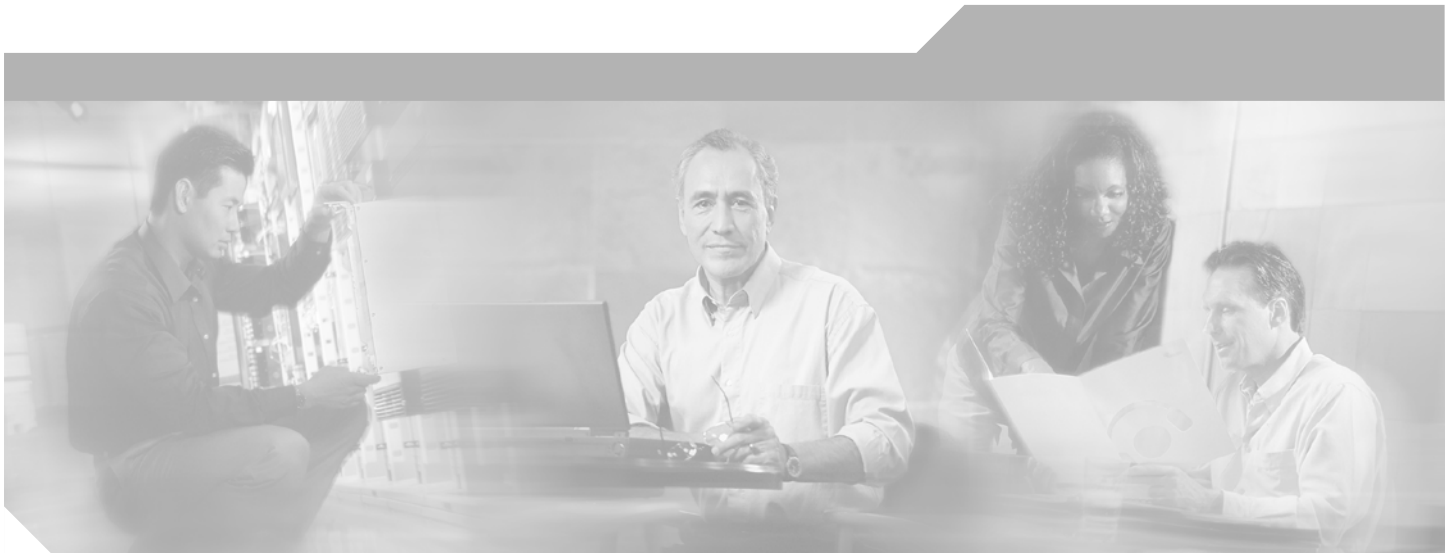


Cisco IOS XR Modular Quality of Service Configuration Guide

Cisco IOS XR Software Release 3.3

Corporate 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: 408 526-4100

Text Part Number: OL-8521-03

Preface

This document describes the Cisco IOS XR Quality of Service (QoS) feature for prioritizing traffic flow and providing preferential forwarding for higher-priority packets. The QoS techniques described include allocating bandwidth, improving loss characteristics, avoiding and managing network congestion, metering network traffic, and setting traffic-flow priorities across the network.

For QoS command descriptions, usage guidelines, and examples, refer to the Cisco IOS XR Modular Quality of Service Command Reference.

The preface contains the following sections:

Changes to This Document

Table 1 lists the technical changes made to this document since it was first printed.

Table 1 Changes to This Document

Revision	Date	Change Summary
OL-8521-03	July 2006	For Release 3.3.1, made the following changes: • Changed minimum and incremental steps for to be 256 kbps (was 125 kbps). • Functionaility: In the past, ACL matches based on the source port were ignored when the ACL was used to specify a QoS class. Source port matching specified through an ACL is now supported in the QoS context.
OL-8521-02	May 2006	The Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software module was modified as follows: • Added the Fabric Quality of Service policies and classes configuration information. See "Fabric Quality of Service Policies and Classes" section on page QC-11.
OL-8521-01	April 2006	Initial release of this document.

Obtaining Documentation

Cisco documentation and additional literature are available on Cisco.com. Cisco also provides several ways to obtain technical assistance and other technical resources. These sections explain how to obtain technical information from Cisco Systems.

Cisco.com

You can access the most current Cisco documentation at this URL:

http://www.cisco.com/techsupport

You can access the Cisco website at this URL:

http://www.cisco.com

You can access international Cisco websites at this URL:

http://www.cisco.com/public/countries_languages.shtml

Product Documentation DVD

The Product Documentation DVD is a comprehensive library of technical product documentation on a portable medium. The DVD enables you to access multiple versions of installation, configuration, and command guides for Cisco hardware and software products. With the DVD, you have access to the same HTML documentation that is found on the Cisco website without being connected to the Internet. Certain products also have .PDF versions of the documentation available.

The Product Documentation DVD is available as a single unit or as a subscription. Registered Cisco.com users (Cisco direct customers) can order a Product Documentation DVD (product number DOC-DOCDVD= or DOC-DOCDVD=SUB) from Cisco Marketplace at this URL:

http://www.cisco.com/go/marketplace/

Ordering Documentation

Registered Cisco.com users may order Cisco documentation at the Product Documentation Store in the Cisco Marketplace at this URL:

http://www.cisco.com/go/marketplace/

Nonregistered Cisco.com users can order technical documentation from 8:00 a.m. to 5:00 p.m. (0800 to 1700) PDT by calling 1 866 463-3487 in the United States and Canada, or elsewhere by calling 011 408 519-5055. You can also order documentation by e-mail at tech-doc-store-mkpl@external.cisco.com or by fax at 1 408 519-5001 in the United States and Canada, or elsewhere at 011 408 519-5001.

Documentation Feedback

You can rate and provide feedback about Cisco technical documents by completing the online feedback form that appears with the technical documents on Cisco.com.

You can submit comments about Cisco documentation by using the response card (if present) behind the front cover of your document or by writing to the following address:

Cisco Systems
Attn: Customer Document Ordering
170 West Tasman Drive
San Jose, CA 95134-9883

We appreciate your comments.

Cisco Product Security Overview

Cisco provides a free online Security Vulnerability Policy portal at this URL:

http://www.cisco.com/en/US/products/products_security_vulnerability_policy.html

From this site, you will find information about how to:

- Report security vulnerabilities in Cisco products.
- Obtain assistance with security incidents that involve Cisco products.
- Register to receive security information from Cisco.

A current list of security advisories, security notices, and security responses for Cisco products is available at this URL:

http://www.cisco.com/go/psirt

To see security advisories, security notices, and security responses as they are updated in real time, you can subscribe to the Product Security Incident Response Team Really Simple Syndication (PSIRT RSS) feed. Information about how to subscribe to the PSIRT RSS feed is found at this URL:

http://www.cisco.com/en/US/products/products_psirt_rss_feed.html

Reporting Security Problems in Cisco Products

Cisco is committed to delivering secure products. We test our products internally before we release them, and we strive to correct all vulnerabilities quickly. If you think that you have identified a vulnerability in a Cisco product, contact PSIRT:

- For Emergencies only—security-alert@cisco.com

  An emergency is either a condition in which a system is under active attack or a condition for which a severe and urgent security vulnerability should be reported. All other conditions are considered nonemergencies.

- For Nonemergencies—psirt@cisco.com

In an emergency, you can also reach PSIRT by telephone:

- 1 877 228-7302
- 1 408 525-6532

Tip We encourage you to use Pretty Good Privacy (PGP) or a compatible product (for example, GnuPG) to encrypt any sensitive information that you send to Cisco. PSIRT can work with information that has been encrypted with PGP versions 2.*x* through 9.*x*.

Never use a revoked or an expired encryption key. The correct public key to use in your correspondence with PSIRT is the one linked in the Contact Summary section of the Security Vulnerability Policy page at this URL:

http://www.cisco.com/en/US/products/products_security_vulnerability_policy.html

The link on this page has the current PGP key ID in use.

If you do not have or use PGP, contact PSIRT at the aforementioned e-mail addresses or phone numbers before sending any sensitive material to find other means of encrypting the data.

Obtaining Technical Assistance

Cisco Technical Support provides 24-hour-a-day award-winning technical assistance. The Cisco Technical Support & Documentation website on Cisco.com features extensive online support resources. In addition, if you have a valid Cisco service contract, Cisco Technical Assistance Center (TAC) engineers provide telephone support. If you do not have a valid Cisco service contract, contact your reseller.

Cisco Technical Support & Documentation Website

The Cisco Technical Support & Documentation website provides online documents and tools for troubleshooting and resolving technical issues with Cisco products and technologies. The website is available 24 hours a day, at this URL:

http://www.cisco.com/techsupport

Access to all tools on the Cisco Technical Support & Documentation website requires a Cisco.com user ID and password. If you have a valid service contract but do not have a user ID or password, you can register at this URL:

http://tools.cisco.com/RPF/register/register.do

Note Use the Cisco Product Identification (CPI) tool to locate your product serial number before submitting a web or phone request for service. You can access the CPI tool from the Cisco Technical Support & Documentation website by clicking the **Tools & Resources** link under Documentation & Tools. Choose **Cisco Product Identification Tool** from the Alphabetical Index drop-down list, or click the **Cisco Product Identification Tool** link under Alerts & RMAs. The CPI tool offers three search options: by product ID or model name; by tree view; or for certain products, by copying and pasting **show** command output. Search results show an illustration of your product with the serial number label location highlighted. Locate the serial number label on your product and record the information before placing a service call.

Submitting a Service Request

Using the online TAC Service Request Tool is the fastest way to open S3 and S4 service requests. (S3 and S4 service requests are those in which your network is minimally impaired or for which you require product information.) After you describe your situation, the TAC Service Request Tool provides recommended solutions. If your issue is not resolved using the recommended resources, your service request is assigned to a Cisco engineer. The TAC Service Request Tool is located at this URL:

http://www.cisco.com/techsupport/servicerequest

For S1 or S2 service requests, or if you do not have Internet access, contact the Cisco TAC by telephone. (S1 or S2 service requests are those in which your production network is down or severely degraded.) Cisco engineers are assigned immediately to S1 and S2 service requests to help keep your business operations running smoothly.

To open a service request by telephone, use one of the following numbers:

Asia-Pacific: +61 2 8446 7411 (Australia: 1 800 805 227)
EMEA: +32 2 704 55 55
USA: 1 800 553-2447

For a complete list of Cisco TAC contacts, go to this URL:

http://www.cisco.com/techsupport/contacts

Definitions of Service Request Severity

To ensure that all service requests are reported in a standard format, Cisco has established severity definitions.

Severity 1 (S1)—An existing network is down, or there is a critical impact to your business operations. You and Cisco will commit all necessary resources around the clock to resolve the situation.

Severity 2 (S2)—Operation of an existing network is severely degraded, or significant aspects of your business operations are negatively affected by inadequate performance of Cisco products. You and Cisco will commit full-time resources during normal business hours to resolve the situation.

Severity 3 (S3)—Operational performance of the network is impaired, while most business operations remain functional. You and Cisco will commit resources during normal business hours to restore service to satisfactory levels.

Severity 4 (S4)—You require information or assistance with Cisco product capabilities, installation, or configuration. There is little or no effect on your business operations.

Obtaining Additional Publications and Information

Information about Cisco products, technologies, and network solutions is available from various online and printed sources.

- The *Cisco Product Quick Reference Guide* is a handy, compact reference tool that includes brief product overviews, key features, sample part numbers, and abbreviated technical specifications for many Cisco products that are sold through channel partners. It is updated twice a year and includes the latest Cisco offerings. To order and find out more about the Cisco Product Quick Reference Guide, go to this URL:

  http://www.cisco.com/go/guide

- Cisco Marketplace provides a variety of Cisco books, reference guides, documentation, and logo merchandise. Visit Cisco Marketplace, the company store, at this URL:

  http://www.cisco.com/go/marketplace/

- *Cisco Press* publishes a wide range of general networking, training and certification titles. Both new and experienced users will benefit from these publications. For current Cisco Press titles and other information, go to Cisco Press at this URL:

  http://www.ciscopress.com

- *Packet* magazine is the Cisco Systems technical user magazine for maximizing Internet and networking investments. Each quarter, Packet delivers coverage of the latest industry trends, technology breakthroughs, and Cisco products and solutions, as well as network deployment and troubleshooting tips, configuration examples, customer case studies, certification and training information, and links to scores of in-depth online resources. You can access Packet magazine at this URL:

  http://www.cisco.com/packet

- *iQ Magazine* is the quarterly publication from Cisco Systems designed to help growing companies learn how they can use technology to increase revenue, streamline their business, and expand services. The publication identifies the challenges facing these companies and the technologies to help solve them, using real-world case studies and business strategies to help readers make sound technology investment decisions. You can access iQ Magazine at this URL:

  http://www.cisco.com/go/iqmagazine

  or view the digital edition at this URL:

  http://ciscoiq.texterity.com/ciscoiq/sample/

- *Internet Protocol Journal* is a quarterly journal published by Cisco Systems for engineering professionals involved in designing, developing, and operating public and private internets and intranets. You can access the Internet Protocol Journal at this URL:

  http://www.cisco.com/ipj

- Networking products offered by Cisco Systems, as well as customer support services, can be obtained at this URL:

  http://www.cisco.com/en/US/products/index.html

- Networking Professionals Connection is an interactive website for networking professionals to share questions, suggestions, and information about networking products and technologies with Cisco experts and other networking professionals. Join a discussion at this URL:

  http://www.cisco.com/discuss/networking

- World-class networking training is available from Cisco. You can view current offerings at this URL:

  http://www.cisco.com/en/US/learning/index.html

Modular Quality of Service Overview on Cisco IOS XR Software

Quality of Service (QoS) is the technique of prioritizing traffic flows and providing preferential forwarding for higher priority packets. The fundamental reason for implementing QoS in your network is to provide better service for certain traffic flows. A traffic flow can be defined as a combination of source and destination addresses, source and destination socket numbers, and the session identifier. A traffic flow can more broadly be described as a packet moving from an incoming interface that is destined for transmission to an outgoing interface. The traffic flow must be identified, classified, and prioritized on all routers and passed along the data forwarding path throughout the network to achieve end-to-end QoS delivery. The terms traffic flow and packet are used interchangeably throughout this module.

To implement QoS on a network requires the configuration of QoS features that provide better and more predictable network service by supporting bandwidth allocation, improving loss characteristics, avoiding and managing network congestion, metering network traffic, or setting traffic flow priorities across the network.

This module contains overview information about how to implement modular QoS features for the network core of a service provider environment.

Note For additional conceptual information about QoS in general and complete descriptions of the QoS commands listed in this module, see the "Related Documents" section of this module. To locate documentation for other commands that might appear in the course of executing a configuration task, search online in the Cisco IOS XR software master command index.

Feature History for Modular QoS Overview on Cisco IOS XR Software

Release	Modification
Release 2.0	This feature was introduced on the Cisco CRS-1 router.
Release 3.0	No modification.
Release 3.2	This feature was first supported on the Cisco XR 12000 Series Router.
Release 3.3.0	No modification.
Release 3.3.1	No modiciation

Contents

Information About Modular Quality of Service Overview on Cisco IOS XR Software

Before configuring modular QoS on your network, you should understand the following concepts:

Benefits Implementing Cisco IOS XR QoS Features

The Cisco IOS XR QoS features enable networks to control and predictably service a variety of networked applications and traffic types. Implementing Cisco IOS XR QoS in your network promotes the following benefits:

- Control over resources. You have control over which resources (bandwidth, equipment, wide-area facilities, and so on) are being used. For example, you can limit bandwidth consumed over a backbone link by FTP transfers or give priority to an important database access.
- Tailored services. If you are an Internet Service Provider (ISP), the control and visibility provided by QoS enables you to offer carefully tailored grades of service differentiation to your customers.
- Coexistence of mission-critical applications. Cisco IOS XR QoS features make certain of the following conditions:
  - That your WAN is used efficiently by mission-critical applications that are most important to your business.
  - That bandwidth and minimum delays required by time-sensitive multimedia and voice applications are available.
  - That other applications using the link get their fair service without interfering with mission-critical traffic.

QoS Techniques on Cisco IOS XR Software

QoS on Cisco IOS XR software relies on the following techniques to provide for end-to-end QoS delivery across a heterogeneous network:

- Packet classification
- Congestion management
- Congestion avoidance

Before implementing the QoS features for these techniques, you should identify and evaluate the traffic characteristics of your network because not all techniques are appropriate for your network environment.

Packet Classification

Packet classification techniques identify the traffic flow, and provide the capability to partition network traffic into multiple priority levels or classes of service. After traffic flow is identified, it can be marked as a traffic class.

Identification of a traffic flow can be performed by using several methods within a single router: access control lists (ACLs), protocol match, IP precedence, IP differentiated service code point (DSCP), and so on.

Marking of a traffic flow is performed by setting IP Precedence bits or the DSCP in the IP Type of Service (ToS) byte.

For detailed conceptual and configuration information about packet classification, see the "Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software" module.

Congestion Management

Congestion management techniques control congestion after it has occurred. One way that network elements handle an overflow of arriving traffic is to use a queueing algorithm to sort the traffic, then determine some servicing method of prioritizing it onto an output link.

Cisco IOS XR QoS software implements the Low Latency Queueing (LLQ) feature, which brings strict priority queueing (PQ) to the Modified Deficit Round Robin (MDRR) scheduling mechanism, providing a guaranteed minimum amount of bandwidth to each traffic class. LLQ with strict PQ allows delay-sensitive data, such as voice, to be dequeued and sent before packets in other queues are dequeued.

For detailed conceptual and configuration information about congestion management, see the "Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software" module.

Congestion Avoidance

Congestion avoidance techniques monitor network traffic flows in an effort to anticipate and avoid congestion at common network and internetwork bottlenecks before problems occur. These techniques are designed to provide preferential treatment for traffic (such as a video stream) that has been classified as real-time critical under congestion situations while concurrently maximizing network throughput and capacity utilization and minimizing packet loss and delay. Cisco IOS XR software supports the Random Early Detection (RED), Weighted RED (WRED), and tail drop QoS congestion avoidance features.

For detailed conceptual and configuration information about congestion avoidance techniques, see the "Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software" module.

Differentiated Service Model for Cisco IOS XR Software

Cisco IOS XR software supports a differentiated service that is a multiple-service model that can satisfy different QoS requirements. However, unlike in the integrated service model, an application using differentiated service does not explicitly signal the router before sending data.

For differentiated service, the network tries to deliver a particular kind of service based on the QoS specified by each packet. This specification can occur in different ways, for example, using the IP Precedence bit settings in IP packets or source and destination addresses. The network uses the QoS specification to classify, mark, shape, and police traffic, and to perform intelligent queueing.

The differentiated service model is used for several mission-critical applications and for providing end-to-end QoS. Typically, this service model is appropriate for aggregate flows because it performs a relatively coarse level of traffic classification.

Additional Cisco IOS XR QoS Supported Features

The following sections describe additional features that play an important role in the implementation of QoS on Cisco IOS XR software.

- Modular QoS Command-Line Interface, page 4
- Traffic Policing, page 4
- Traffic Shaping, page 5

Modular QoS Command-Line Interface

In Cisco IOS XR software, QoS features are enabled through the Modular QoS command-line interface (MQC) feature. The MQC is a command-line interface (CLI) structure that allows users to create traffic policies and attach these polices to interfaces. A traffic policy contains a traffic class and one or more QoS features. A traffic class is used to classify traffic, while the QoS features in the traffic policy determine how to treat the classified traffic. One of the main goals of MQC is to provide a platform-independent interface for configuring QoS across Cisco platforms.

For detailed conceptual and configuration information about the MQC feature, see the "Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software" module.

Traffic Policing

Cisco IOS XR QoS software includes traffic policing capabilities implemented through the rate-limiting aspects of the Traffic Policing feature using class-based shaping.

The Traffic Policing feature limits the input or output transmission rate of a class of traffic based on user-defined criteria, and marks packets by setting the IP Precedence value, QoS group, or DSCP value.

For detailed conceptual and configuration information about the Traffic Policing feature, see the "Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software" module.

Traffic Shaping

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

Cisco IOS XR QoS software supports a class-based traffic shaping method through a CLI mechanism in which parameters are applied per class.

For detailed conceptual and configuration information about the Traffic Shaping feature, see the "Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software" module.

Where to Go Next

To configure the packet classification features that involve identification and marking of traffic flows, see the "Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software" module.

To configure the queueing, scheduling, policing, and shaping features, see the "Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software" module.

To configure the WRED and RED features, see the "Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software" module.

Additional References

The following sections provide references related to implementing QoS on Cisco IOS XR software.

Related Documents

Related Topic	Document Title
Cisco IOS XR QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS XR Modular Quality of Service Command Reference,* Release 3.3
Packet classification	*Configuring Modular QoS Packet Classification on Cisco IOS XR*, Release 3.3
Traffic shaping, traffic policing, low latency queueing, and MDDR	*Configuring QoS Congestion Management on Cisco IOS XR Software,* Release 3.3
WRED, RED, and tail drop	*Configuring QoS Congestion Avoidance on Cisco IOS XR Software,* Release 3.3
Cisco CRS-1 router getting started material	*Cisco CRS-1 Carrier Routing System Getting Started Guide,* Release 3.3
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS-XR Software* module of the *Cisco IOS-XR System Security Configuration Guide,* Release 3.3

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software

Packet classification identifies and marks traffic flows that require congestion management or congestion avoidance on a data path. The Modular Quality of Service (QoS) command-line interface (MQC) is used to define the traffic flows that should be classified, in which each traffic flow is called a *class of service*, or *class*. Subsequently, a traffic policy is created and applied to a class. All traffic not identified by defined classes falls into the category of a default class.

This module provides the conceptual and configuration information for QoS packet classification on Cisco IOS XR software.

Note For additional conceptual information about QoS in general and complete descriptions of the QoS commands listed in this module, see the "Related Documents" section of this module. To locate documentation for other commands that might appear in the course of executing a configuration task, search online in the Cisco IOS XR software master command index.

Feature History for Configuring Modular QoS Packet Classification on Cisco IOS XR Software

Release	Modification
Release 2.0	This feature was introduced on the Cisco CRS-1 router.
Release 3.0	No modification.

Release 3.2	This feature was first supported on the Cisco XR 12000 Series Router with the exception of the following: **match discard-class** command **match protocol** command **match qos-group** command **set cos** command **set qos-group** command Level 2 classes IPv6 addressing was supported for the **match dscp** and **match precedence** commands on the Cisco CRS-1 router only.
Release 3.3.0	The following commands were added: **match cos** **match vlan** The **qos-group** keyword was added to the **set discard-class** command. The **discard-class** keyword was added to the **set qos-group** command. The **not** keyword was added to all **match** commands on the Cisco CRS-1 router except the **match access-group** command. The following features were supported on the Cisco XR 12000 Series Router: **match discard-class** command **match protocol** command **match qos-group** command **set cos** command **set qos-group** command Level 2 classes The Fabric Quality of Service Policies and Classes section was added to the chapter.

Contents

Prerequisites for Configuring Modular QoS Packet Classification on Cisco IOS XR Software

The following prerequisites are required for configuring modular QoS packet classification on your network:

- You must be in a user group associated with a task group that includes the proper task IDs for QoS commands. Task IDs for commands are listed in the *Cisco IOS XR Task ID Reference Guide.*

  For detailed information about user groups and task IDs, see the *Configuring AAA Services on Cisco IOS-XR Software* module of the *Cisco IOS-XR System Security Configuration Guide*.
- You must be familiar with Cisco IOS QoS configuration tasks and concepts.

Information About Configuring Modular QoS Packet Classification on Cisco IOS XR Software

To implement QoS packet classification features in this document, you must understand the following concepts:

- Packet Classification Overview, page 9
- Traffic Class Elements, page 10
- Traffic Policy Elements, page 10
- Default Traffic Class, page 11
- Specification of the CoS for a Packet with IP Precedence, page 15
- Class-Based Packet Marking Feature and Benefits, page 14
- IP Precedence Compared to IP DSCP Marking, page 16

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. For example, by using the three precedence bits in the Type of service (ToS) field of the IP packet header—two of the values are reserved for other purposes—you can categorize packets into a limited set of up to eight traffic classes. 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.

On Cisco IOS XR software, methods of classification may consist of the logical combination of any fields in the packet header. A packet header may be a Layer 2, 3, or 4 header; or classification based on the incoming or outgoing physical or virtual interface.

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, and, if more than one **match** command exists in the traffic class, an instruction on how to evaluate these **match** commands. The traffic class is named in the **class-map** command. For example, if you use the word *cisco* with the **class-map** command, the traffic class would be named *cisco*.

The **match** commands are used to specify various criteria for classifying packets. 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. See the "Default Traffic Class" section in this module.

The instruction on how to evaluate these **match** commands needs to be specified if more than one match criterion exists in the traffic class. The evaluation instruction is specified with the **class-map** *match-any* command. If the *match-any* option is specified as the evaluation instruction, the traffic being evaluated by the traffic class must match at least one of the specified criteria. The *match-any* option is the only evaluation option supported.

The function of these commands is described more thoroughly in the *Cisco IOS XR Modular Quality of Service Command Reference*.

The traffic class configuration task is described in the "Creating a Traffic Class" section of this module.

Traffic Policy Elements

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. The **policy-map** command is used to create a traffic policy. A traffic policy contains three elements: a name, a traffic class (specified with the **class** command), and the QoS policies. The name of a traffic policy is specified in the policy map MQC (for example, issuing the **policy-map** *policy1* command would create a traffic policy named *policy1*). The traffic class that is used to classify traffic to the specified traffic policy is defined in policy map configuration mode. After choosing the traffic class that is used to classify traffic to the traffic policy, the user can enter the QoS features to apply to the classified traffic.

The MQC does not necessarily require that users associate only one traffic class to one traffic policy. When packets match to more than one match criterion, as many as 32 traffic classes can be associated to a single traffic policy. The 32 class-maps include the default-class and the classes of the child policies, if any.

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 function of these commands is described more thoroughly in the *Cisco IOS XR Modular Quality of Service Command Reference*.

The traffic policy configuration task is described in the "Creating a Traffic Policy" section of this module.

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 further information about congestion avoidance techniques, such as tail drop, see the "Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software" module.

Fabric Quality of Service Policies and Classes

Overview

The FabricQ queue selection mechanism is known as Fabric Qos. Three ports are defined; a high-priority port for internal control traffic and classified high-priority traffic,a low-priority port for assured-forwarding (AF) traffic, anda best effort port (BE) for low-priority traffic. Each'port is assigned 1023 queues. The queues map to a physical or logical egress interface and are assigned when the interface is created. The associated quanta for each of the queues are derived from the physical or logical interfaces bandwidth in relative terms to the other interfaces present on that linecard.

By default, internal control traffic is placed in the High priority queue. All other traffic is placed in the best effort queue.

If an ingress QOS policy is configured classifying certain traffic as being priority, then the priority traffic is be placed into the High priority queue together with the internal control traffic. All other traffic is placed in the best effort queue.

The user can configure a Fabric QOS policy which defines the relative MDRR bandwidths associated with the AF and BE ports. This is applied to the logical router (this may be the whole router if no individual service domain routers are configured) and affects all FabricQ ASICs in the logical router.

A maximum of three classes can be specified within the policy. A class known as default is automatically created and equates to the BE port or queues. The name of this class cannot be altered. Any name may be applied to the classes which equate to the Priority and AF ports or queues.

Within the Fabric QoS policy, the only applicable actions are to assign **priority** to the priority class and **bandwidth remaining percent** to the AF and BE classes. The bandwidth remaining percent for the BE class does not need to be specified. It will receive all remaining capacity after the appropriate weight has been allocated to the AF class. The user may wish to specifically enter configuration values for the default class just for clarity.

Fabric QoS policy class maps are restricted to matching a subset of the classification options. These are as follows:

- **precedence**
- **dscp**
- **qos-group**
- **discard-class**
- **mpls experimental topmost**

Note In order to match on **qos-group** or **discard-class**, an ingress QoS policy must be applied setting the required values for **qos-group** or **discard-class**. Both of these variables have local significance only and are not known outside of the router.

Ingress policy and Fabric QoS policy interaction

Care must be taken when applying Ingress QoS policies when they must interact with a Fabric QoS policy. The Fabric QoS policy will override any traffic classification conducted by the Ingress policy when determining which traffic should be placed in the Priority, AF or BE queues within the FabricQ ASIC. In addition, the Fabric QoS policy is used to determine which traffic is placed in the Priority queue within the IngressQ ASIC's fabric queues and the switch fabric ASICs (S2 & S3 stages) rather than any **priority** marking set by the Ingress QoS policy. The **priority** marking is still be used when determining packet scheduling in the shape queues within the IngressQ ASIC.

For example, if an Ingress QoS policy were to classify and mark particular traffic types as being **priority** and a Fabric QoS policy were to be applied either marking alternative traffic as being **priority** or not set **priority** at all, then the Ingress policy's **priority** statement is effectively ignored. Caution should be applied to ensure that there is no conflict between the Ingress QoS policy and the Fabric QoS policy.

A very simplistic illustration would be if an Ingress QoS policy uses a class-map to exclude MPLS EXP 3 from the **priority** class but the Fabric QoS policy places MPLS EXP 3 traffic in the **priority** class. In this case, the MPLS EXP 3 traffic is placed in the high priority S2 and S3 queues and the FabricQ high-priority port or queues.

If the Ingress QoS policy were to remark certain traffic with values that the Fabric QoS policy's class-maps are to match on, then the remarked traffic will be matched and placed in the appropriate port or queues. This provides the ability for the Ingress QoS policy and the Fabric QoS policy to compliment each other, rather than potentially conflicting.

As noted above, Fabric QoS is constrained to a subset of the possible **match** criterion that can be used in its class maps. If the Ingress QoS policy were to set a **qos-group** marking for all traffic which should be placed in the Priority queue and another **qos-group** marking for all traffic to be placed in the AF class, then if the Fabric QoS policy's class maps were to match on the **qos-group** values, the policy will be honored end to end. This approach enables multiple Ingress QoS policies to interact in the expected manner with a Fabric QoS policy.

It is important to remember that if an Ingress Qos policy is applied to an interface and the Fabric QoS policy has been applied to the router, then the Ingress MSC's RX PSE will be required to perform two classification cycles. This will have an impact on the forwarding capacity of the linecard, reducing the performance to about 62.5Mpps.

The user may choose not to apply a specific Fabric QoS policy, giving the Ingress QoS policy the decision on which traffic is placed in the High-priority queues but removing the ability to differentiate between AF and BE classes in the FabricQ Asic.

Examples

The following configuration is an example of a possible Fabric QoS policy:

```
class-map match-any llq
 match mpls experimental topmost 5
 match precedence critical
!
class-map match-any default
 match any
```

```
!
class-map match-any business
 match mpls experimental topmost 3
 match precedence flash
!
policy-map fabric_qos
 class llq
  priority
 !
 class business
  bandwidth remaining percent 65
 !
 class default
  bandwidth remaining percent 35
 !
!
```

Note that if **class default** is specified in the policy-map, then an associated class-map is required.

To apply the policy, use the **switch-fabric service-policy** *policy-name* command.

The following example shows an Ingress QoS policy working in conjunction with a Fabric QoS policy. The Fabric QoS policy is shown first, followed by the Ingress QoS policy:

```
class-map match-any llq
 match qos-group 5
!
class-map match-any default
 match any
!
class-map match-any business&games
 match qos-group 3
!
policy-map fabric_qos
 class llq
  priority
 !
 class business&games
  bandwidth remaining percent 65
 !
 class default
  bandwidth remaining percent 35
 !
!

class-map match-any voip
 match mpls experimental topmost 5
 match precedence critical
 match dscp cs5
!
class-map match-any business
 match mpls experimental topmost 4
 match dscp cs4
 match precedence flash-override
!
class-map match-any broadband-games
 match mpls experimental topmost 3
 match dscp cs3
 march precedence flash
!

Policy-map input-qos
 Class voip
  Priority
  Set qos-group 5
```

```
 Class business
  Set qos-group 3
 Class broadband-games
  Set qos-group 3
```

Note that in the policy-map input-qos, the **priority** keyword under class voip is present for reference purposes rather than configuration reasons.

Class-Based Packet Marking Feature and Benefits

The Class-Based Packet Marking feature provides users with a means for efficient packet marking by which users can differentiate packets based on the designated markings.

The Class-Based Packet Marking feature allows users to perform the following tasks:

- Mark packets by setting the IP Precedence bits or the IP differentiated services code point (DSCP) in the IP ToS byte.
- Associate a local QoS group value to a Multiprotocol Label Switching (MPLS) packet. (This feature is not supported on the Cisco XR 12000 Series Router.)
- Mark packets by setting the Layer 2 Class of Service (CoS) value. (This feature is not supported on the Cisco XR 12000 Series Router.)

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

- Use QoS 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.

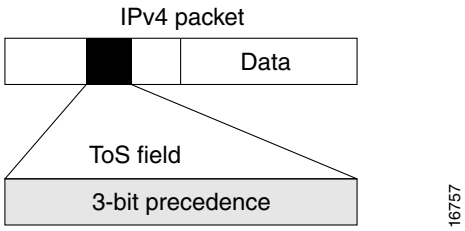
  For example, Weighted Random Early Detection (WRED), a congestion avoidance technique, uses IP Precedence values to determine the probability that a packet is dropped. In addition, Low Latency Queueing (LLQ) can then be configured to put all packets of that mark into the priority queue.
- Use QoS packet 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.
- Use CoS packet marking to assign packets to set the priority value of 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.

The configuration task is described in the "Configuring Class-Based Packet Marking" section of this module.

Specification of the CoS for a Packet with IP Precedence

Use of IP Precedence allows you to specify the CoS for a packet. You use the three precedence bits in the ToS field of the IP version 4 (IPv4) header for this purpose. Figure 1 shows the ToS field.

Figure 1 IPv4 Packet Type of Service Field



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, although IP Precedence is not a queueing feature, queueing features, such as LLQ, can use the IP Precedence setting of the packet to prioritize traffic.

By setting precedence levels on incoming traffic and using them in combination with the Cisco IOS XR QoS queueing features, you can create differentiated service.

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. You can think of IP Precedence as an edge function that allows core, or backbone, QoS features, such as WRED, to forward traffic based on CoS. IP Precedence can also be set in the host or network client, but this setting can be overridden by policy within the network.

The configuration task is described in the "Configuring Class-Based Packet Marking" section of this module.

IP Precedence Bits Used to Classify Packets

You use the three IP Precedence bits in the ToS field of the IP header to specify CoS assignment for each packet. As mentioned earlier, 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.

For historical reasons, each precedence corresponds to a name. These names are defined in the RFC 791 document. Table 2 lists the numbers and their corresponding names, from least to most important.

Table 2 IP Precedence Values

Number	Name
0	routine
1	priority
2	immediate
3	flash
4	flash-override
5	critical
6	internet
7	network

The IP Precedence feature allows you considerable flexibility for precedence assignment. That is, you can define your own classification mechanism. For example, you might want to assign precedence based on application or access router.

Note IP Precedence bit settings 6 and 7 are reserved for network control information, such as routing updates.

IP Precedence Value Settings

By default, the Cisco IOS XR software leaves the IP Precedence value untouched, preserving the precedence value set in the header, allowing 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 precedence set by outside devices, we recommend 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 Packet Marking, LLQ, and WRED features can use the IP precedence bits.

IP Precedence Compared to IP DSCP Marking

IP Precedence and DSCP markings are used to decide how packets should be treated in WRED.

The IP DSCP value is the first six bits in the ToS byte, and the IP Precedence value is the first three bits in the ToS value. The IP Precedence value is actually part of the IP DSCP value. Therefore, both values cannot be set simultaneously. If both values are set simultaneously, the packet is marked with the IP DSCP value.

If you need to mark packets in your network and all of your devices support IP DSCP marking, use the IP DSCP marking to mark your packets because the IP DSCP markings provide more 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 supported by all devices in the network.

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

How to Configure Modular QoS Packet Classification on Cisco IOS XR Software

This section contains instructions for the following tasks:

- Creating a Traffic Class, page 17 (required)
- Creating a Traffic Policy, page 20 (required)
- Attaching a Traffic Policy to an Interface, page 21 (required)
- Configuring Class-Based Packet Marking, page 23 (required)

Creating 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 following **match** commands in class-map configuration mode, as needed.

For conceptual information, see the "Traffic Class Elements" section.

Restrictions

All **match** commands specified in this configuration task are considered optional, but you must configure at least one match criterion for a class.

The **not** keyword is supported only on the Cisco CRS-1 router.

SUMMARY STEPS

1. **configure**
2. **class-map** [**match-any**] *class-map-name*
3. **match access-group** {**ipv4** | **ipv6** | *access-group-name*}
4. **match** [**not**] **any**
5. **match** [**not**] **cos** *cos-value* [*cos-value1 ... cos-value6*]
6. **match** [**not**] **discard-class** *discard-class-value* [*discard-class-value1 ... discard-class-value6*]
7. **match** [**not**] **dscp** [**ipv4** | **ipv6**] *dscp-value* [*dscp-value ... dscp-value*]
8. **match** [**not**] **mpls experimental topmost** *exp-value* [*exp-value1 ... exp-value7*]
9. **match** [**not**] **precedence** [**ipv4** | **ipv6**] *precedence-value* [*precedence-value ... precedence-value*]
10. **match** [**not**] **protocol** {*number* | *name*}
11. **match** [**not**]**qos-group** [*qos-group-value1 ... qos-group-value31*]
12. **match** [**not**] **vlan** [*vlanid* | *beginvlan-endvlan*] [{*vlanid* | *beginvlan-endvlan*} ... {*vlanid* | *beginvlan-endvlan*}]
13. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**class-map** [**match-any**] *class-map-name* **Example:** `RP/0/RP0/CPU0:router(config)# class-map class201`	Enters class map configuration mode. • Creates a class map to be used for matching packets to the class whose name you specify. • If you specific **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.
Step 3	**match access-group** {**ipv4** | **ipv6** | *access-group-name*} **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match access-group map1`	(Optional) Configures the match criteria for a class map based on the specified access control list (ACL) name.
Step 4	**match** [**not**] **any** **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match any`	(Optional) Configures the match criteria for a class map to be successful match criteria for all packets
Step 5	**match** [**not**] **cos** *cos-value* [*cos-value1 ... cos-value6*] **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match cos 1`	(Optional) Specifies a *cos-value* in a class map to match packets. • The *cos-value* argument is specified as an integer from 1 to 7.
Step 6	**match** [**not**] **discard-class** *discard-class-value* [*discard-class-value1 ... discard-class-value6*] **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match discard-class 5`	(Optional) Specifies a *discard-class-value* in a class map to match packets. • The *discard-class-value* argument is specified as an integer from 1 to 7.
Step 7	**match** [**not**] **dscp** [**ipv4** | **ipv6**] *dscp-value* [*dscp-value ... dscp-value*] **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match dscp ipv4 15`	(Optional) Identifies a specific DSCP value as a match criterion. • The value range is from 0 to 63. • Reserved keywords can be specified instead of numeric values.
Step 8	**match** [**not**] **mpls experimental topmost** *exp-value* [*exp-value1 ... exp-value7*] **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match mpls experimental topmoat 3`	(Optional) Configure a class map so that the three-bit experimental field in the topmost Multiprotocol Label Switching (MPLS) labels are examined for experimental (EXP) field values.

	Command or Action	Purpose
Step 9	**match** [**not**] **precedence** [**ipv4** | **ipv6**] *precedence-value* [*precedence-value ... precedence-value*] **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match precedence ipv4 5`	(Optional) Identifies IP precedence values as match criteria. • The value range is from 0 to 63. • Reserved keywords can be specified instead of numeric values.
Step 10	**match** [**not**] **protocol** {*number* | *name*} **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match protocol igmp`	(Optional) Configures the match criteria for a class map on the basis of the specified protocol.
Step 11	**match** [**not**] **qos-group** [*qos-group-value1 ... qos-group-value31*] **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match qos-group 1 2 3 4 5 6 7 8`	(Optional) Specifies service (QoS) group values in a class map to match packets. • A QoS group value identifier argument is specified as the exact value from 1 to 31. • Up to 8 values (separated by spaces) can be entered in one match statement.
Step 12	**match** [**not**] **vlan** [*vlanid* | *beginvlan-endvlan*] [{*vlanid* | *beginvlan-endvlan*} ... {*vlanid* | *beginvlan-endvlan*}] **Example:** `RP/0/RP0/CPU0:router(config-cmap)# match cos 1`	(Optional) Specifies a VLAN ID or range of VLAN IDs in a class map to match packets. • The vlan ID argument is specified as an integer from 0 to 8096.

Step 13	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-cmap)# end`
	or
	`RP/0/RP0/CPU0:router(config-cmap)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Creating a Traffic Policy

To create a traffic policy, use the **policy-map** command to specify the traffic policy name.

The traffic class is associated with the traffic policy when the **class** command is used. The **class** command must be issued after you enter the policy map configuration mode. After entering the **class** command, the router is automatically in policy map class configuration mode, which is where the QoS policies for the traffic policy are defined.

For additional commands that can be entered as match criteria, see the *Cisco IOS XR Modular Quality of Service Command Reference*.

For conceptual information, see the "Traffic Policy Elements" section.

Restrictions

A maximum of 16 classes (including Level 1 and Level 2 hierarchical classes) can be applied to one policy map.

Level 2 hierarchical classes are not supported on the Cisco XR 12000 Series Router.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **police** {**cir** *kbps* | **percent** *percent*} [**bc** *conform-burst*] [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action*] [**pir** *kbps*] [**violate-action** *action*]
5. **set precedence** [*number*]
6. **end**
   or
   commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class1`	Specifies the name of the class whose policy you want to create or change.

	Command or Action	Purpose
Step 4	**police** {**cir** *kbps* | **percent** *percent*} [**bc** *conform-burst*] [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action*] [**pir** *kbps*] [**violate-action** *action*] **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# police rate 250`	Configures traffic policing.
Step 5	**set precedence** [*number*] **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# set precedence 3`	Sets the precedence value in the IP header.

Step 6	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-pmap-c)# end`
	or
	`RP/0/RP0/CPU0:router(config-pmap-c)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Attaching a Traffic Policy to an Interface

After the traffic class and traffic policy are created, you must use the **service-policy** interface configuration command to attach a traffic policy to an interface, and to specify the direction in which the policy should be applied (either on packets coming into the interface or packets leaving the interface).

For additional commands that can be entered in policy map class configuration mode, see the *Cisco IOS XR Modular Quality of Service Command Reference*.

Prerequisites

A traffic class and traffic policy must be created before attaching a traffic policy to an interface.

SUMMARY STEPS

1. **configure**
2. **interface** *type instance*
3. **service-policy** {**input** | **output**} *policy-map*
4. **end**
   or
   commit
5. **show policy-map interface** *type instance* [**input** | **output**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**interface** *type instance* **Example:** `RP/0/RP1/CPU0:router(config)# interface POS 0/2/0/0`	Enters interface configuration mode and configures an interface.
Step 3	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if)# service-policy output policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.

	Command or Action	Purpose

Step 4	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-if)# end`
	or
	`RP/0/RP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 5	**show policy-map interface** *type instance* [**input** | **output**] **Example:** `RP/0/RP0/CPU0:router# show policy-map interface POS 0/2/0/0`	(Optional) Displays policy configuration information for all classes configured for all service policies on the specified interface.

Configuring Class-Based Packet Marking

This configuration task explains how to configure the following class-based packet marking features on your router:

- IP Precedence value
- IP DSCP value
- QoS Group value
- CoS value

Note The QoS group and class of service features are not supported on the Cisco XR 12000 Series Router.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **set precedence** [*number*]
5. **set dscp** *dscp-value*

6. **set qos-group** *qos-group-value* [**discard-class** *discard-class-value*]
7. **set cos** *cos-value*
8. **exit**
9. **exit**
10. **interface** *type instance*
11. **service-policy** {**input** | **output**} *policy-map*
12. **end**
    or
    commit
13. **show policy-map interface** *type instance* [**input** | **output**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class1`	Enters policy class map configuration mode. Specifies the name of the class whose policy you want to create or change.
Step 4	**set precedence** [*number*] **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# set precedence 1`	Sets the precedence value in the IP header.
Step 5	**set dscp** *dscp-value* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# set dscp 5`	Marks a packet by setting the DSCP in the ToS byte.
Step 6	**set qos-group** *qos-group-value* [**discard-class** *discard-class-value*] **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# set qos-group 31`	Sets the QoS group identifiers on IPv4 or MPLS packets.

	Command or Action	Purpose
Step 7	**set cos** *cos-value* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# set cos 7`	Sets the Layer 2 CoS value of an outgoing packet. • This command should be used by a router if a user wants to mark a packet that is being sent to a switch. Switches can leverage Layer 2 header information, including a CoS value marking. • This command can be used only in service policies that are attached in the output direction of an interface. Packets entering an interface cannot be set with a CoS value.
Step 8	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Returns the router to policy map configuration mode.
Step 9	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap)# exit`	Returns the router to global configuration mode.
Step 10	**interface** *type instance* **Example:** `RP/0/RP1/CPU0:router(config)# interface POS 0/2/0/0`	Enters interface configuration mode and configures an interface.
Step 11	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if)# service-policy output policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.

	Command or Action	Purpose

Step 12	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-if)# end`
	or
	`RP/0/RP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 13	**show policy-map interface** *type instance* [**input** | **output**] **Example:** `RP/0/RP0/CPU0:router# show policy-map interface POS 0/2/0/0`	(Optional) Displays policy configuration information for all classes configured for all service policies on the specified interface.

Configuration Examples for Configuring Modular QoS Packet Classification on Cisco IOS XR Software

This section contains the following examples:

Traffic Classes Defined: Example

In the following example, two traffic classes are created and their match criteria are defined. For the first traffic class called class1, ACL 101 is used as the match criterion. For the second traffic class called class2, ACL 102 is used as the match criterion. Packets are checked against the contents of these ACLs to determine if they belong to the class.

```
class-map class1
  match access-group 101
  exit
!
class-map class2
  match access-group 102
  exit
```

Traffic Policy Created: Example

In the following example, a traffic policy called policy1 is defined to contain policy specifications for the two classes—class1 and class2. The match criteria for these classes were defined in the traffic classes (see the "Traffic Classes Defined: Example" section).

For class1, the policy includes a bandwidth allocation request and a maximum byte limit for the queue reserved for the class. For class2, the policy specifies only a bandwidth allocation request.

```
policy-map policy1
  class class1
    bandwidth 3000
    queue-limit bytes 1000000000
    exit
!
  class class2
    bandwidth 2000
    exit
```

Traffic Policy Attached to an Interface: Example

The following example shows how to attach an existing traffic policy to an interface (see the "Traffic Classes Defined: Example" section). 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. Although you can assign the same traffic policy to multiple interfaces, each interface can have only one traffic policy attached at the input and only one traffic policy attached at the output.

```
interface pos 0/1/0/0
  service-policy output policy1
  exit
!
interface TenGigE 0/5/0/1
  service-policy output policy1
  exit
```

Default Traffic Class Configuration: Example

The following example shows how to configure a traffic policy for the default class of the traffic policy called policy1. The default class is named class-default, consists of all other traffic, and is being shaped at 60 percent of the interface bandwidth.

```
policy-map policy1
  class class-default
    shape average percent 60
```

class-map match-any Command Configuration: Example

The following example illustrates how packets are evaluated when multiple match criteria exist. Only one match criterion must be met for the packet in the **class-map match-any** command to be classified as a member of the traffic class (a logical OR operator). In the example, protocol IP OR QoS group 4 OR access group 101 have to be successful match criteria:

```
class-map match-any class1
  match protocol ipv4
  match qos-group 4
  match access-group 101
```

In the traffic class called class1, the match criteria are evaluated consecutively until a successful match criterion is located. The packet is first evaluated to determine whether IPv4 protocol can be used as a match criterion. If IPv4 protocol can be used as a match criterion, the packet is matched to traffic class class1. If IP protocol is not a successful match criterion, then QoS group 4 is evaluated as a match criterion. Each matching criterion is evaluated to see if the packet matches that criterion. Once a successful match occurs, the packet is classified as a member of traffic class class1. If the packet matches at least one of the specified criteria, the packet is classified as a member of the traffic class.

Note The **match protocol** and **match qos-group** commands are not supported on the Cisco XR 12000 Series Router.

Traffic Policy as a QoS Policy (Hierarchical Traffic Policies) Configuration: Example

A traffic policy can be nested within a QoS policy when the **service-policy** command is used in policy map class configuration mode. A traffic policy that contains a nested traffic policy is called a hierarchical traffic policy.

A hierarchical traffic policy contains a child and a parent policy. The child policy is the previously defined traffic policy that is being associated with the new traffic policy through the use of the **service-policy** command. The new traffic policy using the pre-existing traffic policy is the parent policy. In the example in this section, the traffic policy called child is the child policy, and the traffic policy called parent is the parent policy.

Hierarchical traffic policies can be attached to all supported interfaces for this Cisco IOS XR software release, such as the OC-192 and 10-Gigabit Ethernet interfaces.

In the following example, the child policy is responsible for prioritizing traffic, and the parent policy is responsible for shaping traffic. In this configuration, the parent policy allows packets to be sent from the interface, and the child policy determines the order in which the packets are sent.

```
policy-map child
  class mpls
    priority
!
policy-map parent
  class class-default
    shape average 10000000
    service-policy child
```

IP Precedence Value Configuration: Example

In the following example, a service policy called policy1 is created. This service policy is associated to a previously defined classification policy called class1 through the use of the **class** command, and then the service policy is attached to the output Packet-over-SONET interface 0/1/0/0. The IP Precedence bit in the ToS byte is set to 1:

```
policy-map policy1
  class class1
    set precedence 1
!
interface pos 0/1/0/0
  service-policy output policy1
```

IP DSCP Value Configuration: Example

In the following example, a service policy called policy1 is created. This service policy is associated to a previously defined classification policy through the use of the **class** command. In this example, it is assumed that a classification policy called class1 was previously configured.

In the following example, the IP DSCP value in the ToS byte is set to 5:

```
policy-map policy1
  class class1
    set dscp 5

  class class2
    set dscp ef
```

After you configure the settings shown for voice packets at the edge, all intermediate routers are configured to provide low latency treatment to the voice packets, as follows:

```
class-map voice
  match dscp ef
policy qos-policy
  class voice
    priority
```

The service policy configured in this section is not yet attached to an interface. For information on attaching a service policy to an interface, see the "Modular Quality of Service Overview on Cisco IOS XR Software" in this module.

Where to Go Next

To configure class-based traffic shaping, traffic policing, and low latency queueing, see the "Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software" module.

To configure WRED and tail drop, see the "Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software" module.

Additional References

The following sections provide references related to implementing QoS service packet classification on Cisco IOS XR software.

Related Documents

Related Topic	Document Title
Cisco IOS XR QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS XR Modular Quality of Service Command Reference,* Release 3.3
Traffic shaping, traffic policing, low latency queueing, and MDDR	*Configuring QoS Congestion Management on Cisco IOS XR Software,* Release 3.3
WRED, RED, and tail drop	*Configuring QoS Congestion Avoidance on Cisco IOS XR Software,* Release 3.3
Cisco CRS-1 router getting started material	*Cisco CRS-1 Carrier Routing System Getting Started Guide,* Release 3.3
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS-XR Software* module of the *Cisco IOS-XR System Security Configuration Guide,* Release 3.3
Cisco IOS XR QoS overview information	*Modular Quality of Service Overview on Cisco IOS XR Software*, Release 3.3

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Modular Quality of Service Congestion Management on Cisco IOS XR Software

Congestion management controls congestion after it has occurred on a network. Congestion can be managed on Cisco IOS XR software by using packet queueing methods, and by shaping the packet flow through use of traffic regulation mechanisms.

Packet queueing methods define packet scheduling or the order in which packets are dequeued to the interface for transmission on the physical wire. Furthermore, queueing methods support minimum bandwidth guarantees and low latencies based on the order and number of times that a queue is serviced.

The following types of queueing methods and traffic regulation mechanisms are supported on the Cisco IOS XR software:

- Modified Deficit Round Robin (MDRR)
- Low-latency queueing (LLQ) with strict priority queueing (PQ)
- Traffic shaping
- Traffic policing

This module provides the conceptual and configuration information for Quality of Service (QoS) packet queueing methods and traffic regulation mechanisms on Cisco IOS XR software.

Note For additional conceptual information about QoS in general and complete descriptions of the QoS commands listed in this module, see the "Related Documents" section of this module. To locate documentation for other commands that might appear in the course of executing a configuration task, search online in the Cisco IOS XR software master command index.

Feature History for Configuring Modular QoS Congestion Management on Cisco IOS XR Software Contents

Release	Modification
Release 2.0	This feature was introduced on the Cisco CRS-1 router.
Release 3.0	No modification.
Release 3.2	This feature was first supported on the Cisco XR 12000 Series Router with the exception of the following: Two-rate policer and two-token bucket algorithm. The **pir** and **violate-action** keywords for the **police** command. Packet-by-packet MDRR scheduling mechanism. Added configuration examples.
Release 3.3.0	The **police** command was updated in examples. Step added to the "Configuring Low-Latency Queueing with Strict Priority Queueing" section on page 44.
Release 3.3.1	The **bandwidth**, **police**, and **shape average** commands were updated in the Summary Steps.

Contents

Prerequisites for Configuring QoS Congestion Management on Cisco IOS XR Software

The following prerequisites are required for configuring QoS congestion management on your network:

- You must be in a user group associated with a task group that includes the proper task IDs for QoS commands. Task IDs for commands are listed in the *Cisco IOS XR Task ID Reference Guide*.

  For detailed information about user groups and task IDs, see the *Configuring AAA Services on Cisco IOS-XR Software* module of the *Cisco IOS-XR System Security Configuration Guide*.
- You must be familiar with Cisco IOS QoS configuration tasks and concepts.

Information About Configuring QoS Congestion Management on Cisco IOS XR Software

To implement QoS congestion management features in this document, you must understand the following concepts:

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 congestion management features in Cisco IOS XR software allow you to specify creation of a different number of queues, affording greater or lesser degree of differentiation of traffic, and to specify the order in which that traffic is sent.

During periods with light traffic flow, that is, when no congestion exists, packets are sent out the interface as soon as they arrive. During periods of transmit congestion at the outgoing interface, packets arrive faster than the interface can send them. If you use congestion management features, packets accumulating at an interface are queued until the interface is free to send them; they are then scheduled for transmission according to their assigned priority and the queueing method configured for the interface. The router determines the order of packet transmission by controlling which packets are placed in which queue and how queues are serviced with respect to each other.

In addition to queueing methods, QoS congestion management mechanisms, such as policers and shapers, are needed to ensure that a packet adheres to a contract and service. Both policing and shaping mechanisms use the traffic descriptor for a packet. See the "Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software" module for information about the traffic descriptor.

Policers and shapers usually identify traffic descriptor violations in an identical manner through the token bucket mechanism, but they differ in the way they respond to violations. A policer typically drops traffic flow; whereas, a shaper delays excess traffic flow using a buffer, or queueing mechanism, to hold Low-Latency Queueing feature that uses a strict priority queue with policing to shape the flow.)

Traffic shaping and policing can work in tandem. For example, a good traffic shaping scheme should make it easy for nodes inside the network to detect abnormal flows. This activity is sometimes called policing the traffic of the flow.

Modified Deficit Round Robin

MDRR is a class-based composite scheduling mechanism that allows for queueing of up to eight traffic classes. It operates in the same manner as class-based weighted fair queueing (CBWFQ) and allows definition of traffic classes based on customer match criteria (such as access lists); however, MDRR does not use the weighted fair queueing algorithm.

With MDRR configured in the queueing strategy, nonempty queues are served one after the other. Each time a queue is served, a fixed amount of data is dequeued. The algorithm then services the next queue. When a queue is served, MDDR keeps track of the number of bytes of data that were dequeued in excess of the configured value. In the next pass, when the queue is served again, less data is dequeued to compensate for the excess data that was served previously. As a result, the average amount of data dequeued per queue is close to the configured value. In addition, MDRR allows for a strict priority queue for delay-sensitive traffic.

Each queue within MDRR is defined by two variables:

- Quantum value—Average number of bytes served in each round.
- Deficit counter—Number of bytes a queue has sent in each round. The counter is initialized to the quantum value.

Packets in a queue are served as long as the deficit counter is greater than zero. Each packet served decreases the deficit counter by a value equal to its length in bytes. A queue can no longer be served after the deficit counter becomes zero or negative. In each new round, the deficit counter for each nonempty queue is incremented by its quantum value.

Note In general, the quantum size for a queue should not be smaller than the maximum transmission unit (MTU) of the interface to ensure that the scheduler always serves at least one packet from each nonempty queue.

The Cisco CRS-1 router implements a slight variation of the MDRR scheduling mechanism called packet-by-packet MDRR (P2MDRR). Using P2MDRR, queues are scheduled after every packet is sent compared to MDRR in which queues are scheduled after a queue is emptied. All non-high-priority queues with minimum bandwidth guarantees use P2MDRR.

Low-Latency Queueing with Strict Priority Queueing

The LLQ feature brings strict PQ to the MDRR scheduling mechanism. 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.

Strict PQ allows delay-sensitive data, such as voice, to be dequeued and sent before packets in other queues are dequeued.

LLQ enables use of a single, strict priority queue within MDRR at the class level, allowing you to direct traffic belonging to a class. To rank class traffic to the strict priority queue, you specify the named class within a policy map and then configure the **priority** command for the class. (Classes to which the **priority** command is applied are considered priority classes.) Within a policy map, you can give one or more classes priority status. When multiple classes within a single policy map are configured as priority classes, all traffic from these classes is enqueued to the same, single, strict priority queue.

Through use of the **priority** command, you can assign a strict PQ to any of the valid match criteria used to specify traffic. These methods of specifying traffic for a class include matching on access lists, protocols, IP precedence, and IP differentiated service code point (DSCP) values. Moreover, within an access list you can specify that traffic matches are allowed based on the DSCP value that is set using the first six bits of the IP type of service (ToS) byte in the IP header.

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, thereby eliminating bottlenecks in topologies with data-rate mismatches.

To match the rate of transmission of data from the source to the target interface, you can limit the transfer of data to one of the following:

- A specific configured rate
- A derived rate based on the level of congestion

The rate of transfer depends on these three components that constitute the token bucket: burst size, mean rate, and time (measurement) interval. The mean rate is equal to the burst size divided by the interval.

When traffic shaping is enabled, the bit rate of the interface does not exceed the mean rate over any integral multiple of the interval. In other words, during every interval, a maximum of burst size can be sent. Within the interval, however, the bit rate may be faster than the mean rate at any given time.

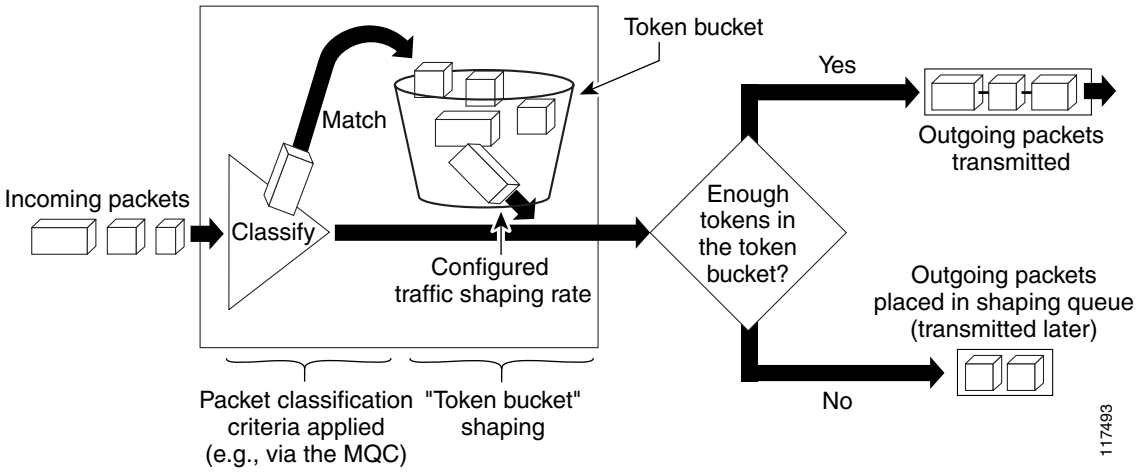
When the exceed burst size (Be) equals 0, the interface sends no more than the burst size every interval, achieving an average rate no higher than the mean rate. However, when the Be size is greater than 0, the interface can send as many as the conform burst size (Bc) plus Be bits in a burst, if in a previous time period the maximum amount was not sent. Whenever less than the burst size is sent during an interval, the remaining number of bits, up to the Be size, can be used to send more than the burst size in a later interval.

Traffic Shaping Mechanism Regulates Traffic

When incoming packets arrive at an interface. The packets are classified using a classification technique, such as use of an access control list (ACL) or setting of the IP Precedence bits through the Modular QoS CLI (MQC). If the packet matches the specified classification, the traffic shaping mechanism continues. Otherwise, no further action is taken.

Figure 2 illustrates how a traffic shaping mechanism regulates traffic flow.

Figure 2 How a Traffic Shaping Mechanism Regulates Traffic



Packets matching the specified criteria are placed in the token bucket. The maximum size of the token bucket is the confirm burst (Bc) size plus the Be size. The token bucket is filled at a constant rate of Bc worth of tokens at every Tc. This is the configured traffic shaping rate.

If the traffic shaping mechanism is active (that is, packets exceeding the configured traffic shaping rate already exist in a transmission queue) at every Tc, the traffic shaper checks to see if the transmission queue contains enough packets to send (that is, up to either Bc [or Bc plus Be] worth of traffic).

If the traffic shaper is not active (that is, there are no packets exceeding the configured traffic shaping rate in the transmission queue), the traffic shaper checks the number of tokens in the token bucket. One of the following occurs:

- If there are enough tokens in the token bucket, the packet is sent (transmitted).
- If there are not enough tokens in the token bucket, the packet is placed in a shaping queue for transmission at a later time.

Traffic Policing

In general, 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 can use the 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 (depending on where the traffic policy with traffic policing is configured) and is useful in managing network bandwidth in cases in which several large packets are sent in the same traffic stream.

Traffic entering the interface with traffic policing configured is placed into one of these categories. Within these three categories, users can decide packet treatments. For instance, packets that conform can be configured to be sent, packets that exceed can be configured to be sent with a decreased priority, and packets that violate can be configured to be dropped.

Traffic policing is often configured on interfaces at the edge of a network to limit the rate of traffic entering or leaving the network. In the most common traffic policing configurations, traffic that conforms is sent and traffic that exceeds is sent with a decreased priority or is dropped. Users can change these configuration options to suit their network needs.

For Cisco IOS XR software, a single-rate, two-color policer is supported that provides one token bucket with two actions for each packet: a conform action and an exceed action. Traffic policing provides a certain amount of bandwidth management by allowing you to set the burst size (Bc) for the committed information rate (CIR). When the peak information rate (PIR) is supported, a second token bucket is enforced and the traffic policer is then called a two-rate policer.

Note The two-rate policer and two-token bucket algorithm is not supported on this release of Cisco IOS XR software.

Packet Marking Through the IP Precedence Value, IP DSCP Value, and the MPLS Experimental Value Setting

In addition to rate-limiting, traffic policing allows you to independently mark (or classify) the packet according to whether the packet conforms or violates a specified rate. Packet marking also allows you to partition your network into multiple priority levels or CoS.

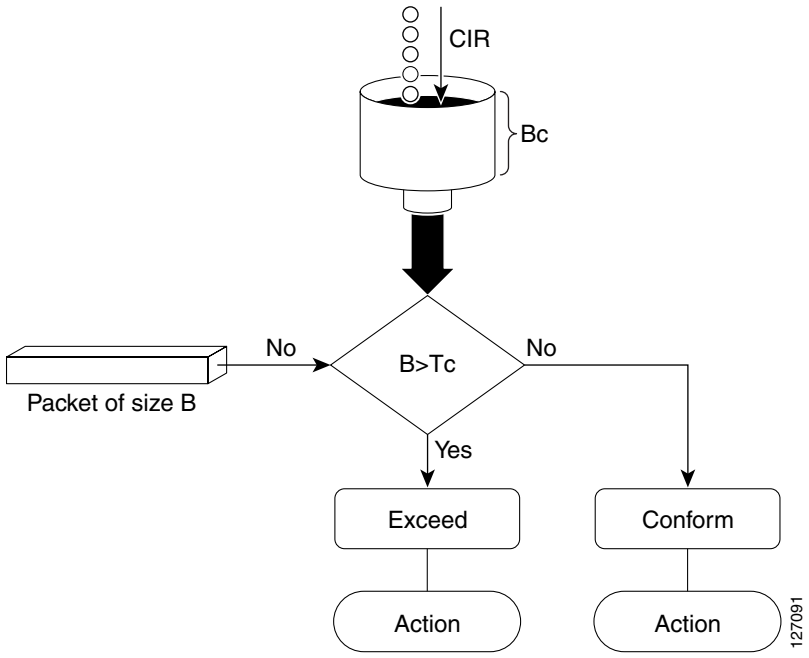
Use the traffic policer to set the IP precedence value, IP DSCP value, or Multiprotocol Label Switching (MPLS) experimental value for packets that enter the network. Then networking devices within your network can use this setting to determine how the traffic should be treated. For example, the Weighted Random Early Detection (WRED) feature uses the IP precedence value to determine the probability that a packet is dropped.

If you want to mark traffic but do not want to use traffic policing, see "Class-Based Packet Marking Feature and Benefits" to learn how to perform packet classification.

Traffic Policing Mechanism Regulates Traffic

Figure 3 illustrates how a single-rate token bucket policer marks packets as either conforming or exceeding a CIR.

Figure 3 How a Traffic Policing Mechanism Regulates Traffic



The time interval between token updates (Tc) to the token bucket is updated at the CIR value each time a packet arrives at the traffic policer. The Tc token bucket can contain up to the Bc value. If a packet of size B is greater than the Tc token bucket, then the packet exceeds the CIR value and a configured action is performed. If a packet of size B is less than the Tc token bucket, then the packet conforms and a different configured action is performed.

Traffic Shaping Versus Traffic Policing

Although traffic shaping and traffic policing can be implemented together on the same network, there are distinct differences between them, as shown in Table 3.

Table 3 Differences Between Traffic Shaping and Traffic Policing

	Traffic Shaping	Traffic Policing
Triggering Event	• Occurs automatically at regular intervals (Tc). or Occurs whenever a packet arrives at an interface.	• Occurs whenever a packet arrives at an interface.
What it Does	• Classifies packets. • If a packet does not meet match criteria, no further action is taken. • Packets meeting match criteria are sent (if there are enough tokens in the token bucket) or Packets are placed in a queue for transmission later. • If the number of packets in the queue exceed the queue limit, the packets are dropped.	• Classifies packets. • If packet does not meet match criteria, no further action is taken. • Packets meeting match criteria and conforming to or exceeding a specified rate, receive the configured policing action (for example, drop, send, mark, then send). • Packets are not placed in a queue for transmission later.

How to Configure QoS Congestion Management on Cisco IOS XR Software

This section contains instructions for the following tasks:

- Configuring Guaranteed and Remaining Bandwidths, page 41 (required)
- Configuring Low-Latency Queueing with Strict Priority Queueing, page 44 (required)
- Configuring Traffic Shaping, page 47 (required)
- Configuring Traffic Policing, page 49 (required)

Configuring Guaranteed and Remaining Bandwidths

The **bandwidth** command allows you to specify the exact amount of bandwidth to be allocated for a specific class of traffic. MDRR is implemented as the scheduling algorithm.

The **bandwidth remaining** command specifies a weight for the class to the MDRR. The MDRR algorithm derives the weight for each class from the bandwidth remaining value allocated to the class. If you do not configure the **bandwidth remaining** command for any class, the leftover bandwidth is allocated equally to all classes.

Restrictions

The amount of bandwidth configured should be large enough to also accommodate Layer 2 overhead.

A policy map can have all class bandwidths specified in kilobits per second or percentages but not a mixture of both in the same class.

The **bandwidth** command is supported only on policies configured on outgoing interfaces.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **bandwidth** {*kbps* | **percent** *value*}
5. **bandwidth remaining percent** *value*
6. **exit**
7. **class** *class-name*
8. **bandwidth** {*kbps* | **percent** *value*}
9. **bandwidth remaining percent** *value*
10. **exit**
11. **exit**
12. **interface** *type instance*
13. **service-policy** {**input** | **output**} *policy-map*
14. **end**
    or
    commit
15. **show policy-map interface** *type instance* [**input** | **output**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class1`	Specifies the name of the class whose policy you want to create or change.

	Command or Action	Purpose
Step 4	**bandwidth** {*kbps* | **percent** *value*} **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# bandwidth percent 50`	Enters policy map class configuration mode. • Specifies the bandwidth allocated for a class belonging to a policy map. • In this example, class class1 is guaranteed 50 percent of the interface bandwidth.
Step 5	**bandwidth remaining percent** *value* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# bandwidth remaining percent 20`	Specifies how to allocate leftover bandwidth to various classes. • The remaining bandwidth of 40 percent is shared by class class1 and class2 (see Steps 8 and 9) in a 20:80 ratio: class class1 receives 20 percent of the 40 percent, and class class2 receives 80 percent of the 40 percent.
Step 6	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Returns the router to policy map configuration mode.
Step 7	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class2`	Specifies the name of a different class whose policy you want to create or change.
Step 8	**bandwidth** {*kbps* | **percent** *value*} **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# bandwidth percent 10`	Specifies the bandwidth allocated for a class belonging to a policy map. • In this example, class class2 is guaranteed 10 percent of the interface bandwidth.
Step 9	**bandwidth remaining percent** *value* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# bandwidth remaining percent 80`	Specifies how to allocate leftover bandwidth to various classes. • The remaining bandwidth of 40 percent is shared by class class1 (see Steps 4 and 5) and class2 in a 20:80 ratio: class class1 receives 20 percent of the 40 percent, and class class2 receives 80 percent of the 40 percent.
Step 10	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Returns the router to policy map configuration mode.
Step 11	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap)# exit`	Returns the router to global configuration mode.
Step 12	**interface** *type instance* **Example:** `RP/0/RP1/CPU0:router(config)# interface POS 0/2/0/0`	Enters interface configuration mode and configures an interface.

	Command or Action	Purpose
Step 13	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if)# service-policy output policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.

Step 14	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-if)# end`
	or
	`RP/0/RP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 15	**show policy-map interface** *type instance* [**input** | **output**] **Example:** `RP/0/RP0/CPU0:router# show policy-map interface POS 0/2/0/0`	(Optional) Displays policy configuration information for all classes configured for all service policies on the specified interface.

Configuring Low-Latency Queueing with Strict Priority Queueing

The priority command configures low-latency queueing (LLQ), providing strict priority queueing (PQ). Strict PQ allows delay-sensitive data, such as voice, to be dequeued and sent before packets in other queues are dequeued. When a class is marked as high priority using the **priority** command, we recommend that you configure a policer to limit the priority traffic. This configuration ensures that the priority traffic does not starve all of the other traffic on the line card, which protects low priority traffic from starvation. Use the **police** command to configure the policer explicitly.

Restrictions

Within a policy map, you can give one or more classes priority status. When multiple classes within a single policy map are configured as priority classes, all traffic from these classes is queued to the same single priority queue.

The **bandwidth**, **priority**, and **shape average** commands should not be configured in the same class. See *Cisco IOS XR Modular Quality of Service Command Reference* for important command details.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **police** {**rate** {*rate* [**bps** | **gbps** | **kbps** | **mbps**] | **percent** *percent*} [**burst** *burst-size*] [**peak-burst** *peak-burst*] [**conform-action** *action*] [**exceed-action** *action*] [**peak-rate** *kbps*] [**violate-action** *action*]
5. **priority**
6. **exit**
7. **exit**
8. **interface** *type instance*
9. **service-policy** {**input** | **output**} *policy-map*
10. **end**
    or
    commit
11. **show policy-map interface** *type instance* [**input** | **output**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map voice`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class voice`	Enters policy map class configuration mode. • Specifies the name of the class whose policy you want to create or change.
Step 4	**police** {**rate** {*kbps* | **percent** *percent*}} [**burst** *burst-size*] [**peak-burst** *peak-burst*] [**conform-action** *action*] [**exceed-action** *action*] [**peak-rate** *kbps*] [**violate-action** *action*] **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# police rate 250 exceed-action drop`	Configures traffic policing. • In this example, the low-latency queue is restricted to 250 kbps to protect low-priority traffic from starvation and to release bandwidth.
Step 5	**priority** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# priority`	Specifies priority to a class of traffic belonging to a policy map.

	Command or Action	Purpose
Step 6	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Returns the router to policy map configuration mode.
Step 7	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap)# exit`	Returns the router to global configuration mode.
Step 8	**interface** *type instance* **Example:** `RP/0/RP1/CPU0:router(config)# interface POS 0/2/0/0`	Enters interface configuration mode, and configures an interface.
Step 9	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if)# service-policy output voice`	Attaches a policy map to an input interface or an output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.

Step 10	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-if)# end`
	or
	`RP/0/RP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 11	**show policy-map interface** *type instance* [**input** | **output**] **Example:** `RP/0/RP0/CPU0:router# show policy-map interface POS 0/2/0/0`	(Optional) Displays policy configuration information for all classes configured for all service policies on the specified interface.

Configuring Traffic Shaping

Traffic shaping allows you to control the traffic 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.

Shaping performed on outgoing interfaces is done at the Layer 2 level and includes the Layer 2 header in the rate calculation.

Shaping performed on incoming interfaces is done at the Layer 3 level and does not include the Layer 2 header in the rate calculation.

Restrictions

The **bandwidth**, **priority**, and **shape average** commands should not be configured together in the same class.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **shape average** {*rate* [**bp**s | **gbps** | **kbps** | **mbps**] | **percent** *percent*}
5. **exit**
6. **exit**
7. **interface** *type instance*
8. **service-policy** {**input** | **output**} *policy-map*
9. **end**
   or
   commit
10. **show policy-map interface** *type instance* [**input** | **output**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.

	Command or Action	Purpose
Step 3	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class1`	Enters policy map class configuration mode. • Specifies the name of the class whose policy you want to create or change.
Step 4	**shape average** {*rate* [**bps** | **gbps** | **kbps** | **mbps**] | **percent** *percent*} **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# shape average percent 50`	Shapes traffic to the indicated bit rate according to average rate shaping in kilobits per second or the interface bandwidth in percentage. In this release, the minimum is 256 kbps, so even if you enter 1000 bps, the minimum of 256 kbps is the actual rate. In the example, the average is expressed as a percent.
Step 5	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Returns the router to policy map configuration mode.
Step 6	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap)# exit`	Returns the router to global configuration mode.
Step 7	**interface** *type instance* **Example:** `RP/0/RP1/CPU0:router(config)# interface POS 0/2/0/0`	Enters interface configuration mode and configures an interface.
Step 8	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if)# service-policy output policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.

	Command or Action	Purpose

Step 9	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-if)# end`
	or
	`RP/0/RP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 10	**show policy-map interface** *type instance* [**input** | **output**] **Example:** `RP/0/RP0/CPU0:router# show policy-map interface POS 0/2/0/0`	(Optional) Displays policy configuration information for all classes configured for all service policies on the specified interface.

Configuring Traffic Policing

Traffic policing allows you to control the maximum rate of traffic sent or received on an interface.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **police rate** {*rate* [**bps** | **gbps** | **kbps** | **mbps**] | **percent** *percent*} [**burst** *burst-size*] [**peak-burst** *peak-burst*] [**conform-action** *action*] [**exceed-action** *action*] [**peak-rate** *kbps*] [**violate-action** *action*]
5. **exit**
6. **exit**
7. **interface** *type instance*
8. **service-policy** {**input** | **output**} *policy-map*

9. **end**
   or
   commit
10. **show policy-map interface** *type instance* [**input** | **output**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class1`	Enters policy map class configuration mode. • Specifies the name of the class whose policy you want to create or change.
Step 4	**police** {*rate* [**bps** | **gbps** | **kbps** | **mbps**] | **percent** *percent*} [**burst** *burst-size*] [**peak-burst** *peak-burst*] [**conform-action** *action*] [**exceed-action** *action*] [**peak-rate** *kbps*] [**violate-action** *action*] **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# police rate 250 conform-action set mpls experimental topmost 3 exceed-action set mpls experimental topmost 4`	Configures traffic policing. The traffic policing feature works with a token bucket algorithm. The *action* argument is specified by one of the following keywords: • **drop**—Drops the packet. • **set dscp dscp-value**—Sets the DSCP value and sends the packet. • **set prec new-precedence**—Sets the IP precedence and sends the packet. • **set mpls experimental topmost**—Sets the EXP value on the MPLS packet topmost label • **set discard class**—Sets the discard class and QoS group identifiers on IP Version 4 (IPv4) or Multiprotocol Label Switching (MPLS) packets.
Step 5	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Returns the router to policy map configuration mode.
Step 6	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap)# exit`	Returns the router to global configuration mode.

	Command or Action	Purpose
Step 7	**interface** *type instance* **Example:** `RP/0/RP0/CPU0:router(config)# interface POS 0/5/0/0`	Enters configuration mode and configures an interface.
Step 8	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if) service-policy input policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.

Step 9	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-if)# end`
	or
	`RP/0/RP0/CPU0:router(config-if)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Step 10	**show policy-map interface** *type instance* [**input** | **output**] **Example:** `RP/0/RP0/CPU0:router# show policy-map interface POS 0/2/0/0`	(Optional) Displays policy configuration information for all classes configured for all service policies on the specified interface.

Configuration Examples for Configuring QoS Congestion Management on Cisco IOS XR Software

This section provides the following configuration examples:

Traffic Shaping for an Input Interface on Cisco IOS XR Software: Example

The following example shows how to configure a policy map on an input interface:

```
policy-map p2
 class voip
 shape average percent 20

!
interface bundle-pos 1
 service-policy input p2
 commit
RP/0/RP0/CPU0:Jun 8 16:55:11.819 : config[65546]: %MGBL-LIBTARCFG-6-COMMIT : Configuration
committed by user 'cisco'. Use 'show configuration commit changes 1000006140' to view the
changes.
```

The following example shows the display output for the previous policy map configuration:

```
RP/0/RP0/CPU0:router# show policy-map interface bundle-pos 1 input

Bundle-POS1 input: p2
  Class voip
    Classification statistics                      (packets/bytes)      (rate - kbps)
            Matched                              : 0/0      0
            Transmitted                          : 0/0      0
            Total Dropped                        : 0/0      0
    Queueing statistics
            Vital           (packets)            : 0
    Queueing statistics
            Queue ID                             : 38
            High watermark  (packets)            : 0
            Inst-queue-len  (bytes)              : 0
            Avg-queue-len   (bytes)              : 0
            TailDrop Threshold(bytes)            : 47923200
            Taildropped(packets/bytes)           : 0/0
  Class default
    Classification statistics                      (packets/bytes)      (rate - kbps)
            Matched                              : 0/0      0
            Transmitted                          : 0/0      0
            Total Dropped                        : 0/0      0
    Queueing statistics
            Vital           (packets)            : 0
    Queueing statistics
            Queue ID                             : 36
            High watermark  (packets)            : 0
            Inst-queue-len  (bytes)              : 0
            Avg-queue-len   (bytes)              : 0
            TailDrop Threshold(bytes)            : 239616000
            Taildropped(packets/bytes)           : 0/0
```

Traffic Policing for a Bundled Interface on Cisco IOS XR Software: Example

The following example shows how to configure a policy map for a bundled interface. Note that for bundled interfaces, policing can be configured only in percentage and not kilobits per second:

```
policy-map p2
 class voip
 police rate 23425
!
interface bundle-poS 1
 service-policy input p2
```

```
 commit
RP/0/RP0/CPU0:Jun  8 16:51:36.623 : qos_ma[286]: %QOS-QOS_RC_QOSMGR-3-RC_BUNDLE_BW_NOPCT :
Absolute bw specified for bundle interfaces, use percentage values instead
RP/0/RP0/CPU0:Jun  8 16:51:36.624 : qos_ma[286]: %QOS-QOS-3-MSG_SEND_FAIL : Failed to send
message to feature rc while adding class. Error code - Invalid argument

% Failed to commit one or more configuration items during an atomic operation, no changes
have been made. Please use 'show configuration failed' to view the errors
!
```

An error occurred after the attempted commit of an invalid configuration.

```
!
!
policy-map p2
 class voip
 police rate percent 20
 commit
RP/0/RP0/CPU0:Jun  8 16:51:51.679 : config[65546]: %MGBL-LIBTARCFG-6-COMMIT :
Configuration committed by user 'cisco'.   Use 'show configuration commit changes
1000006135' to view the changes.
 exit
exit
interface bundle-poS 1
 service-policy input p2
 commit
RP/0/RP0/CPU0:Jun  8 16:52:02.650 : config[65546]: %MGBL-LIBTARCFG-6-COMMIT :
Configuration committed by user 'cisco'.   Use 'show configuration commit changes
1000006136' to view the changes.
```

The following example shows the display output for the successful policy map configuration in which policing was configured in percentage:

```
RP/0/RP0/CPU0:router# show policy-map interface bundle-poS 1

Bundle-POS1 input: p2
  Class voip
    Classification statistics                    (packets/bytes)      (rate - kbps)
            Matched                             : 0/0      0
    Policing statistics                          (packets/bytes)      (rate - kbps)
            Policed(conform)                    : 0/0   0
            Policed(exceed)                     : 0/0   0
            Policed(violate)                    : 0/0   0
            Policed and dropped                 : 0/0
  Class default
    Classification statistics                    (packets/bytes)      (rate - kbps)
            Matched                             : 0/0      0
            Transmitted                         : 0/0      0
            Total Dropped                       : 0/0      0
    Queueing statistics
            Vital           (packets)           : 0
    Queueing statistics
            Queue ID                            : 36
            High watermark  (packets)           : 0
            Inst-queue-len  (bytes)             : 0
            Avg-queue-len   (bytes)             : 0
            TailDrop Threshold(bytes)           : 239616000
            Taildropped(packets/bytes)          : 0/0
```

Where to Go Next

To configure WRED and tail drop, see the "Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software" module.

To configure traffic classification techniques, see the "Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software" module.

To read Cisco IOS XR QoS overview information, see the "Modular Quality of Service Overview on Cisco IOS XR Software" module.

Additional References

The following sections provide references related to implementing QoS congestion management on Cisco IOS XR software.

Related Documents

Related Topic	Document Title
Packet classification	*Configuring Modular QoS Packet Classification on Cisco IOS XR*, Release 3.3
WRED, RED, and tail drop	*Configuring QoS Congestion Avoidance on Cisco IOS XR Software,* Release 3.3
Cisco CRS-1 router getting started material	*Cisco CRS-1 Carrier Routing System Getting Started Guide,* Release 3.3
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS-XR Software* module of the *Cisco IOS-XR System Security Configuration Guide,* Release 3.3
Cisco IOS XR QoS overview information	*Modular Quality of Service Overview on Cisco IOS XR Software*, Release 3.3

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	—

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software

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. Cisco IOS XR software supports the following Quality of Service (QoS) congestion avoidance techniques that drop packets:

- Random Early Detection (RED)
- Weighted Random Early Detection (WRED)
- Tail Drop

The module describes the concepts and tasks related to these congestion avoidance techniques.

Note For additional conceptual information about QoS in general and complete descriptions of the QoS commands listed in this module, see the "Related Documents" section of this module. To locate documentation for other commands that might appear in the course of executing a configuration task, search online in the Cisco IOS XR software master command index.

Feature History for Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software

Release	Modification
Release 2.0	This feature was introduced on the Cisco CRS-1 router.
Release 3.0	No modification.
Release 3.2	This feature was first supported on the Cisco XR 12000 Series Router.
Release 3.3.0	No modification.

Contents

Information About Configuring Modular QoS Congestion Avoidance on Cisco IOS XR

To configure QoS congestion avoidance techniques in this document you must understand the following concepts:

Random Early Detection and TCP

The 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.

Weighted Random Early Detection for Preferential Traffic Handling

WRED combines the capabilities of the RED algorithm with the IP Precedence feature to provide for preferential traffic handling of higher priority packets. WRED can selectively discard lower priority traffic when the interface begins to get congested and provide differentiated performance characteristics for different classes of service. You can configure WRED to ignore IP precedence when making drop decisions so that nonweighted RED behavior is achieved.

WRED makes early detection of congestion possible and provides for multiple classes of traffic. It also protects against global synchronization. For these reasons, WRED is useful on any output interface in which you expect congestion to occur.

However, WRED is usually used in the core routers of a network, rather than at the edge of the network. Edge routers assign IP precedences to packets as they enter the network. WRED uses these precedences to determine how to treat different types of traffic.

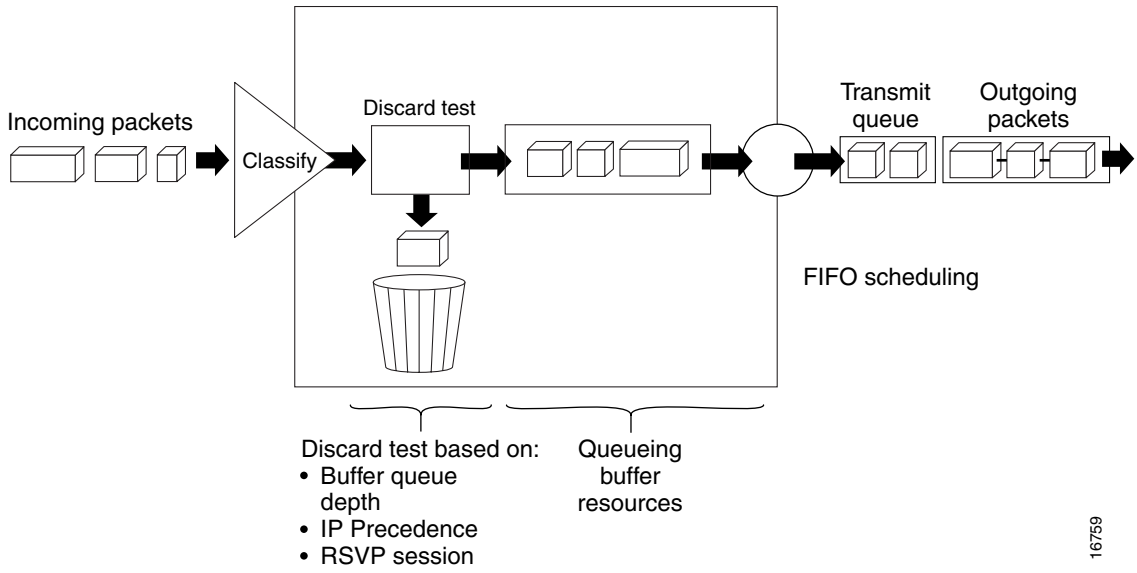
WRED provides separate thresholds and weights for different IP precedences, allowing you to provide different qualities of service in regard to packet dropping for different traffic types. Standard traffic may be dropped more frequently than premium traffic during periods of congestion.

WRED treats non-IP traffic as precedence 0, the lowest precedence. Therefore, non-IP traffic, in general, is more likely to be dropped than IP traffic.

WRED is useful only when the bulk of the traffic is TCP/IP traffic. With TCP, dropped packets indicate congestion, so the packet source reduces its transmission rate. With other protocols, packet sources may not respond or may resend dropped packets at the same rate. Thus, dropping packets does not decrease congestion.

Figure 4 illustrates how WRED works.

Figure 4 Weighted Random Early Detection



Average Queue Size for WRED

The router automatically determines 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 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.

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.

If the value of *x* gets too low, WRED overreacts to temporary traffic bursts and drops traffic unnecessarily.

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.

See "Default Traffic Class" in the *Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software* module.

How to Configure Modular QoS Congestion Avoidance on Cisco IOS XR

This section contains instructions for the following tasks:

- Configuring NonWeighted WRED or RED, page 60 (required)
- Configuring Weighted Random Early Detection, page 63 (required)
- Configuring Tail Drop, page 66 (required)

Configuring NonWeighted WRED or RED

You can configure WRED to ignore IP precedence when making drop decisions so that nonweighted RED behavior is achieved.

This configuration task is similar to that used for WRED except that the **random-detect precedence** command is not configured and **random-detect** [*default*] command must be used to enable RED.

Note The **random-detect** [*default*] command is not supported on the Cisco XR 12000 Series Router.

Restrictions

When configuring the **random-detect dscp** command, you must configure one of the following commands: **shape average**, **bandwidth**, and **bandwidth remaining**.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **random-detect** [*default*]
5. **random-detect dscp** *dscp-value min-threshold max-threshold*
6. **bandwidth** {*bandwidth-kbps* | **percent** *value*}
7. **bandwidth remaining percent** *value*
8. **shape average** {**percent** | *kbps*}
9. **exit**

10. **exit**
11. **interface** *type instance*
12. **service-policy** {**input** | **output**} *policy-map*
13. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class1`	Enters policy map class configuration mode. • Specifies the name of the class whose policy you want to create or change.
Step 4	**random-detect** [*default*] **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# random-detect`	Enables RED with minimum and maximum thresholds.

	Command or Action	Purpose
Step 5	**random-detect dscp** *dscp-value min-threshold max-threshold* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# random-detect dscp af11 1000000 2000000`	Changes the minimum and maximum packet thresholds for the differentiated services code point (DSCP) value. • Enables RED when used on the Cisco XR 12000 Series Router. • *dscp-value*—Number from 0 to 63 that sets the DSCP value. Reserved keywords can be specified instead of numeric values. • *min-threshold*—Minimum threshold in number of bytes. The value range of this argument is from 512 to 1073741823. When the average queue length reaches the minimum threshold, WRED randomly drops some packets with the specified DSCP value. • *max-threshold*—Maximum threshold in number of bytes. The value range of this argument is from the value of the *min-threshold* argument to 1073741823. When the average queue length exceeds the maximum threshold, WRED drops all packets with the specified DSCP value. • The example shows that for packets with DSCP AF11, the WRED minimum threshold is 1,000,000 bytes and maximum threshold is 2,000,000 bytes:
Step 6	**bandwidth** {*bandwidth-kbps* | **percent** *value*} **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# bandwidth percent 30`	(Optional) Specifies the bandwidth allocated for a class belonging to a policy map.
Step 7	**bandwidth remaining percent** *value* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# bandwidth remaining percent 20`	(Optional) Specifies how to allocate leftover bandwidth to various classes.
Step 8	**shape average** {**percent** | *kbps*} **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# shape average percent 50`	(Optional) Shapes traffic to the indicated bit rate according to average rate shaping in kilobits or the interface bandwidth in milliseconds.
Step 9	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Returns the router to policy map configuration mode.
Step 10	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap)# exit`	Returns the router to global configuration mode.

	Command or Action	Purpose
Step 11	**interface** *type instance* **Example:** `RP/0/RP1/CPU0:router(config)# interface POS 0/2/0/0`	Enters configuration mode and configures an interface.
Step 12	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if)# service-policy output policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.

Step 13	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-cmap)# end`
	or
	`RP/0/RP0/CPU0:router(config-cmap)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring Weighted Random Early Detection

WRED drops packets selectively based on IP precedence. Edge routers assign IP precedences to packets as they enter the network. WRED uses these precedences to determine how it treats different types of traffic.

When a packet arrives, the following actions occur:

- The average queue size is calculated.
- If the average is less than the minimum queue threshold, the arriving packet is queued.
- If the average is between the minimum queue threshold for that type of traffic and 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.

Restrictions

You cannot configure WRED on the same interface as priority queueing (PQ).

You can define a class policy to use either tail drop by using the **queue-limit bytes** command or WRED by using the **random-detect** command. When using either tail drop or WRED, you can configure the **bandwidth** command when either the **queue-limit bytes** or **random-detect** command is configured in a class policy. The **bandwidth** command specifies the amount of bandwidth allocated for the class.

When configuring the **random-detect dscp** command, you must configure one of the following commands: **shape average**, **bandwidth**, and **bandwidth remaining**.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **random-detect dscp** *dscp-value min-threshold max-threshold*
5. **random-detect precedence** *precedence-value min-threshold max-threshold*
6. **bandwidth** {*kbps* | **percent** *value*}
7. **bandwidth remaining percent** *value*
8. **shape average** {**percent** | *kbps*}
9. **exit**
10. **interface** *type instance*
11. **service-policy** {**input** | **output**} *policy-map*
12. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class1`	Enters policy map class configuration mode. • Specifies the name of the class whose policy you want to create or change.

	Command or Action	Purpose
Step 4	**random-detect dscp** *dscp-value min-threshold max-threshold* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# random-detect dscp AF11 1000000 2000000`	Changes the minimum and maximum packet thresholds for the DSCP value. • Enables RED when used on the Cisco XR 12000 Series Router. • *dscp-value*—Number from 0 to 63 that sets the DSCP value. Reserved keywords can be specified instead of numeric values. • *min-threshold*—Minimum threshold in number of packets. The value range of this argument is from 512 to 1073741823. When the average queue length reaches the minimum threshold, WRED randomly drops some packets with the specified DSCP value. • *max-threshold*—Maximum threshold in number of packets. The value range of this argument is from the value of the *min-threshold* argument to 1073741823. When the average queue length exceeds the maximum threshold, WRED drops all packets with the specified DSCP value. • This example shows that for packets with DSCP AF11, the WRED minimum threshold is 1,000,000 bytes and maximum threshold is 2,000,000 bytes:
Step 5	**random-detect precedence** *precedence-value min-threshold max-threshold* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# random-detect precedence 3 1000000 2000000`	Configures WRED parameters for a particular IP precedence for a class policy in a policy map. • This example shows that for packets with precedence 3, the WRED minimum threshold is 1,000,000 bytes and maximum threshold is 2000000 bytes.
Step 6	**bandwidth** {*kbps* | **percent** *value*} **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# bandwidth percent 30`	(Optional) Specifies the bandwidth allocated for a class belonging to a policy map. • This example guarantees 30 percent of the interface bandwidth to class class1.
Step 7	**bandwidth remaining percent** *value* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# bandwidth remaining percent 20`	(Optional) Specifies how to allocate leftover bandwidth to various classes. • The remaining bandwidth of 70 percent is shared by all configured classes. • In this example, class class1 receives 20 percent of the 70 percent.
Step 8	**shape average** {**percent** | *kbps*} **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# shape average percent 50`	(Optional) Shapes traffic to the indicated bit rate according to average rate shaping in kilobits or the interface bandwidth in milliseconds.

	Command or Action	Purpose
Step 9	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap)# exit`	Returns the router to global configuration mode.
Step 10	**interface** *type instance* **Example:** `RP/0/RP1/CPU0:router(config)# interface POS 0/2/0/0`	Enters configuration mode and configures an interface.
Step 11	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if)# service-policy output policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.

Step 12	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-cmap)# end`
	or
	`RP/0/RP0/CPU0:router(config-cmap)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Configuring 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 bytes** command is used to define the maximum threshold for a class. When that threshold is reached, enqueued packets to the class queue result in tail drop (packet drop). The maximum threshold for tail drop is 200 milliseconds (ms).

Restrictions

You can define a class policy to use either tail drop by using the **queue-limit bytes** command or WRED by using the **random-detect** command. When using tail drop, you can configure the **bandwidth** command when either the **queue-limit bytes** or **random-detect** command is configured in a class policy. The **bandwidth** command specifies the amount of bandwidth allocated for the class.

SUMMARY STEPS

1. **configure**
2. **policy-map** *policy-name*
3. **class** *class-name*
4. **queue-limit** *value* [**bytes** | **cells** | **ms** | **packets** | **us**]
5. **class** *class-name*
6. **random-detect precedence** *precedence-value min-threshold max-threshold*
7. **bandwidth** {*kbps* | **percent** *value*}
8. **exit**
9. **exit**
10. **interface** *type instance*
11. **service-policy** {**input** | **output**} *policy-map*
12. **end**
    or
    commit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure** **Example:** `RP/0/RP0/CPU0:router# configure`	Enters global configuration mode.
Step 2	**policy-map** *policy-name* **Example:** `RP/0/RP0/CPU0:router(config)# policy-map policy1`	Enters policy map configuration mode. • Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy.
Step 3	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class1`	Enters policy map class configuration mode. • Specifies the name of the class whose policy you want to create or change.

	Command or Action	Purpose
Step 4	**queue-limit** *value* [**bytes** | **cells** | **ms** | **packets** | **us**] **Example:** `RP/0/RP0/CPU0:router(config-pmap)# queue-limit 1000000 bytes`	Specifies or modifies the maximum number of bytes the queue can hold for a class policy configured in a policy map. • In this example, when the queue limit reaches 1,000,000 bytes, enqueued packets to the class queue are dropped.
Step 5	**class** *class-name* **Example:** `RP/0/RP0/CPU0:router(config-pmap)# class class2`	Specifies the name of the class whose policy you want to create or change. • In this example, class2 is configured.
Step 6	**random-detect precedence** *precedence-value min-threshold max-threshold* **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# random-detect precedence 3 1000000 2000000`	Configures WRED parameters for a particular IP precedence for a class policy in a policy map. • In this example, WRED is configured for class2.
Step 7	**bandwidth** {*kbps* | **percent** *value*} **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# bandwidth percent 30`	(Optional) Specifies the bandwidth allocated for a class belonging to a policy map. • This example guarantees 30 percent of the interface bandwidth to class class1.
Step 8	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap-c)# exit`	Returns the router to policy map configuration mode.
Step 9	**exit** **Example:** `RP/0/RP0/CPU0:router(config-pmap)# exit`	Returns the router to global configuration mode.
Step 10	**interface** *type instance* **Example:** `RP/0/RP1/CPU0:router(config)# interface POS 0/2/0/0`	Enters configuration mode, and configures an interface.

	Command or Action	Purpose
Step 11	**service-policy** {**input** | **output**} *policy-map* **Example:** `RP/0/RP0/CPU0:router(config-if)# service-policy output policy1`	Attaches a policy map to an input or output interface to be used as the service policy for that interface. • The traffic policy evaluates all traffic leaving that interface.

Step 12	**end**
	or
	commit
	
	Example:
	`RP/0/RP0/CPU0:router(config-cmap)# end`
	or
	`RP/0/RP0/CPU0:router(config-cmap)# commit`
		Saves configuration changes.
		• When you issue the **end** command, the system prompts you to commit changes:
		`Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]:`
		– Entering **yes** saves configuration changes to the running configuration file, exits the configuration session, and returns the router to EXEC mode.
		– Entering **no** exits the configuration session and returns the router to EXEC mode without committing the configuration changes.
		– Entering **cancel** leaves the router in the current configuration session without exiting or committing the configuration changes.
		• Use the **commit** command to save the configuration changes to the running configuration file and remain within the configuration session.

Where to Go Next

To configure traffic congestion techniques, see the "Configuring Modular QoS Congestion Avoidance on Cisco IOS XR Software" module.

To configure traffic classification techniques, see the "Configuring Modular Quality of Service Packet Classification on Cisco IOS XR Software" module.

To read Cisco IOS XR QoS overview information, see the "Modular Quality of Service Overview on Cisco IOS XR Software" module.

Additional References

The following sections provide references related to implementing QoS congestion avoidance on Cisco IOS XR software.

Related Documents

Related Topic	Document Title
Cisco IOS XR QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS XR Modular Quality of Service Command Reference,* Release 3.3
Packet classification	*Configuring Modular QoS Packet Classification on Cisco IOS XR*, Release 3.3
Traffic shaping, traffic policying, low latency queueing, and MDDR	*Configuring QoS Congestion Management on Cisco IOS XR Software,* Release 3.3
Cisco CRS-1 router getting started material	*Cisco CRS-1 Carrier Routing System Getting Started Guide,* Release 3.3
Information about user groups and task IDs	*Configuring AAA Services on Cisco IOS-XR Software* module of the *Cisco IOS-XR System Security Configuration Guide,* Release 3.3
Cisco IOS XR QoS overview information	*Modular Quality of Service Overview on Cisco IOS XR Software*, Release 3.3

Standards

Standards	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	—

MIBs

MIBs	MIBs Link
There are no applicable MIBs for this module.	To locate and download MIBs for selected platforms using Cisco IOS XR software, use the Cisco MIB Locator found at the following URL: http://cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

RFCs

RFCs	Title
No new or modified RFCs are supported by this feature, and support for existing RFCs has not been modified by this feature.	To locate and download MIBs for Cisco IOS XR releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

Technical Assistance

Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

HC	Cisco IOS XR Interface and Hardware Component Configuration Guide
IC	Cisco IOS XR IP Addresses and Services Configuration Guide
MCC	Cisco IOS XR Multicast Configuration Guide
MPC	Cisco IOS XR MPLS Configuration Guide
QC	Cisco IOS XR Modular Quality of Service Configuration Guide
RC	Cisco IOS XR Routing Configuration Guide
SBC	Cisco IOS XR Session Border Controller Configuration Guide
SC	Cisco IOS XR System Security Configuration Guide
SMC	Cisco IOS XR System Management Configuration Guide

B

C

D

I

L

M

P

Q

R

S

T

W