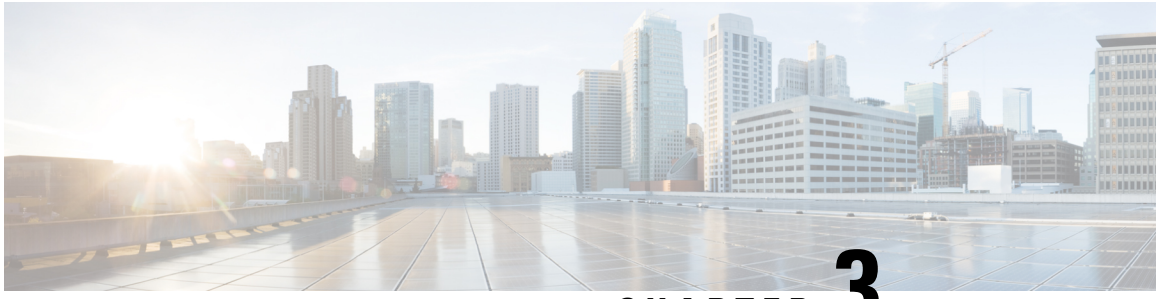
The IPv4 or IPv6 ingress packet undergoes MPLS encapsulation during the egress processing in the PE router which performs the label imposition. The MPLS experimental bits are marked on the basis of egress policy-map which performs the following actions:

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

Running Configuration:

```
policy-map egress
 class qos1_disc0 # This class matches on qos-group 1 and discard-class 0
  set mpls experimental imposition 1
 !
 class qos1_disc1 # This class matches on qos-group 1 and discard-class 1
  set mpls experimental imposition 5
 !
 class qos2_disc0 # This class matches on qos-group 2 and discard-class 0
  set mpls experimental imposition 2
```

```
 !
 class qos2_disc1 # This class matches on qos-group 2 and discard-class 1
  set mpls experimental imposition 6
 !
 class qos3_disc0 # This class matches on qos-group 3 and discard-class 0
  set mpls experimental imposition 3
 !
 class qos3_disc1 # This class matches on qos-group 3 and discard-class 1
  set mpls experimental imposition 7
 !
 class class-default
 !
 end-policy-map
!
```



CHAPTER 3

Configuring Modular QoS Congestion Avoidance

This chapter covers the following topics:

- Modular QoS Congestion Avoidance , on page 27
- Tail Drop and the FIFO Queue, on page 27
- Random Early Detection and TCP, on page 29
- Weighted Random Early Detection, on page 31

Modular QoS Congestion Avoidance

Congestion avoidance techniques monitor traffic flow in an effort 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.

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

- Tail Drop and the FIFO Queue, on page 27
- Random Early Detection and TCP, on page 29
- Weighted Random Early Detection, on page 31

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 27

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).

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 29

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.

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**.

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

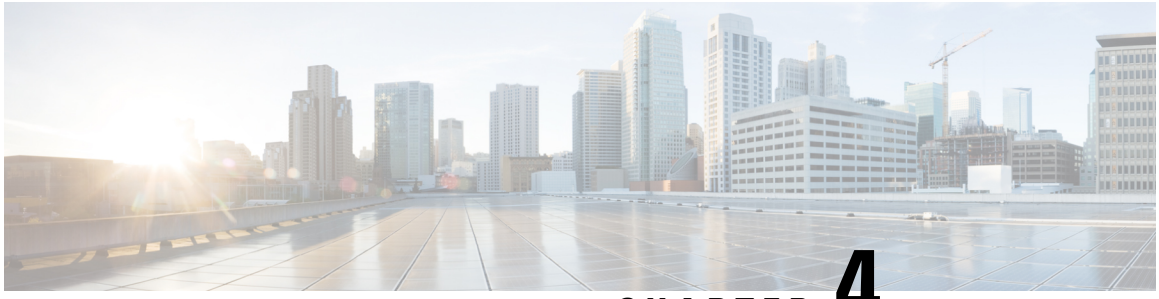
```
Router# show qos int hundredGigE 0/0/0/20 output

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 31
- Configure Random Early Detection, on page 29

Associated Commands

- random-detect



CHAPTER 4

Configuring Modular QoS Congestion Management

This chapter covers the following topics:

- Congestion Management Overview, on page 35
- Class-based Weighted Fair Queueing, on page 35
- Low-Latency Queueing with Strict Priority Queueing, on page 38
- Traffic Shaping, on page 40
- Traffic Policing, on page 42
- References for Modular QoS Congestion Management, on page 50

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 35
- Low-Latency Queueing with Strict Priority Queueing, on page 38
- Traffic Shaping, on page 40
- Traffic Policing, on page 42

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.

Configure Minimum Bandwidth and Bandwidth Remaining

Guidelines

- The **bandwidth**, **bandwidth remaining**, **shaping**, **queue-limit**, and **random-detect** commands may be configured together in the same class. The **priority** command cannot be configured along with **bandwidth**, **bandwidth remaining** commands, but can be configured with **shaping**, **queue-limit** and **random-detect** commands in the same class.

Configuration Example

You have to accomplish the following to complete the minimum bandwidth and 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 minimum bandwidth and leftover bandwidth for the class
4. Attaching the policy-map to an output interface

```
Router# configure
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-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 input 100g-s1-1
 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 36

Associated Commands

- bandwidth
- bandwidth remaining

Low-Latency Queueing with Strict Priority Queueing

The Low-Latency Queueing (LLQ) feature brings strict priority queuing (PQ) to the CBWFQ scheduling mechanism. Priority Queueing (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.

Configure Low Latency Queueing with Strict Priority Queueing

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

Guidelines

- Only priority level 1 to 7 is supported, with 1 being the highest priority and 7 being the lowest. However, the default CoSQ 0 has the lowest priority among all.
- Priority level 1 to 7 is supported for non-H-QoS profiles, with 1 being the highest priority and 7 being the lowest. For H-QoS profiles, priority level 1 to 4 is supported. For all profiles, however, the class default is CoSQ 0 and has the lowest priority among all.
- Egress policing is not supported. Hence, in the case of strict priority queuing, there are chances that the other queues do not get serviced.
- You can configure **shape average** and **queue-limit** commands along with **priority**.

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
```

Related Topics

- Congestion Management Overview, on page 35
- Configure Traffic Shaping, on page 40
- Configure Minimum Bandwidth and Bandwidth Remaining, on page 36

Associated Commands

- priority

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.

Note Traffic shaping is supported only in egress direction.

Configure 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

- Only egress traffic shaping is supported.
- It is mandatory to configure all the eight qos-group classes (including class-default) for the egress policies.
- You can configure **shape average** command along with **priority** command.

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
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 percent 40
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
!
```

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
```

The shaper rate and peak burst size can be adjusted based on the hardware's queue credit parameter. The shaping rate accuracy may vary up to 5% in the worst case.

Related Topics

- Congestion Management Overview, on page 35

Associated Commands

- shape average

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 43.

The router supports the following traffic policing mode(s):

- Single-Rate Two-Color (SR2C) in color-blind mode. See Single-Rate Policer, on page 43.
- Single-Rate Three-Color (SR3C) in color-blind mode.
- Two-Rate Three-Color (2R3C) in color-blind mode. See Two-Rate Policer, on page 47 .

Restrictions

- Traffic policing is supported only in ingress direction, and only color-blind mode is supported.

- 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.

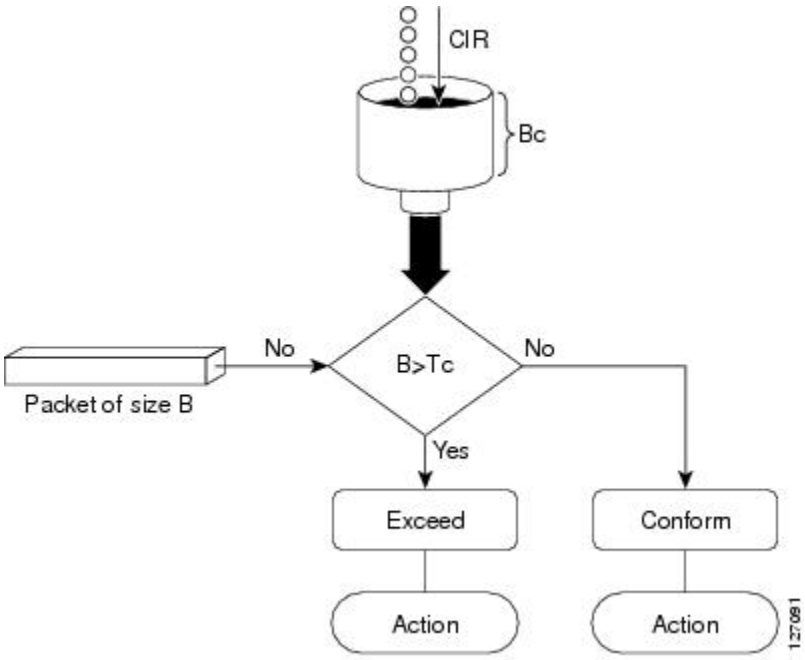
For more details, see Committed Bursts, on page 50 and Excess Bursts, on page 51.

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 2: 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 42

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 42

Associated Commands

- police rate

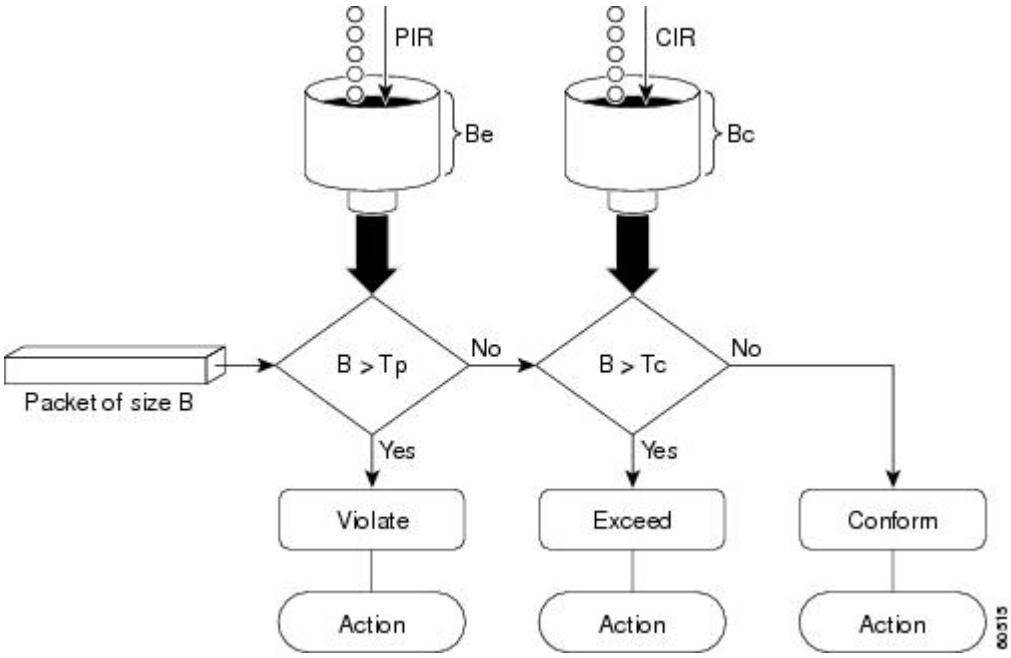
Two-Rate Policer

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 3: Marking Packets and Assigning Actions—Two-Rate Policer



Also, see Two-Rate Policer Details, on page 51.

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)
```

The minimum policer rate is 22 kbps for commit rate and a difference of 22 kbps is expected between commit and excess rates. Any value lower than 22 kbps is rounded up to 22 kbps in the hardware.

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 47

Associated Commands

- police rate

Per-thousand and Per-million Units

Shaper and policer rates can be configured in units of per-thousand and per-million on bundle interfaces. This provides the ability to provision shape and police rates down to 100 kbps on bundle or link aggregation (LAG) interfaces even with 100 GE bundle members.

For example, consider a 100GE interface and simple policy.

```
Interface HundredGig0/0/0/0
Service-policy output TEST
Policy-map TEST
Class C
 Shape average per-thousand 5
End-policy
```

Per thousand represents 0.1% of the link bandwidth and per million represents 0.0001% of the link bandwidth.

Which means that for a 100G link, 5 parts per thousand is 0.5% of the link bandwidth. Hence, the shape average per thousand of 5 in the above example enforces a shaper of 500 Mbps.

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.

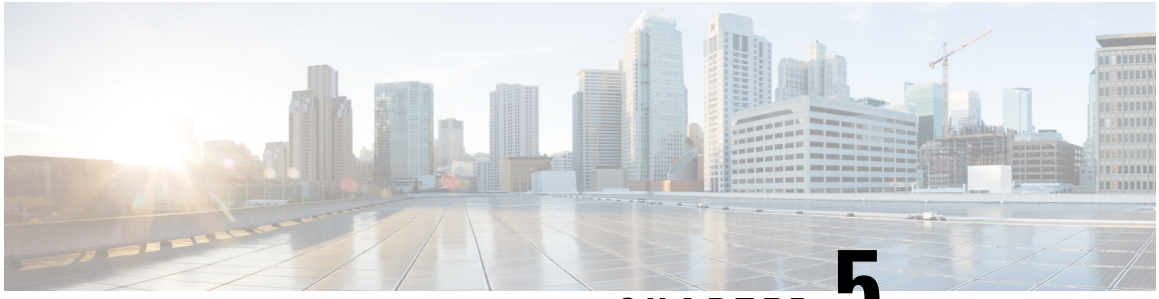
For example, if a data stream with a rate of 250 kbps arrives at the two-rate policer, and the CIR is 100 kbps and the PIR is 200 kbps, the policer marks the packet in the following way:

- 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 47.



CHAPTER 5

Configuring Modular QoS on Link Bundles

This chapter covers the following topics:

- QoS on Link Bundles, on page 53

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.

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 (ingress or egress direction), the policy is applied at each member interface. The reference bandwidth that is used to calculate shaper or bandwidth values is applied as per the physical member interface bandwidth.
- 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 9 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
```

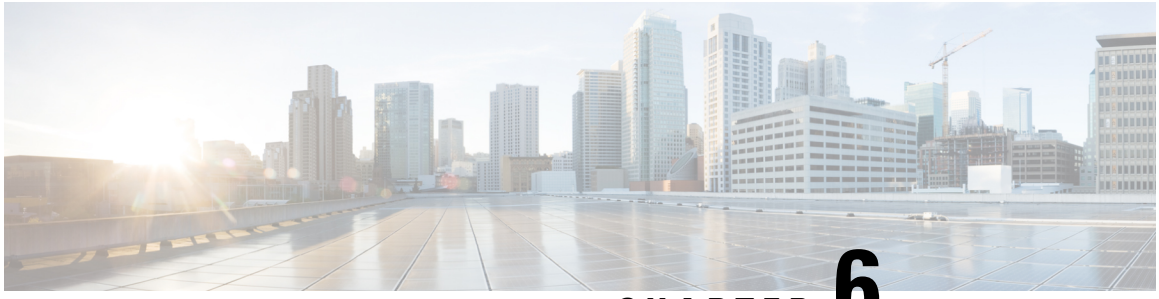
```
    Classification statistics          (packets/bytes)     (rate - kbps)
      Matched             :             244106/79922040             74
      Transmitted         :             244106/79922040             74
      Total Dropped       :                  0/0                    0
    Queueing 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 53

Associated Commands

- bundle maximu-active links
- interface Bundle-Ether



CHAPTER 6

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.

Table 5: Feature History for Hierarchical Modular QoS

Release	Modification
Release 6.3.1	Initial Release

This chapter covers the following topics:

- Overview of Hierarchical Modular QoS, on page 59
- Restrictions for Configuring H-QoS, on page 59
- Configuring Hierarchical Queuing, on page 60

Overview of Hierarchical Modular QoS

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 additional 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.

Restrictions for Configuring H-QoS

The following restrictions are applicable while configuring H-QoS:

1. The parent traffic policy only supports the traffic class of type class-default.

2. The parent traffic policy only supports the class-action **shape** and no other queuing action can be configured in it.
3. While configuring on the router, it is mandatory that the priority class must have traffic shaper in the child traffic policy.
4. The sum of the bandwidth of the child policies must be less than the parent policy's traffic shaper.
5. For congestion avoidance and management, the traffic shaper in the parent traffic policy calculates the queue limit and drop priority.
6. H-QoS profile and ingress peering profile do not work simultaneously. Hence, features requiring peering profile like lawful intercept also do not work with the HQoS profile enabled.
7. PBTS feature does not work when the H-QoS profile is enabled. This is due to TCAM limitations.
8. A maximum of 896 bundle sub-interfaces are only supported in the system, even if there are no QoS policies applied. This is due to an internal LAG_ID resource consumption in HQoS profile mode for bundle sub-interfaces with or without QoS policies being applied.
9. A maximum of 4 priority levels are only supported in HQoS 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.
10. 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.
11. Priority classes must have rate limit configuration by using a Shaping configuration. The effective shaper value is taken as priority bandwidth reservation. Sum of priority bandwidth reservations across all sub-interfaces and main interfaces must not exceed the network interface (NIF) port speed. This is to avoid over-subscription of priority traffic across the network interface port.

    Rates of non-priority classes and parent shaping can be over-subscribed.
12. The granularity of bandwidth or bandwidth remaining ration (BRR) is 1:64 as compared to 1:4096 in non-hqos mode. So, there could be accuracy differences in bandwidth performance based on the values used.
13. Filtering for egress IPv4 and IPv6 multicast traffic is not supported if H-QoS is configured on the router.

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
```