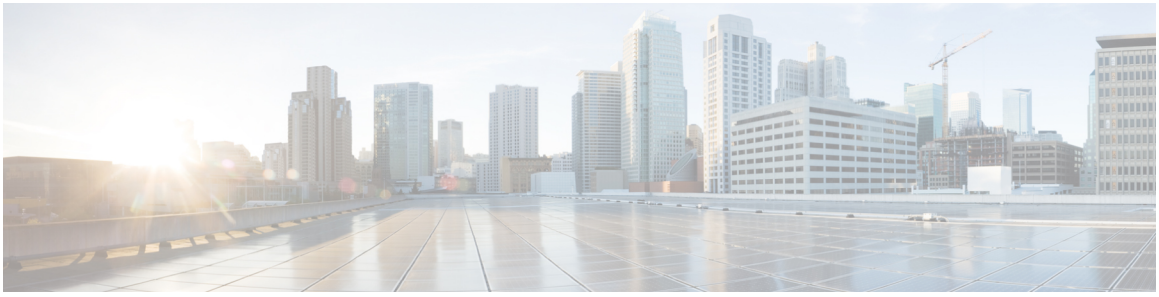
Quality of Service Configuration Guide, Cisco IOS XE 17.x

First Published: 2022-03-11

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



CONTENTS

Preface

This preface describes the audience, organization, and conventions of this document. It also provides information on how to obtain other documentation.

This preface includes the following sections:

Preface

This preface describes the audience, organization, and conventions of this document. It also provides information on how to obtain other documentation.

This preface includes the following sections:

Audience and Scope

This document is designed for the person who is responsible for configuring your Cisco Enterprise router. This document is intended primarily for the following audiences:

- Customers with technical networking background and experience.
- System administrators familiar with the fundamentals of router-based internetworking but who might not be familiar with Cisco IOS software.
- System administrators who are responsible for installing and configuring internetworking equipment, and who are familiar with Cisco IOS software.

Feature Compatibility

For more information about the Cisco IOS XE software, including features available on your device as described in the configuration guides, see the respective router documentation set.

To verify support for specific features, use the Cisco Feature Navigator tool. This tool enables you to determine the Cisco IOS XE software images that support a specific software release, feature set, or a platform.

Document Conventions

This documentation uses the following conventions:

Convention	Description
^ or **Ctrl**	The ^ and **Ctrl** symbols represent the Control key. For example, the key combination **^D** or **Ctrl-D** means hold down the **Control** key while you press the **D** key. Keys are indicated in capital letters but are not case sensitive.
string	A string is a nonquoted set of characters shown in italics. For example, when setting an SNMP community string to public, do not use quotation marks around the string or the string will include the quotation marks.

The command syntax descriptions use the following conventions:

Convention	Description
bold	Bold text indicates commands and keywords that you enter exactly as shown.
italics	Italic text indicates arguments for which you supply values.
[x]	Square brackets enclose an optional element (keyword or argument).
|	A vertical line indicates a choice within an optional or required set of keywords or arguments.
[x | y]	Square brackets enclosing keywords or arguments separated by a vertical line indicate an optional choice.
{x | y}	Braces enclosing keywords or arguments separated by a vertical line indicate a required choice.

Nested sets of square brackets or braces indicate optional or required choices within optional or required elements. For example, see the following table.

Convention	Description
[x {y | z}]	Braces and a vertical line within square brackets indicate a required choice within an optional element.

Examples use the following conventions:

Convention	Description
`screen`	Examples of information displayed on the screen are set in Courier font.
`bold screen`	Examples of text that you must enter are set in Courier bold font.
< >	Angle brackets enclose text that is not printed to the screen, such as passwords.
!	An exclamation point at the beginning of a line indicates a comment line. Exclamation points are also displayed by the Cisco IOS XE software for certain processes.
[ ]	Square brackets enclose default responses to system prompts.

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Note Means *reader take note*. Notes contain helpful suggestions or references to materials that may not be contained in this manual.

Communications, Services, and Additional Information

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

Cisco Bug Search Tool

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

Documentation Feedback

To provide feedback about Cisco technical documentation, use the feedback form available in the right pane of every online document.

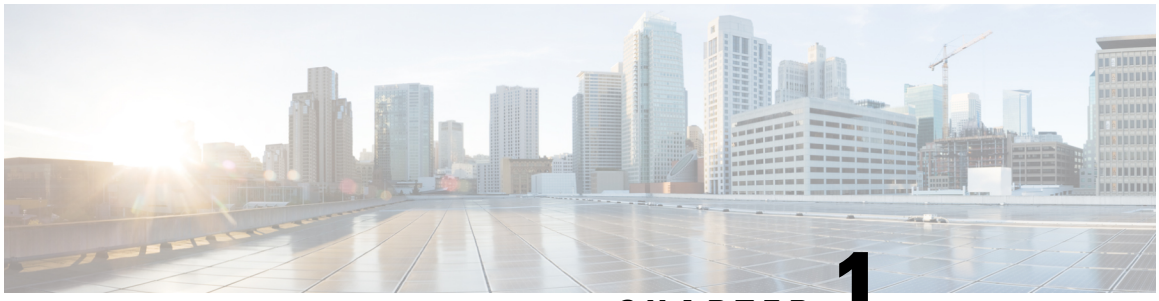
Troubleshooting

For the most up-to-date, detailed troubleshooting information, see the Cisco TAC website at https://www.cisco.com/en/US/support/index.html.

Go to **Products by Category** and choose your product from the list, or enter the name of your product. Look under **Troubleshoot and Alerts** to find information for the issue that you are experiencing.

PART I

Introduction to Policing, Shaping, Marking, and Queuing



CHAPTER 1

QoS Classification, Policing, and Marking on a LAC

The QoS Classification, Policing, and Marking on a LAC feature allows service providers to classify packets based upon the IP type of service (ToS) bits in an embedded IP packet. The classification is used to police the incoming traffic according to the differentiated services code point (DSCP) value. The purpose of classifying the packet by examining its encapsulation is to simplify the implementation and configuration needed for a large number of PPP sessions.

Reference the Chapter Map here

Prerequisites for QoS Classification Policing and Marking on a LAC

Configure the Routers

You must configure the client router, the Layer 2 Tunneling Protocol (L2TP) Access Concentrator (LAC), and the L2TP Network Server (LNS) before applying the QoS policy map as described in the "Configuration Examples for QoS Classification, Policing, and Marking on a LAC" section on page 4.

Verify the State of the Subscriber Service Switch Sessions

You must use the **show sss session** command to verify that the user sessions are enabled on a LAC.

Configure the Interface

You must configure the virtual-template interface before applying the policy map to the session.

Restrictions for QoS Classification, Policing, and Marking on a LAC

- Service-policy on PPP over X.25 (PPPoX) interfaces is not supported.
- Class-based queueing and class-based shaping are not supported.
- Layer 2 marking is not supported.
- The QoS MIB is not supported.
- The **clear counters** command does not clear the counters of the QoS policy map.
- Multihop virtual private dialup networks (VPDNs) are not supported.

Information About QoS Classification Policing and Marking on a LAC

Benefits of the QoS Classification Policing and Marking on a LAC Feature

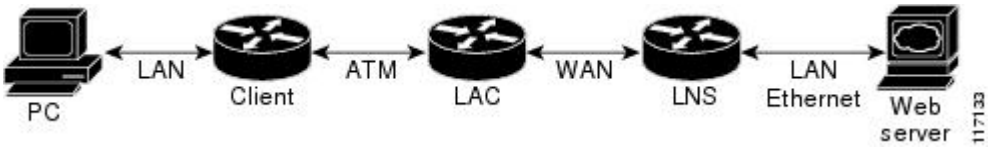
- This feature provides policing and marking on a per-session basis for traffic forwarded into L2TP tunnels to the appropriate LNS and for traffic coming from an L2TP tunnel toward a customer edge router.
- This feature helps recognize the IP ToS value in the Point-to-Point Protocol over Ethernet (PPPoE) encapsulated traffic in order to classify and police the traffic according to the DSCP value.

QoS Policy Maps and a LAC

QoS policing and marking can be achieved by attaching a QoS policy map to the user interface on a LAC in the input and output directions. By using tunnels, input and output service policies can be attached to interfaces. Policy maps get enforced as the packet enters or leaves the tunnel.

The figure below shows the deployment of QoS on PPPoE sessions originating at the client and terminating at the LNS.

Figure 1: Sample Topology for QoS on PPoE Sessions



Note In this sample topology, the LAC is a Cisco 7200 series router.

Upstream Traffic from the LAC to the LNS

Upstream traffic corresponds to packets traversing from the tunnel source to the tunnel destination; in this case, the traffic moves from the LAC to the LNS. The input QoS policy map acts on the upstream traffic before the packet gets encapsulated with the tunnel header.

Downstream Traffic from the LNS to the LAC

Downstream traffic corresponds to packets traversing from the tunnel destination to the tunnel source; in this case, the traffic going from the LNS to the LAC. The output QoS policy map acts on the downstream traffic after the tunnel encapsulation is removed from the packet header.

SSS Sessions on the LAC

The Subscriber Service Switch (SSS) session provides you with the infrastructure to apply QoS features on a per-session basis. The SSS session is preconfigured on the virtual template, and you can use this template to provide QoS classification, policing, and marking.

You can verify the statistics of the upstream and downstream traffic from a QoS policy map in an SSS session by using the **show policy-map session** command.

How to Configure QoS Classification Policing and Marking on a LAC

Enabling the Service Provider to Verify Traffic Statistics

SUMMARY STEPS

1. **enable**
2. **show policy-map session** [**uid** *uid-number*] [**input** | **output** [**class** *class-name*]]
3. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show policy-map session** [**uid** *uid-number*] [**input** | **output** [**class** *class-name*]] **Example:** `Router# show policy-map session uid 401 output`	Displays the information about the session identified by the unique ID.
Step 3	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Configuration Examples for QoS Classification, Policing, and Marking on a LAC

The following examples show you how to apply QoS policy maps to upstream and downstream user session traffic to achieve the required Service Level Agreements (SLAs) provided by the service provider.

Example Configuring the Routers

The following example shows the configuration of the routers before the QoS policy map is verified.

Client Configuration

When you log in to the PC, a PPPoE session is established at the client that faces the LAC. This PPPoE session is forwarded through the L2TP tunnel from the LAC to the LNS at which point the PPPoE session terminates.

To apply QoS sessions to the user traffic that originates from the PC to the web server and to the traffic that originates from the web server to the PC, you should apply a QoS policy map to the user session on the LAC in the input and output directions. The classification will be based on the user traffic that originates at the PC and the web traffic that originates at the web server.

This topology supports bidirectional traffic, meaning that traffic can flow from the PC to the web server and from the web server to the PC.

```
username xyz@cisco.com password 0 password1
username qos4-72a password 0 password1
username qos4-72b password 0 password1
aaa authentication ppp default local
aaa session-id common
ip cef
vpdn enable
!
vpdn-group 1
 request-dialin
  protocol pppoe
!
interface ATM0/0/0
 no ip address
 no ip redirects
```

```
 no ip proxy-arp
 no ip mroute-cache
 load-interval 30
 no atm ilmi-keepalive
!
interface ATM0/0/0.1 point-to-point
 pvc 0/100
  encapsulation aal5snap
  pppoe max-sessions 100
  pppoe-client dial-pool-number 1
 !
 interface Dialer1
  mtu 1492
  ip address negotiated
  encapsulation ppp
  dialer pool 1
  no peer default ip address
  no cdp enable
  ppp authentication chap callin
  ppp chap hostname xyz@cisco.com
  ppp chap password 0 cisco
  ppp ipcp dns request
!
```

LAC Configuration

The following example shows that the interfaces between the client and the LAC are ATM5/0 interfaces.

```
username xyz@cisco.com password 0 password1
username qos4-72a password 0 password1
username qos4-72b password 0 password1
aaa new-model
!
!
aaa authentication ppp default local
aaa session-id common
ip cef
vpdn enable
!
vpdn-group 1
 accept-dialin
  protocol pppoe
  virtual-template 1
!
vpdn-group 2
 request-dialin
  protocol l2tp
  domain cisco.com
 initiate-to ip 10.10.101.2
 local name lac
 no l2tp tunnel authentication
 ip tos reflect
!
interface Serial0/0/0
 bandwidth 2015
 ip address 10.10.100.1 255.255.255.0
 no ip redirects
 no ip proxy-arp
 load-interval 30
 no keepalive
 no cdp enable
!
interface ATM0/0/0
```

```
 no ip address
 no ip redirects
 no ip proxy-arp
 load-interval 30
 no atm ilmi-keepalive
!
interface ATM0/0/0.1 point-to-point
 pvc 0/100
  encapsulation aal5snap
  pppoe max-sessions 100
  protocol ppp Virtual-Template1
  protocol pppoe
!
!
interface Virtual-Template1
 mtu 1492
 no ip address
 no peer default ip address
 ppp authentication chap
!
```

LNS Configuration

The following example shows that the interface between the LAC and the LNS is a Serial3/6 interface.

```
username xyz@cisco.com password 0 password1
username qos4-72b password 0 password1
username qos4-72a password 0 password1
aaa new-model
!
!
aaa authentication ppp default local
aaa session-id common
ip cef
vpdn enable
!
vpdn-group 1
 accept-dialin
  protocol any
  virtual-template 1
 terminate-from hostname lac
 local name lns
 lcp renegotiation always
 no l2tp tunnel authentication
 ip tos reflect
!
interface Serial0/0/0
 bandwidth 2015
 ip address 10.10.100.1 255.255.255.0
 no ip redirects
 no ip proxy-arp
 no ip mroute-cache
 load-interval 30
 no keepalive
 no cdp enable
!
```

Example Verifying the SSS Session

The following example from the **show sss session**command shows that a user session is enabled on the LAC:

```
Router# show sss session
Current SSS Information: Total sessions 1
Uniq ID Type       State         Service      Identifier           Last Chg
401     PPPoE/PPP  connected     Forwarded    xyz@cisco.com        00:02:06
```

Example Applying the QoS Policy Map

The following output shows a QoS policy map to be applied to the user session in the output direction, which is the downstream traffic coming into the PC from the web server. The first subclass of traffic within the session is marked with dscp af11, the second subclass is policed, and the third subclass is dropped.

```
class-map match-any customer1234
 match ip dscp cs1  cs2  cs3  cs4
class-map match-any customer56
 match ip dscp cs5  cs6
class-map match-any customer7
 match ip dscp cs7
policy-map downstream-policy
 class customer1234
  set ip dscp af11
 class customer56
  police cir 20000 bc 10000 pir 40000 be 10000
    conform-action set-dscp-transmit af21
    exceed-action set-dscp-transmit af22
    violate-action set-dscp-transmit af23
 class customer7
   drop
```

Example Configuring the LAC

The following example from the **interface virtual-template** command shows a QoS policy map being applied to the user session on the LAC:

```
Router# configure terminal
Router(config)# interface virtual-template1
Router(config-if)# service-policy output downstream-policy
Router(config-if)# end
```

Example Verifying the QoS Policy Map for Downstream Traffic

In the following example from the **show policy-map session** command, the QoS policy map is applied for traffic in the downstream direction.

Note The session ID, 401, is obtained from the output of the **show sss session** command shown in the "Example Verifying the SSS Session" section on page 7.

```
Router# show policy-map session uid 401 output
SSS session identifier 401 -
  Service-policy output: downstream-policy
    Class-map: customer1234 (match-any)
      4464 packets, 249984 bytes
```

```
    5 minute offered rate 17000 bps, drop rate 0 bps
    Match: ip dscp cs1  cs2  cs3  cs4
      4464 packets, 249984 bytes
      5 minute rate 17000 bps
    QoS Set
      dscp af11
        Packets marked 4464
  Class-map: customer56 (match-any)
    2232 packets, 124992 bytes
    5 minute offered rate 8000 bps, drop rate 0 bps
    Match: ip dscp cs5  cs6
      2232 packets, 124992 bytes
      5 minute rate 8000 bps
    police:
        cir 20000 bps, bc 10000 bytes
        pir 40000 bps, be 10000 bytes
      conformed 2232 packets, 124992 bytes; actions:
        set-dscp-transmit af21
      exceeded 0 packets, 0 bytes; actions:
        set-dscp-transmit af22
      violated 0 packets, 0 bytes; actions:
        set-dscp-transmit af23
      conformed 8000 bps, exceed 0 bps, violate 0 bps
  Class-map: customer7 (match-any)
    1116 packets, 62496 bytes
    5 minute offered rate 4000 bps, drop rate 4000 bps
    Match: ip dscp cs7
      1116 packets, 62496 bytes
      5 minute rate 4000 bps
    drop
  Class-map: class-default (match-any)
    1236 packets, 68272 bytes
    5 minute offered rate 4000 bps, drop rate 0 bps
    Match: any
```

Example Applying the QoS Policy Map to the Session

In the following example, the service provider applies a QoS policy map to the user session in order to limit the amount of bandwidth that the user session is permitted to consume in the upstream direction from the PC to the web server:

```
Router# configure terminal
Router(config)# policy-map upstream-policy
Router(config-pmap)# class class-default
Router(config-pmap-c)# police cir 8000 bc 1500 be 1500 conform-action transmit exceed-action
 drop
Router(config-if)# end
```

This QoS policy map is then applied to the user session as follows:

```
Router# configure terminal
Router(config)# interface virtual-template1
Router(config-if)# service-policy input upstream-policy
Router(config-if)# end
```

Example Verifying the QoS Policy Map for Upstream Traffic

In the following example from the **show policy-map session** command, the QoS policy map is applied for traffic in the upstream direction:

Note The session ID, 401, is obtained from the output of the **show sss session** command in the "Example Verifying the SSS Session" section on page 7.

```
Router# show policy-map session uid 401 input
 SSS session identifier 401 -
  Service-policy input: upstream-policy
    Class-map: class-default (match-any)
      1920 packets, 111264 bytes
      5 minute offered rate 7000 bps, drop rate 5000 bps
      Match: any
      police:
          cir 8000 bps, bc 1500 bytes
        conformed 488 packets, 29452 bytes; actions:
          transmit
        exceeded 1432 packets, 81812 bytes; actions:
          drop
        conformed 7000 bps, exceed 5000 bps
```

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Information about attaching policy maps to interfaces using the modular quality of service (QoS) command-line interface (CLI) (MQC)	"Applying QoS Features Using the MQC" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

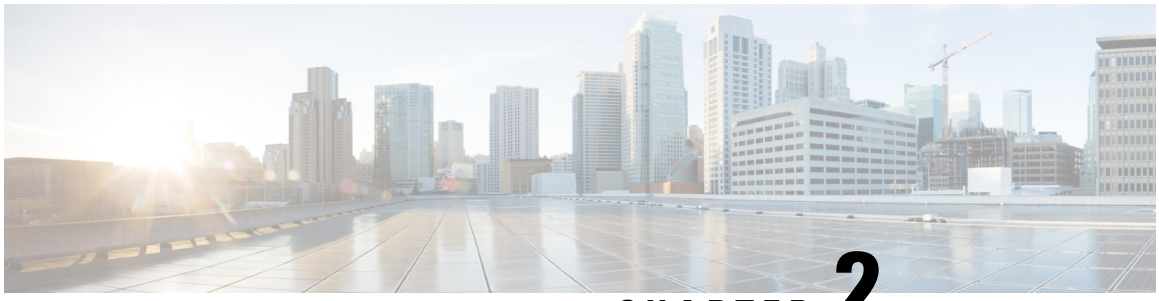
Feature Information for QoS Classification Policing and Marking on a LAC

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 1: Feature Information for QoS: Classification, Policing, and Marking on a LAC

Feature Name	Releases	Feature Information
QoS: Classification, Policing, and Marking on a LAC	Cisco IOS XE Release 2.1	The QoS: Classification, Policing, and Marking on a LAC feature allows service providers to classify packets based upon the IP type of service (ToS) bits in an embedded IP packet. The classification is used to police the incoming traffic according to the differentiated services code point (DSCP) value. The following command was introduced or modified by this feature: **show policy-map session**.



CHAPTER 2

Policing and Shaping Overview

Cisco IOS XE QoS offers two kinds of traffic regulation mechanisms--policing and shaping.

You can deploy these traffic regulation mechanisms (referred to as policers and shapers) throughout your network to ensure that a packet, or data source, adheres to a stipulated contract and to determine the QoS to render the packet. Both policing and shaping mechanisms use the traffic descriptor for a packet--indicated by the classification of the packet--to ensure adherence and service.

Policers and shapers usually identify traffic descriptor violations in an identical manner. They usually differ, however, in the way they respond to violations, for example:

- A policer typically drops traffic, but it can also change the setting or "marking" of a packet. (For example, a policer will either drop the packet or rewrite its IP precedence, resetting the type of service bits in the packet header.)
- A shaper typically delays excess traffic using a buffer, or queueing mechanism, to hold packets and shape the flow when the data rate of the source is higher than expected. (For example, Class-Based Shaping uses a weighted fair queue to delay packets in order 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 misbehaving flows. This activity is sometimes called policing the traffic of the flow.

This chapter gives a brief description of the Cisco IOS XE QoS traffic policing and shaping mechanisms. Because policing and shaping both use the token bucket mechanism, this chapter first explains how a token bucket works. This chapter includes the following sections:

What Is a Token Bucket

A token bucket is a formal definition of a rate of transfer. It has three components: a burst size, a mean rate, and a time interval (Tc). Although the mean rate is generally represented as bits per second, any two values may be derived from the third by the relation shown as follows:

```
mean rate = burst size / time interval
```

Here are some definitions of these terms:

- Mean rate--Also called the committed information rate (CIR), it specifies how much data can be sent or forwarded per unit time on average.
- Burst size--Also called the Committed Burst (Bc) size, it specifies in bits (or bytes) per burst, how much traffic can be sent within a given unit of time to not create scheduling concerns. (For a shaper, such as GTS, it specifies bits per burst; for a policer, such as CAR, it specifies bytes per burst, per second.)
- Time interval--Also called the measurement interval, it specifies the time quantum in seconds per burst.

By definition, over any integral multiple of the interval, the bit rate of the interface will not exceed the mean rate. The bit rate, however, may be arbitrarily fast within the interval.

A token bucket is used to manage a device that regulates the data in a flow. For example, the regulator might be a traffic policer, such as CAR, or a traffic shaper, such as FRTS or GTS. A token bucket itself has no discard or priority policy. Rather, a token bucket discards tokens and leaves to the flow the problem of managing its transmission queue if the flow overdrives the regulator. (Neither CAR nor FRTS and GTS implement either a true token bucket or true leaky bucket.)

In the token bucket metaphor, tokens are put into the bucket at a certain rate. The bucket itself has a specified capacity. If the bucket fills to capacity, newly arriving tokens are discarded. Each token is permission for the source to send a certain number of bits into the network. To send a packet, the regulator must remove from the bucket a number of tokens equal in representation to the packet size.

If not enough tokens are in the bucket to send a packet, the packet either waits until the bucket has enough tokens (in the case of GTS) or the packet is discarded or marked down (in the case of CAR). If the bucket is already full of tokens, incoming tokens overflow and are not available to future packets. Thus, at any time, the largest burst a source can send into the network is roughly proportional to the size of the bucket.

Note that the token bucket mechanism used for traffic shaping has both a token bucket and a data buffer, or queue; if it did not have a data buffer, it would be a policer. For traffic shaping, packets that arrive that cannot be sent immediately are delayed in the data buffer.

For traffic shaping, a token bucket permits burstiness but bounds it. It guarantees that the burstiness is bounded so that the flow will never send faster than the token bucket's capacity, divided by the time interval, plus the established rate at which tokens are placed in the token bucket. See the following formula:

```
(token bucket capacity in bits / time interval in seconds) + established rate in bps =
maximum flow speed in bps
```

This method of bounding burstiness also guarantees that the long-term transmission rate will not exceed the established rate at which tokens are placed in the bucket.

Traffic Policing

Traffic policing allows you to control the maximum rate of traffic sent or received on an interface and to partition a network into multiple priority levels or class of service (CoS).

Traffic policing manages the maximum rate of traffic through a token bucket algorithm. The token bucket algorithm 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 when several large packets are sent in the same traffic stream.

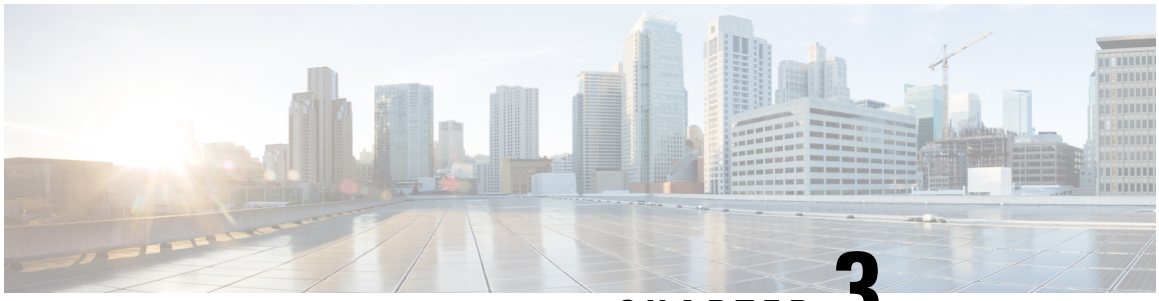
The token bucket algorithm provides users with three actions for each packet: a conform action, an exceed action, and an optional violate action. Traffic that is entering the interface with Traffic Policing configured

is placed in to one of these categories. Within these three categories, users can decide packet treatments. For instance, packets that conform can be configured to be transmitted, 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 that is entering or leaving the network. In the most common traffic policing configurations, traffic that conforms is transmitted and traffic that exceeds is sent with a decreased priority or is dropped. Users can change these configuration options to suit their network needs.

Traffic Shaping to Regulate Packet Flow

Regulating the packet flow (that is, the flow of traffic) on the network is also known as traffic shaping. Traffic shaping allows you to control the speed of traffic that is leaving an interface. This way, you can match the flow of the traffic to the speed of the interface receiving the packet.



CHAPTER 3

IPv6 QoS: MQC Traffic Shaping

Traffic shaping allows you to limit the packet dequeue rate by holding additional packets in the queues and forwarding them as specified by parameters configured for traffic shaping features

Information About IPv6 QoS: MQC Traffic Shaping

Implementation Strategy for QoS for IPv6

IPv6 packets are forwarded by paths that are different from those for IPv4. QoS features supported for IPv6 environments include packet classification, queueing, traffic shaping, weighted random early detection (WRED), class-based packet marking, and policing of IPv6 packets. These features are available at both the process switching and Cisco Express Forwarding switching paths of IPv6.

All of the QoS features available for IPv6 environments are managed from the modular QoS command-line interface (MQC). The MQC allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach those traffic policies to interfaces.

To implement QoS in networks running IPv6, follow the same steps that you would follow to implement QoS in networks running only IPv4. At a very high level, the basic steps for implementing QoS are as follows:

- Know which applications in your network need QoS.
- Understand the characteristics of the applications so that you can make decisions about which QoS features would be appropriate.
- Know your network topology so that you know how link layer header sizes are affected by changes and forwarding.
- Create classes based on the criteria you establish for your network. In particular, if the same network is also carrying IPv4 traffic along with IPv6, decide if you want to treat both of them the same way or treat them separately and specify match criteria accordingly. If you want to treat them the same, use match statements such as **match precedence**, **match dscp**. If you want to treat them separately, add match criteria such as **match protocol ip** and **match protocol ipv6** in a match-all class map.
- Create a policy to mark each class.

- Work from the edge toward the core in applying QoS features.
- Build the policy to treat the traffic.
- Apply the policy.

Traffic Policing in IPv6 Environments

Congestion management for IPv6 is similar to IPv4, and the commands used to configure queueing and traffic shaping features for IPv6 environments are the same commands as those used for IPv4. Traffic shaping allows you to limit the packet dequeue rate by holding additional packets in the queues and forwarding them as specified by parameters configured for traffic shaping features. Traffic shaping uses flow-based queueing by default. CBWFQ can be used to classify and prioritize the packets. Class-based policer and generic traffic shaping (GTS) or Frame Relay traffic shaping (FRTS) can be used for conditioning and policing traffic.

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS commands	*Cisco IOS Master Commands List, All Releases*
IPv6 commands	*Cisco IOS IPv6 Command Reference*
Cisco IOS IPv6 features	*Cisco IOS IPv6 Feature Mapping*

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	*IPv6 RFCs*

MIBs

MIB	MIBs Link
	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

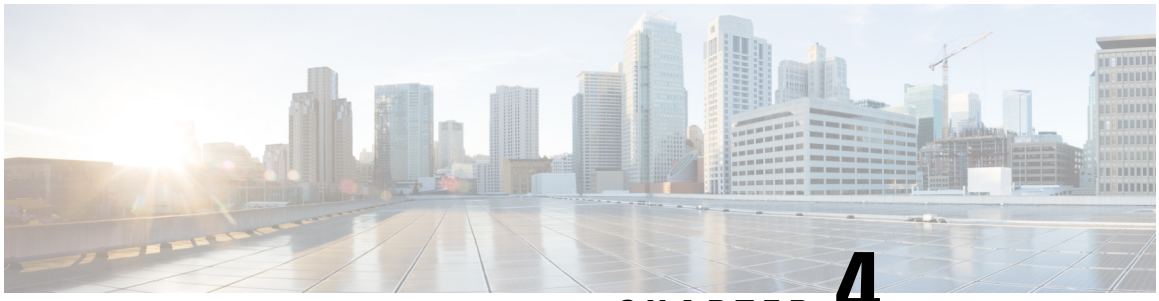
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for IPv6 QoS: MQC Traffic Shaping

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 2: Feature Information for IPv6 QoS: MQC Traffic Shaping



CHAPTER 4

Distribution of Remaining Bandwidth Using Ratio

The Distribution of Remaining Bandwidth Using Ratio feature allows service providers to configure a bandwidth-remaining ratio on subinterfaces and class queues. This ratio specifies the relative weight of a subinterface or queue with respect to other subinterfaces or queues. During congestion, the router uses this bandwidth-remaining ratio to determine the amount of excess bandwidth (unused by priority traffic) to allocate to a class of nonpriority traffic. The router allocates excess bandwidth relative to the other subinterface-level queues and class queues configured on the physical interface. By administration of a bandwidth-remaining ratio, traffic priority is not based solely on speed. Instead, the service provider can base priority on alternative factors such as service product and subscription rate.

Prerequisites for Distribution of Remaining Bandwidth Using Ratio

Before enabling the Distribution of Remaining Bandwidth Using Ratio feature, create as many traffic classes as you need by using the class-map command.

Restrictions for Distribution of Remaining Bandwidth Using Ratio

- Bandwidth-remaining ratios can be used on outbound interfaces only.
- The bandwidth remaining ratio command cannot coexist with another bandwidth command in different traffic classes of the same policy map. For example, the following configuration is not valid and causes an error message to display:

```
policy-map Prec1
 class precedence_0
  bandwidth remaining ratio 10
 class precedence_2
  bandwidth 1000
```

- The bandwidth remaining ratio command cannot coexist with another bandwidth command in the same class. For example, the following configuration is not valid and causes an error message to display:

```
policy-map Prec1
 class precedence_0
  bandwidth 1000
  bandwidth remaining ratio 10
```

- The bandwidth remaining ratio command cannot coexist with the priority command in the same class. For example, the following configuration is not valid and causes an error message to display:

```
policy-map Prec1
 class precedence_1
  priority percent 10
  bandwidth remaining ratio 10
```

Information About Distribution of Remaining Bandwidth Using Ratio

Benefits of the Distribution of Remaining Bandwidth Using Ratio Feature

The Distribution of Remaining Bandwidth Using Ratio feature allows service providers to prioritize subscriber traffic during periods of congestion. A bandwidth-remaining ratio is used to influence how the router allocates excess bandwidth (unused by priority traffic) to a class of nonpriority traffic. Instead of using only bandwidth rate, the router considers configured minimum bandwidth rates, maximum bandwidth rates, and bandwidth-remaining ratios when determining excess bandwidth allocation. A bandwidth-remaining ratio adds more flexibility in prioritizing traffic and enables you to influence excess bandwidth allocation by basing the bandwidth-remaining ratio on factors other than speed.

With bandwidth-remaining ratios, service providers have more flexibility in assigning priority to subinterfaces and queues during congestion. In addition to speed, you can base the bandwidth-remaining ratio on alternative factors, such as a service product or subscription rate. In this way, for example, you can give higher weight to subinterfaces that carry business services and lower weight to subinterfaces that carry residential services.

Bandwidth-Remaining Ratio Functionality

A bandwidth-remaining ratio, specified by the **bandwidth remaining ratio** command, is a value from 1 to 1000 that is used to determine the amount of unused (excess) bandwidth to allocate to a class-level queue or subinterface-level queue during congestion. The router allocates the excess bandwidth relative to the other class-level queues and subinterface-level queues configured on the physical interface. The bandwidth-remaining ratio value does not indicate a percentage. As the name implies, a ratio is used. For example, a subinterface with a bandwidth-remaining ratio of 100 receives 10 times the unused (excess) bandwidth during congestion than a subinterface with a bandwidth-remaining ratio of 10.

Without bandwidth-remaining ratios, the queueing mechanism or scheduler on the router allocates unused (excess) bandwidth equally among the classes or subinterfaces.

With bandwidth-remaining ratios, unused (excess) bandwidth allocation can be based on factors other than the bandwidth rate (for example, the service product or the subscription rate).

Using the bandwidth remaining ratio command, the bandwidth-remaining ratio can be configured differently on each subinterface or class. The bandwidth-remaining ratio can range from 1 to 1000. For example, if there are three subscribers, and the bandwidth-remaining ratios are configured as 9, 7, and 1, and if after priority traffic is served, there are 1700 kbps of excess bandwidth, the subscribers get 900 kbps, 700 kbps, and 100 kbps, respectively.

Note The bandwidth remaining ratio can range from 1-1000 on per interface basis and the sum of all interface ratios cannot exceed 1000.

How to Configure Distribution of Remaining Bandwidth Using Ratio

You can apply bandwidth-remaining ratios to subinterfaces and/or classes queues.

Configuring and Applying Bandwidth-Remaining Ratios to Subinterfaces

Note You can apply bandwidth-remaining ratios to outbound subinterfaces only.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *child-policy-name*
4. **class** *class-map-name*
5. **bandwidth** *bandwidth-kbps*
6. Repeat steps 4 and 5 to configure the additional traffic classes, if needed.
7. **exit**
8. **exit**
9. **policy-map** *parent-policy-name*
10. **class class-default**
11. **bandwidth remaining ratio** *ratio*
12. **shape** {**average** | **peak**} *cir* [*bc*] [*be*]
13. **service-policy** *child-policy-name*
14. **exit**
15. **exit**
16. **interface** *type slot* / *module* / *port* **.** *subinterface* [**point-to-point** | **multipoint**]

17. **service-policy output** *parent-policy-name*
18. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *child-policy-name* **Example:** `Router(config)# policy-map Child`	Creates or modifies a child policy map and enters policy-map configuration mode. • Enter the name of the child policy map.
Step 4	**class** *class-map-name* **Example:** `Router(config-pmap)# class precedence_0`	Configures the class map and enters policy-map class configuration mode.
Step 5	**bandwidth** *bandwidth-kbps* **Example:** `Router(config-pmap-c)# bandwidth 10000`	Specifies the bandwidth, in kbps, to be allocated to this traffic class. • Enter the amount of bandwidth, in kilobits per second (kbps).
Step 6	Repeat steps 4 and 5 to configure the additional traffic classes, if needed.	
Step 7	**exit** **Example:** `Router(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 8	**exit** **Example:** `Router(config-pmap)# exit`	Exits policy-map configuration mode.
Step 9	**policy-map** *parent-policy-name* **Example:** `Router(config)# policy-map Parent`	Creates or modifies a parent policy map and enters policy-map configuration mode. • Enter the name of the parent policy map.

	Command or Action	Purpose
Step 10	**class class-default** **Example:** `Router(config-pmap)# class class-default`	Configures the class-default class and enters policy-map class configuration mode. **Note** The router interprets any features that are configured under the class-default class as aggregate features on the subinterface.
Step 11	**bandwidth remaining ratio** *ratio* **Example:** `Router(config-pmap-c)# bandwidth remaining ratio 10`	Specifies the bandwidth-remaining ratio for the subinterface. • Enter the ratio. The ratio is the value used to determine the amount of unused bandwidth to allocate to each queue on the subinterface during periods of congestion. The scheduler allocates the excess bandwidth relative to other subinterfaces. Valid values are 1 to 1000. The default value is 1.
Step 12	**shape** {**average** | **peak**} *cir* [*bc*] [*be*] **Example:** `Router(config-pmap-c)# shape average 100000000`	(Optional) Shapes the average or peak rate to the rate that you specify. Enter either the **average** or **peak** keyword along with the CIR and any optional arguments. Note the following: • average--Specifies average-rate shaping. • peak--Specifies peak-rate shaping. • cir--Specifies the committed information rate (CIR), in bits per second (bps). • (Optional) bc--Specifies the committed burst size, in bits. • (Optional) be--Specifies the excess burst size, in bits.
Step 13	**service-policy** *child-policy-name* **Example:** `Router(config-pmap-c)# service-policy Child`	Applies the child policy map that you specify to the traffic class. • Enter the name of the previously configured child policy map. The router applies the QoS actions (features) specified in the child policy map to the traffic class. **Note** The **service-policy** command typically requires that you specify the direction of the traffic using the input or output keywords. However, when applying a child policy to a parent policy, do not specify a traffic direction.

	Command or Action	Purpose
Step 14	**exit** **Example:** `Router(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 15	**exit** **Example:** `Router(config-pmap)# exit`	Exits policy-map configuration mode.
Step 16	**interface** *type slot* **/** *module* **/** *port* **.** *subinterface* [**point-to-point** | **multipoint**] **Example:** `Router(config)# interface GigabitEthernet 1/0/0.1`	Creates or modifies the interface that you specify and enters subinterface configuration mode. Enter the interface type. Note the following: • type--Specifies the interface type (for example, Gigabit Ethernet). • slot/module/port.subinterface--Specifies the number of the subinterface that identifies the subinterface (for example, 1/0/0.1). • (Optional) point-to-point--Indicates that the subinterface is a point-to-point subinterface. • (Optional) multipoint--Indicates that the subinterface is a point-to-multipoint subinterface.
Step 17	**service-policy output** *parent-policy-name* **Example:** `Router(config-subif)# service-policy output Parent`	Applies the parent policy map to the subinterface. • Enter the **output** keyword and the name of the parent policy map. **Note** The router shapes the subinterface traffic to the shaping rate specified in the parent class-default class and applies the QoS actions (features) specified in the child policy map. **Note** During periods of congestion, the router uses the bandwidth-remaining ratio specified in the parent policy map to allocate unused bandwidth on this subinterface relative to other subinterfaces.
Step 18	**end** **Example:** `Router(config-subif)# end`	Returns to privileged EXEC mode.

Configuring and Applying Bandwidth-Remaining Ratios to Class Queues

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *child-policy-name*
4. **class** *class-map-name*
5. **shape** {**average** | **peak**} *cir* [*bc*] [*be*]
6. **bandwidth remaining ratio** *ratio*
7. Repeat steps 4, 5 and 6 for each class queue that you want to define, specifying the bandwidth-remaining ratio as applicable.
8. **exit**
9. **exit**
10. **policy-map** *parent-policy-name*
11. **class class-default**
12. **shape** {**average** | **peak**} *cir* [*bc*] [*be*]
13. **bandwidth remaining ratio** *ratio*
14. **service-policy** *child-policy-name*
15. **exit**
16. **exit**
17. **interface** *type slot* / *module* / *port* **.** *subinterface* [**point-to-point** | **multipoint**]
18. **service-policy output** *parent-policy-name*
19. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *child-policy-name* **Example:** `Router(config)# policy-map Child`	Creates or modifies a child policy map and enters policy-map configuration mode. • Enter the name of the child policy map.
Step 4	**class** *class-map-name* **Example:**	Configures the class map and enters policy-map class configuration mode.

	Command or Action	Purpose
	`Router(config-pmap)# class precedence_0`	

Step 5	**shape** {**average** | **peak**} *cir* [*bc*] [*be*]
	Example:
	`Router(config-pmap-c)# shape average 100000000`
		(Optional) Shapes the average or peak rate to the rate that you specify.
		• Enter either the **average** or **peak** keyword along with the CIR and any optional arguments. Note the following:
		  • average--Specifies average-rate shaping.
		  • peak--Specifies peak-rate shaping.
		  • cir--Specifies the committed information rate (CIR), in bits per second (bps).
		  • (Optional) bc--Specifies the committed burst size, in bits.
		  • (Optional) be--Specifies the excess burst size, in bits.

Step 6	**bandwidth remaining ratio** *ratio* **Example:** `Router(config-pmap-c)# bandwidth remaining ratio 10`	Specifies the bandwidth-remaining ratio for the traffic class. • Enter the bandwidth-remaining ratio. The ratio is the value used to determine the amount of unused bandwidth to allocate to each queue on the subinterface during periods of congestion. The queueing mechanism or scheduler allocates the excess bandwidth relative to other subinterfaces. Valid values are 1 to 1000. The default value is 1. **Note** In a hierarchical policy map structure, the **bandwidth remaining ratio***ratio* command must be used for at least one class. Using it in other classes is optional. When this command is not explicitly enabled in the other classes, the queueing mechanism uses 1 as the default.
Step 7	Repeat steps 4, 5 and 6 for each class queue that you want to define, specifying the bandwidth-remaining ratio as applicable.	
Step 8	**exit** **Example:** `Router(config-pmap-c)# exit`	Exits policy-map class configuration mode.

	Command or Action	Purpose
Step 9	**exit** **Example:** `Router(config-pmap)# exit`	Exits policy-map configuration mode.
Step 10	**policy-map** *parent-policy-name* **Example:** `Router(config)# policy-map Parent`	Creates or modifies a parent policy map and enters policy-map configuration mode. • Enter the name of the parent policy map.
Step 11	**class class-default** **Example:** `Router(config-pmap)# class class-default`	Configures the class-default class and enters policy-map class configuration mode. **Note** The router interprets any features that are configured under the class-default class as aggregate features on the subinterface.
Step 12	**shape** {**average** | **peak**} *cir* [*bc*] [*be*] **Example:** `Router(config-pmap-c)# shape average 100000000`	(Optional) Shapes the average or peak rate to the rate that you specify. • Enter either the **average** or **peak** keyword along with the CIR and any optional arguments. Note the following: • average--Specifies average-rate shaping. • peak--Specifies peak-rate shaping. • cir--Specifies the committed information rate (CIR), in bits per second (bps). • (Optional) bc--Specifies the committed burst size, in bits. • (Optional) be--Specifies the excess burst size, in bits.
Step 13	**bandwidth remaining ratio** *ratio* **Example:** `Router(config-pmap-c)# bandwidth remaining ratio 10`	(Optional for class-default or other classes in a hierarchical policy map structure) Specifies the bandwidth-remaining ratio for the subinterface. • Enter the bandwidth-remaining ratio. The ratio is the value used to determine the amount of unused bandwidth to allocate to each queue on the subinterface during periods of congestion. The queueing mechanism or scheduler allocates the excess bandwidth relative to other subinterfaces. Valid values are 1 to 1000. The default value is 1.

	Command or Action	Purpose
		Note In a hierarchical policy map structure, the **bandwidth remaining ratio***ratio* command must be used for at least one class. Using it in other classes is optional. When this command is not explicitly enabled in the other classes, the queueing mechanism uses 1 as the default.
Step 14	**service-policy** *child-policy-name* **Example:** `Router(config-pmap-c)# service-policy Child`	Applies the child policy map that you specify to the traffic class. • Enter the name of the child policy map. The router applies the QoS actions (features) specified in the child policy map to the traffic class. **Note** The **service-policy**command typically requires that you specify the direction of the traffic using the input or output keywords. However, when applying a child policy map to a parent policy map, do not specify traffic direction.
Step 15	**exit** **Example:** `Router(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 16	**exit** **Example:** `Router(config-pmap)# exit`	Exits policy-map configuration mode.
Step 17	**interface** *type slot / module / port . subinterface* [**point-to-point** | **multipoint**] **Example:** `Router(config)# interface GigabitEthernet 1/0/0.1`	Creates or modifies the interface that you specify and enters subinterface configuration mode. • Enter the interface type. Note the following: • type--Specifies the interface type (for example, Gigabit Ethernet). • slot/module/port.subinterface--Specifies the number of the subinterface that identifies the subinterface (for example, 1/0/0.1). • (Optional) point-to-point--Indicates that the subinterface is a point-to-point subinterface. • (Optional) multipoint--Indicates that the subinterface is a point-to-multipoint subinterface.
Step 18	**service-policy output** *parent-policy-name*	Attaches the parent policy map to the subinterface.

	Command or Action	Purpose
	Example: `Router(config-subif)# service-policy output Parent`	• Enter the **output**keyword and the name of the parent policy map. **Note** When congestion occurs, the class queues receive bandwidth according to the specified class-level bandwidth-remaining ratios.
Step 19	**end** **Example:** `Router(config-subif)# end`	Returns to privileged EXEC mode.

Configuration Examples for Distribution of Remaining Bandwidth Using Ratio

Example Configuring Bandwidth-Remaining Ratios on Ethernet Subinterfaces

The following example shows how to configure bandwidth-remaining ratios on an Ethernet subinterface using a hierarchical policy. In the example, Gigabit Ethernet subinterface 1/0/0.1 is shaped to 100 Mbps. During congestion, the router uses the bandwidth-remaining ratio of 10 to determine the amount of excess bandwidth (unused by priority traffic) to allocate to the nonpriority traffic on subinterface 1/0/0.1, relative to the other subinterface-level and class-level queues on the interface.

```
policy-map Child
 class precedence_0
  bandwidth 10000
 class precedence_1
  shape average 100000
  bandwidth 100
policy-map Parent
 class class-default
  bandwidth remaining ratio 10
  shape average 100000000
  service-policy Child
interface GigabitEthernet1/0/0.1
 encapsulation dot1Q 100
 ip address 10.1.0.1 255.255.255.0
 service-policy output Parent
```

Example Verifying Bandwidth-Remaining Ratios on Class Queues

In the following sample configuration, vlan10_policy is applied on the Gigabit Ethernet subinterface 1/0/0.10 and vlan20_policy is applied on the Gigabit Ethernet subinterface 1/0/0.20. During congestion on the interface, subinterface Gigabit Ethernet 1/0/0.20 has 10 times more available bandwidth than subinterface Gigabit Ethernet 1/0/0.10 because the bandwidth-remaining ratio for subinterface Gigabit Ethernet 1/0/0.20 is 10 times more than the bandwidth-remaining ratio for subinterface 1/0/0.10: 100 on subinterface 1/0/0.20 and 10 on subinterface 1/0/0.10.

When congestion occurs within a subinterface level, the class queues receive bandwidth according to the class-level bandwidth-remaining ratios. In the example, the bandwidth for classes precedence_0, precedence_1, and precedence_2 is allocated based on the bandwidth-remaining ratios of the classes: 20, 40, and 60, respectively.

Router# **show policy-map**

```
Policy Map child-policy
    Class precedence_0
      Average Rate Traffic Shaping
      cir 500000 (bps)
      bandwidth remaining ratio 20 <---- Class-level ratio
    Class precedence_1
      Average Rate Traffic Shaping
      cir 500000 (bps)
      bandwidth remaining ratio 40 <---- Class-level ratio
    Class precedence_2
      Average Rate Traffic Shaping
      cir 500000 (bps)
      bandwidth remaining ratio 60 <---- Class-level ratio
Policy Map vlan10_policy
    Class class-default
      Average Rate Traffic Shaping
      cir 1000000 (bps)
      bandwidth remaining ratio 10 <---- Subinterface-level ratio
      service-policy child-policy
Policy Map vlan20_policy
    Class class-default
      Average Rate Traffic Shaping
      cir 1000000 (bps)
      bandwidth remaining ratio 100 <---- Subinterface-level ratio
      service-policy child-policy
interface GigabitEthernet1/0/0.10
 encapsulation dot1Q 10
 snmp trap link-status
 service-policy output vlan10_policy
interface GigabitEthernet1/0/0.20
 encapsulation dot1Q 20
 snmp trap link-status
 service-policy output vlan20_policy
```

Example: Verifying Bandwidth Remaining Ratios

The following sample output from the show policy-map interface command indicates that bandwidth-remaining ratios are configured on class-level queues in the policy maps named vlan10_policy and child-policy, which are attached to Gigabit Ethernet subinterface 1/0/0.10.

```
Router# show policy-map interface GigabitEthernet 1/0/0.10
GigabitEthernet1/0/0.10
  Service-policy output: vlan10_policy
    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
      shape (average) cir 1000000, bc 4000, be 4000
      target shape rate 1000000
```

```
      bandwidth remaining ratio 10
      Service-policy : child-policy
        Class-map: precedence_0 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: ip precedence 0
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          shape (average) cir 500000, bc 2000, be 2000
          target shape rate 500000
          bandwidth remaining ratio 20
        Class-map: precedence_1 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: ip precedence 1
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          shape (average) cir 500000, bc 2000, be 2000
          target shape rate 500000
          bandwidth remaining ratio 40
        Class-map: precedence_2 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: ip precedence 2
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          shape (average) cir 500000, bc 2000, be 2000
          target shape rate 500000
          bandwidth remaining ratio 60
        Class-map: class-default (match-any)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any

          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
```

The following sample output from the show policy-map interface command indicates that bandwidth-remaining ratios are configured on class-level queues in the policy maps named vlan20_policy and child-policy, which are attached to Gigabit Ethernet subinterface 1/0/0.20.

```
Router# show policy-map interface GigabitEthernet 1/0/0.20
GigabitEthernet1/0/0.20
  Service-policy output: vlan20_policy
    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
      shape (average) cir 1000000, bc 4000, be 4000
      target shape rate 1000000
      bandwidth remaining ratio 100
      Service-policy : child-policy
```

```
        Class-map: precedence_0 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: ip precedence 0
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          shape (average) cir 500000, bc 2000, be 2000
          target shape rate 500000
          bandwidth remaining ratio 20
        Class-map: precedence_1 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: ip precedence 1
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          shape (average) cir 500000, bc 2000, be 2000
          target shape rate 500000
          bandwidth remaining ratio 40
        Class-map: precedence_2 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: ip precedence 2
          Queueing
          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          shape (average) cir 500000, bc 2000, be 2000
          target shape rate 500000
          bandwidth remaining ratio 60
        Class-map: class-default (match-any)
          0 packets, 0 bytes
          5 minute offered rate 0 bps, drop rate 0 bps
          Match: any

          queue limit 64 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
```

The following sample output from the show policy-map command indicates that a bandwidth-remaining ratio of 10 is configured on the parent class-default class of the policy map named vlan10_policy.

```
Router# show policy-map vlan10_policy
  Policy Map vlan10_policy
    Class class-default
      Average Rate Traffic Shaping
      cir 1000000 (bps)
      bandwidth remaining ratio 10
      service-policy child-policy
```

The following sample output from the show policy-map command indicates that a bandwidth-remaining ratio of 100 is configured on the parent class-default class of the policy map named vlan20_policy.

```
Router# show policy-map vlan20_policy
  Policy Map vlan20_policy
    Class class-default
      Average Rate Traffic Shaping
      cir 1000000 (bps)
      bandwidth remaining ratio 100
      service-policy child-policy
```

The following sample output from the show policy-map command indicates that bandwidth-remaining ratios of 20, 40, and 60 are configured on the class queues precedence_0, precedence_1, and precedence_2, respectively.

```
Router# show policy-map child-policy
  Policy Map child-policy
    Class precedence_0
      Average Rate Traffic Shaping
      cir 500000 (bps)
      bandwidth remaining ratio 20
    Class precedence_1
      Average Rate Traffic Shaping
      cir 500000 (bps)
      bandwidth remaining ratio 40
    Class precedence_2
      Average Rate Traffic Shaping
      cir 500000 (bps)
      bandwidth remaining ratio 60
```

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Congestion avoidance	"Congestion Avoidance Overview" module
Class maps, policy maps, hierarchical policy maps, Modular Quality of Service Command-Line Interface (CLI) (MQC)	"Applying QoS Features Using the MQC" module
Traffic shaping, traffic policing	"Policing and Shaping Overview" module

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	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

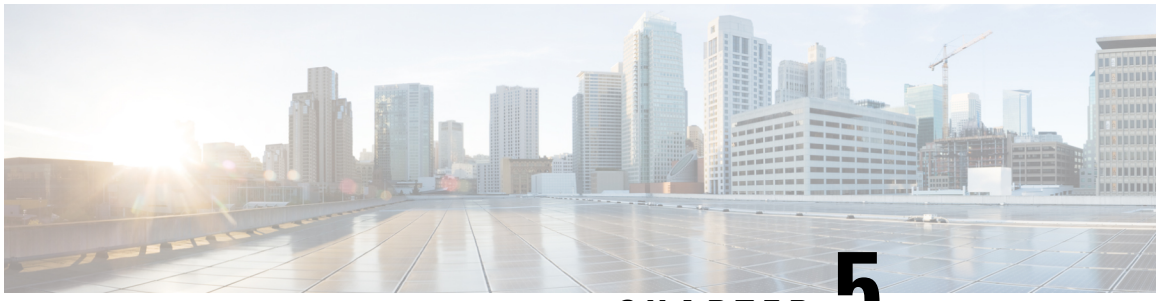
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Distribution of Remaining Bandwidth Using Ratio

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 3: Feature Information for Distribution of Remaining Bandwidth Using Ratio



CHAPTER 5

QoS Percentage-Based Shaping

The QoS: Percentage-Based Shaping feature allows you to configure traffic shaping on the basis of a percentage of bandwidth available on the interface. This feature also allows you to specify the committed (conform) burst (bc) size and the excess (peak) burst (be) size (used for configuring traffic shaping) in milliseconds (ms). Configuring traffic shaping in this manner enables you to use the same policy map for multiple interfaces with differing amounts of bandwidth.

Information About QoS Percentage-Based Shaping

Benefits for QoS Percentage-Based Shaping

This feature provides the ability to configure traffic shaping on the basis of a percentage of bandwidth available on an interface, and it allows you to specify burst sizes in milliseconds. Configuring traffic shaping in this manner enables you to use the same policy map for multiple interfaces with differing amounts of bandwidth. That is, you do not have to recalculate the bandwidth for each interface or configure a different policy map for each type of interface.

Class and Policy Maps for QoS Percentage-Based Shaping

To configure the QoS: Percentage-Based Shaping feature, you must define a traffic class, configure a policy map, and then attach that policy map to the appropriate interface.

In the MQC, the **class-map** command is used to define a traffic class (which is then associated with a traffic policy). The purpose of a traffic class is to classify traffic.

The MQC consists of the following three processes:

- Defining a traffic class with the **class-map** command.
- Creating a traffic policy by associating the traffic class with one or more QoS features (using the **policy-map** command).

- Attaching the traffic policy to the interface with the **service-policy** command.

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 (that is, match-all or match-any). The traffic class is named in the **class-map** command line; for example, if you enter the **class-map cisco** command while configuring the traffic class in the CLI, 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.

Traffic Regulation Mechanisms and Bandwidth Percentages

Cisco IOS XE quality of service (QoS) offers two kinds of traffic regulation mechanisms--traffic policing and traffic shaping. A traffic policer typically drops traffic that violates a specific rate. A traffic shaper typically delays excess traffic using a buffer to hold packets and shapes the flow when the data rate to a queue is higher than expected.

Traffic shaping and traffic policing can work in tandem and can be configured in a class map. Class maps organize data packets into specific categories ("classes") that can, in turn, receive a user-defined QoS treatment when used in policy maps (sometimes referred to as "service policies").

Before this feature, traffic policing and traffic shaping were configured on the basis of a user-specified amount of bandwidth available on the interface. Policy maps were then configured on the basis of that specific amount of bandwidth, meaning that separate policy maps were required for each interface.

This feature provides the ability to configure traffic policing and traffic shaping on the basis of a percentage of bandwidth available on the interface. Configuring traffic policing and traffic shaping in this manner enables customers to use the same policy map for multiple interfaces with differing amounts of bandwidth.

Configuring traffic policing and shaping on the basis of a percentage of bandwidth is accomplished by using the **police** (percent) and **shape** (percent) commands.

Burst Size Specified in Milliseconds Option

The purpose of the burst parameters (bc and be) is to specify the amount of traffic to anticipate under normal operating conditions before traffic is dropped or delayed. Setting sufficiently high burst values helps to ensure good throughput.

This feature allows you the option of specifying the committed (conform) burst (bc) size and the excess (peak) burst (be) as milliseconds (ms) of the class bandwidth when you configure traffic shaping. The number of milliseconds is used to calculate the number of bytes to be used by the QoS: Percentage-Based Shaping feature.

Specifying these burst sizes in milliseconds is accomplished by using the **bc** and **be** keywords (and their associated arguments) of the **shape** (percent) command.

How to Configure QoS Percentage-Based Shaping

Configuring a Class and Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-name*
4. **class** {*class-name*| **class-default**}
5. **shape** {**average** | **peak**} **percent** *percentage* [**be** *excess-burst-in-msec* **ms**] [**bc** *committed-burst-in-msec* **ms**]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-name* **Example:** `Router(config)# policy-map policy1`	Specifies the name of the policy map to be created. Enters policy-map configuration mode. • Enter the policy map name.
Step 4	**class** {*class-name*| **class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the class so that you can configure or modify its policy. Enters policy-map class configuration mode. • Enter the class name or specify the default class (class-default).
Step 5	**shape** {**average**	**peak**} **percent** *percentage* [**be** *excess-burst-in-msec* **ms**] [**bc** *committed-burst-in-msec* **ms**] **Example:** `Router(config-pmap-c)# shape average percent 25 be 300 ms bc 400 ms`	Configures either average or peak rate traffic shaping on the basis of the specified bandwidth percentage and the optional burst sizes. • Enter the bandwidth percentage and optional burst sizes.

	Command or Action	Purpose
Step 6	**end** **Example:** `Router(config-pmap-c)# end`	Exits policy-map class configuration mode.

Attaching the Policy Map to an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **pvc** [*name*] *vpi* / *vci* [**ilmi** | **qsaal** | **smds**]
5. **service-policy** {**input**| **output**} *policy-map-name*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**interface** *type number*
	Example:
	`Router(config)#`
	`interface serial4/0/0`
		Configures an interface (or subinterface) type and enters interface configuration mode.
		• Enter the interface type number.
		Note Depending on the needs of your network, you may need to attach the policy map to a subinterface, an ATM PVC, a Frame Relay DLCI, or other type of interface.

Step 4	**pvc** [*name*] *vpi* / *vci* [**ilmi**	**qsaal**	**smds**] **Example:** `Router(config-if)# pvc cisco 0/16 ilmi`	(Optional) Creates or assigns a name to an ATM PVC and specifies the encapsulation type on an ATM PVC. Enters ATM VC configuration mode.

	Command or Action	Purpose
		Note This step is required only if you are attaching the policy map to an ATM PVC. If you are not attaching the policy map to an ATM PVC, skip this step and proceed with Step 5.

Step 5	**service-policy** {**input**| **output**} *policy-map-name*
	Example:
	`Router(config-if)#`
	`service-policy input policy1`
	Example:
		Specifies the name of the policy map to be attached to the input or output direction of the interface.
		Note Policy maps can be configured on ingress or egress routers. They can also be attached in the input or output direction of an interface. The direction (input or output) and the router (ingress or egress) to which the policy map should be attached varies according your network configuration. When using the **service-policy** command to attach the policy map to an interface, be sure to choose the router and the interface direction that are appropriate for your network configuration.
		Note Traffic shaping is supported on service policies attached to output interfaces or output VCs only.
		• Enter the policy map name.

Step 6	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode.

Verifying the QoS Percentage-Based Shaping Configuration

SUMMARY STEPS

1. **enable**
2. **show class-map** [*class-map-name*]
3. **show policy-map interface** *interface-name*
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**show class-map** [*class-map-name*] **Example:** `Router# show class-map class1`	Displays all information about a class map, including the match criterion. • Enter class map name.

Step 3	**show policy-map interface** *interface-name*
	Example:
	`Router#`
	`show policy-map interface serial4/0/0`
		Displays the packet statistics of all classes that are configured for all service policies either on the specified interface or subinterface or on a specific PVC on the interface.
		• Enter the interface type and number.

Step 4	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Troubleshooting Tips

The commands in the Verifying the QoS Percentage-Based Shaping Configuration, on page 39 section allow you to verify that you achieved the intended configuration and that the feature is functioning correctly. If, after using the **show** commands listed above, you find that the configuration is not correct or the feature is not functioning as expected, perform these operations:

If the configuration is not the one you intended, complete the following procedures:

1. Use the **show running-config** command and analyze the output of the command.
2. If the policy map does not appear in the output of the **show running-config** command, enable the **logging console** command.
3. Attach the policy map to the interface again.

If the packets are not being matched correctly (for example, the packet counters are not incrementing correctly), complete the following procedures:

1. Run the **show policy-map**command and analyze the output of the command.
2. Run the **show running-config** command and analyze the output of the command.
3. Use the **show policy-map interface** command and analyze the output of the command. Check the the following findings:
   a. If a policy map applies queueing, and the packets are matching the correct class, but you see unexpected results, compare the number of the packets in the queue with the number of the packets matched.
   b. If the interface is congested, and only a small number of the packets are being matched, check the tuning of the transmission (tx) ring, and evaluate whether the queueing is happening on the tx ring. To do this, use the **show controllers** command, and look at the value of the tx count in the output of the command.

Configuration Examples for QoS Percentage-Based Shaping

Example Specifying Traffic Shaping on the Basis of a Bandwidth Percentage

The following example configures traffic shaping using an average shaping rate on the basis of a percentage of bandwidth. In this example, 25 percent of the bandwidth has been specified. Additionally, an optional be value and bc value (300 ms and 400 ms, respectively) have been specified.

```
Router> enable
Router# configure terminal
Router(config)# policy-map policy1
Router(config-pmap)# class class1

Router(config-pmap-c)# shape average percent 25 be 300 ms bc 400 ms

Router(config-pmap-c)# end
```

After the policy map and class maps are configured, the policy map is attached to interface as shown in the following example:

```
Router> enable
Router# configure terminal
Router(config)#

interface serial4/0/0
Router(config-if)#

service-policy input policy1
Router(config-if)# end
```

Example Verifying the QoS Percentage-Based Shaping Configuration

This section contains sample output from the **show policy-map**command and the **show policy-map interface**command. The output from these commands can be used to verify and monitor the configuration on your network.

The following is sample output from the **show policy-map** command. This sample output displays the contents of a policy map called "policy3." In policy 3, average rate traffic shaping on the basis of an committed information rate (CIR) of 30 percent has been configured, and the bc and be have been specified in milliseconds.

```
Router# show policy-map
  Policy Map policy3
    Class class-default
      Average Rate Traffic Shaping
      cir 30% bc 10 (msec) be 10 (msec)
```

The following is sample output from the **show policy-map interface** command. This sample displays the statistics for the serial 2/0 interface on which average rate traffic shaping has been enabled.

```
Router# show policy-map interface serial2/0/0
 Serial2/0/0
  Service-policy output: policy3 (1032)
```

```
Class-map: class-default (match-any) (1033/0)
  0 packets, 0 bytes
  5 minute offered rate 0 bps, drop rate 0 bps
  Match: any  (1034)
    0 packets, 0 bytes
    5 minute rate 0 bps
  Queueing
  queue limit 64 packets
  (queue depth/total drops/no-buffer drops) 0/0/0
  (pkts queued/bytes queued) 0/0
  shape (average) cir 614400 bc 6144 be 6144
  target shape rate 614400
```

In this example, the CIR is displayed in bps, and both the committed burst (bc) and excess burst (be) are displayed in bits.

The CIR, bc, and be are calculated on the basis of the formulas described below.

Formula for Calculating the CIR

When calculating the CIR, the following formula is used:

CIR percentage specified (as shown in the output of the **show policy-map**command) * bandwidth (BW) of the interface (as shown in the output of the **show interfaces** command) = total bits per second

On the serial 2/0 interface, the bandwidth (BW) is 2048 kbps. To see the bandwidth of the interface, use the **show interfaces** command. A sample is shown below:

```
Router # show interfaces serial2/0/0
Serial2/0 is administratively down, line protocol is down
  Hardware is M4T
  MTU 1500 bytes, BW 2048 Kbit, DLY 20000 usec, rely 255/255, load 1/255
```

Therefore, the following values are used in the formula:

30% * 2048 kbps = 614400 bps

Formula for Calculating the Committed Burst (bc) and the Excess Burst (be)

When calculating both the bc and the be, the following formula is used:

The bc (or be) in milliseconds (as shown in the **show policy-map** command) * the CIR in kilobytes (as shown in the **show policy-map** command) / 1000 = total number of bits

Therefore, the following values are used in the formula:

10 ms * 614400 bps = 6144 bits

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*

Related Topic	Document Title
Modular QoS Command-Line Interface (CLI) (MQC) information about attaching policy maps to interfaces	"Applying QoS Features Using the MQC" module
Traffic shaping concepts and overview	"Policing and Shaping Overview" module
Traffic policing	"Traffic Policing" module

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2697	*A Single Rate Three Color Marker*
RFC 2698	*A Two Rate Three Color Marker*

Technical Assistance

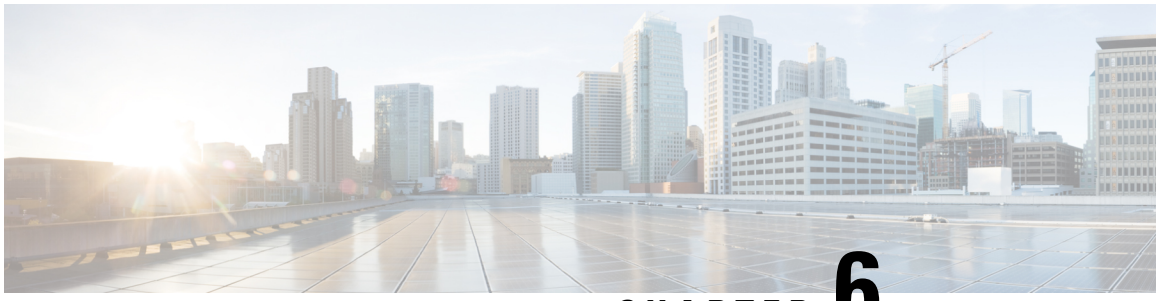
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for QoS Percentage-Based Shaping

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 4: Feature Information for QoS: Percentage-Based Shaping



CHAPTER 6

Ethernet Overhead Accounting

The Ethernet Overhead Accounting feature enables the router to account for downstream Ethernet frame headers when applying shaping to packets.

Restrictions for Ethernet Overhead Accounting

- Ethernet overhead accounting allows the automatic inclusion of downstream Ethernet frame headers in the shaped rate.
- If you enable overhead accounting on a child policy, you must enable overhead accounting on the parent policy.
- In a policy map, you must either enable overhead accounting for all classes in the policy or disable overhead accounting for all classes in the policy. You cannot enable overhead accounting for some classes and disable overhead accounting for other classes in the same policy.
- Overhead accounting is not reflected in any QoS counters (classification, policing, or queuing).
- Implicit ATM overhead accounting for policers are not supported.
- Implicit L2 overhead (ATM or otherwise) for policers are not supported for certain logical targets (tunnels) when the policy is applied to the logical target. The same limitation exists for queuing and scheduling overhead accounting.
- Police overhead cannot be configured on conditional policers (priority and rate), however, the priority queue it used will inherit the queueing overhead from parent shaper if configured.
- Police overhead is not added to the counters and are not reflected in statistics reported by the control plane.
- The overhead accounting type or value used by policing within a policy map and between the parent policy map and the child policy map (in a hierarchical policy map structure) must be consistent.

- The overhead accounting type or value used by queuing features within a policy map and between the parent policy map and the child policy map (in a hierarchical policy map structure) must be consistent.

Information About Ethernet Overhead Accounting

Benefits of Ethernet Overhead Accounting

The Ethernet Overhead Accounting feature enables the router to account for downstream Ethernet frame headers when applying shaping to packets. A user-defined offset specifies the number of overhead bytes that the router is to use when calculating the overhead per packet. Valid offset values are from +63 bytes to -63 bytes of overhead. Before applying shaping, the router calculates the overhead.

Any interface that supports QoS policies will support overhead accounting. Using the **policy-map**, **shape** or **bandwidth** command, you can configure accounting on the interfaces.

Subscriber Line Encapsulation Types

The *subscriber-encapsulation* argument of the **shape** and **bandwidth** commands specifies the encapsulation type at the subscriber line. The router supports the following subscriber line encapsulation types:

- snap-1483routed
- mux-1483routed
- snap-dot1q-rbe
- mux-dot1q-rbe
- snap-pppoa
- mux-pppoa
- snap-rbe
- mux-rbe

Overhead Calculation on the Router

When calculating overhead for traffic shaping, the router considers the encapsulation type used between the broadband aggregation system (BRAS) and the digital subscriber line access multiplexer (DSLAM) and between the DSLAM and the customer premises equipment (CPE).

The table below describes the fields that the router uses for the various encapsulation types when calculating ATM overhead.

Table 5: Overhead Calculation

Encapsulation Type	Number of Bytes	Description
802.1Q	18	6-byte destination MAC address + 6-byte source MAC address + 2-byte protocol ID (0x8100) + 2-byte VID/CFI/PRIORITY + 2-byte length/type
802.3	14	6-byte destination MAC address + 6-byte source MAC address + 2-byte protocol ID (0x8000)
AAL5 MUX plus 1483	8	8-byte AAL5 trailer
AAL5 MUX plus PPPoA	10	8-byte AAL5 trailer + 2-byte protocol ID (0x002
AAL5 SNAP plus 1483	18	8-byte AAL5 trailer + 3-byte LLC header (0xAAAA03) + 3-byte OUI (0x0080c2) + 2-byte protocol ID (0x0007) + 2-byte PAD (0x0000)
AAL5 SNAP plus PPPoA	12	8-byte AAL5 trailer + 3-byte LLC header (0xFEFE03) + 1-byte protocol ID (0xCF)
PPPoE	6	1-byte version/type (0x11) + 1-byte code (0x00) + 2-byte session ID + 2-byte lengt
qinq	22	6-byte destination MAC address + 6-byte source MAC address + 2-byte protocol ID (0x8100) + 2-byte VID/CFI/PRIORITY + 2-byte protocol ID + 2-byte inner tag + 2-byte length or type

Overhead Accounting and Hierarchical Policies

In hierarchical policies, you can configure overhead accounting for policing, shaping, and bandwidth on top-level parent policies, middle-level child policies, and bottom-level child policies. Overhead accounting policies configured at the parent or grandparent level are inherited by the child queueing features. Overhead accounting configured on a child policy must also be configured on the parent policy; therefore configuring on the parent or grandparent level is easier.

The parent and child classes must specify the same encapsulation type when enabling overhead accounting and configuring an offset using the **user-defined** *offset* [**atm**] arguments of the **bandwidth** (policy-map class) command.

The table below summarizes the configuration requirements for overhead accounting.

Table 6: Overhead Accounting Configuration Requirements

Policy Map or Class	Current Configuration	Configuration Requirement
Parent	Enabled	Enabled on child policy
Child	Enabled	Enabled on parent policy
Child class	Enabled	Enabled on all classes in the child policy map, except priority classes with policing

Policy Map or Class	Current Configuration	Configuration Requirement
Child class (nonpriority without policing)	Disabled	Disabled on all classes in the child policy map
Child class (priority with policing)	Disabled	Disabled or enabled on all nonpriority classes in the child policy map

Overhead Accounting and Priority Queues

Overhead accounting configuration is supported for queuing features (shape, bandwidth and priority) and non-queuing feature (police) separately. However, priority queue can be integrated with policer. When overhead accounting is configured on a priority queue, through inheritance, it operates in the following fashion:

- Overhead accounting is added to (or subtracted from) the priority packet for queuing features in the hierarchy (for example, shape in the parent class).
- Overhead accounting is not added to the packet for priority rate enforcement (**priority** {*bandwidth-kbps* | **percent** *percentage*} [**burst**]). Although policing overhead accounting is supported, it does not apply to the conditional policer (rate enforcement is implemented through this conditional policer).

How to Configure Ethernet Overhead Accounting

Configuring Ethernet Overhead Accounting in a Hierarchical Policy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-map-name*
5. **bandwidth** {*bandwidth-kbps* | [**remaining**] **percent** *percentage*} **account** {**qinq** | **dot1q**} {**aal5** | **aal3**} *subscriber-encapsulation* **user-defined** *offset* [**atm**]
6. **exit**
7. **policy-map** *policy-map-name*
8. **class class-default**
9. **shape** [**average**] *rate* **account** {{**qinq** | **dot1q**} {**aal5** | **aal3**} *subscriber-encapsulation* | **user-defined** *offset* [**atm**]}
10. **service-policy** *policy-map-name*
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map Business`	Creates or modifies the child policy. Enters policy-map configuration mode. • The *policy-map-name* argument represents the name of the child policy map.
Step 4	**class** *class-map-name* **Example:** `Router(config-pmap)# class video`	Assigns the traffic class you specify to the policy map. Enters policy-map class configuration mode. • The *class-map-name* argument represents the name of a previously configured class map.
Step 5	**bandwidth** {*bandwidth-kbps* | [**remaining**] **percent** *percentage*} **account** {**qinq** | **dot1q**} {**aal5** | **aal3**} *subscriber-encapsulation* **user-defined** *offset* [**atm**] **Example:** `Router(config-pmap-c)# bandwidth 8000 account dot1q aal5 snap-pppoa`	Enables class-based fair queueing and overhead accounting. • *bandwidth-kbps*—The minimum bandwidth allocated for a class belonging to a policy map. Valid values are from 8 to 2,488,320, which represents from 1 to 99 percent of the link bandwidth. • *percentage*—The maximum percentage of the link bandwidth allocated for a class belonging to a policy map. Valid values are from 1 to 99. • **remaining** *percentage*—The minimum percentage of unused link bandwidth allocated for a class belonging to a policy map. Valid values are from 1 to 99. • **account**—Enables ATM overhead accounting. • **qinq**—Specifies queue-in-queue encapsulation as the BRAS-DSLAM encapsulation type. • **dot1q**—Specifies IEEE 802.1Q VLAN encapsulation as the BRAS-DSLAM encapsulation type. • **aal5**—Specifies the ATM Adaptation Layer 5 that supports connection-oriented variable bit rate (VBR) services. • **aal3**—Specifies the ATM Adaptation Layer 5 that supports both connectionless and connection-oriented links.

	Command or Action	Purpose
		• *subscriber-encapsulation*—Specifies the encapsulation type at the subscriber line. For more information, see the "Configuring Ethernet Overhead Accounting in a Hierarchical Policy" section. • **user-defined**—Indicates that the router is to use the offset value that you specify when calculating ATM overhead. • *offset*—Specifies the number of bytes that the router is to use when calculating overhead. Valid values are from -63 to 63 bytes. • **atm**—(Optional) Applies the ATM cell tax in the ATM overhead calculation.
Step 6	**exit** **Example:** `router(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 7	**policy-map** *policy-map-name* **Example:** `Router(config-pmap)# policy-map Test`	Creates or modifies the top-level parent policy. • *policy-map-name*—Specifies the name of the parent policy map.
Step 8	**class class-default** **Example:** `Router(config-pmap)# class class-default`	Specifies a default class.
Step 9	**shape** [**average**] *rate* **account** {{**qinq** | **dot1q**} {**aal5** | **aal3**} *subscriber-encapsulation* | **user-defined** *offset* [**atm**]} **Example:** `Router(config-pmap-c)# shape 8000 account qinq aal5 snap-dot1-rbe`	Shapes traffic to the indicated bit rate and enables overhead accounting. • **average** (Optional)—Is the committed burst (Bc) that specifies the maximum number of bits sent out in each interval. This option is only supported on the PRE3. • **rate**—Indicates the bit rate used to shape the traffic, in bits per second. When this command is used with backward explicit congestion notification (BECN) approximation, the bit rate is the upper bound of the range of bit rates that are permitted. • **account**—Enables ATM overhead accounting. • **qinq**—Specifies queue-in-queue encapsulation as the BRAS-DSLAM encapsulation type. • **dot1q**—Specifies IEEE 802.1Q VLAN encapsulation as the BRAS-DSLAM encapsulation type.

	Command or Action	Purpose
		• **aal5**—Specifies the ATM Adaptation Layer 5 that supports connection-oriented variable bit rate (VBR) services. • **aal3**—Specifies the ATM Adaptation Layer 5 that supports both connectionless and connection-oriented links. • *subscriber-encapsulation*—Specifies the encapsulation type at the subscriber line. For more information, see the "Configuring Ethernet Overhead Accounting in a Hierarchical Policy" section. • **user-defined**—Indicates that the router is to use the offset value that you specify when calculating ATM overhead. • *offset*—Specifies the number of bytes that the router is to use when calculating overhead. Valid values are from -63 to 63 bytes. • **atm**—(Optional) Applies the ATM cell tax in the ATM overhead calculation. Configuring both the offset and atm options adjusts the packet size to the offset size and then adds the ATM cell tax.
Step 10	**service-policy** *policy-map-name* **Example:** `Router(config-pmap-c)# service-policy map1`	Applies a child policy to the parent class-default class. *policy-map-name*—Specifies the name of a previously configured child policy map. **Note** Do not specify the input or output keywords when applying a child policy to a parent class-default class.
Step 11	**end** **Example:** `Router(config-pmap-c)# end`	Exits policy-map class configuration mode and returns to privileged EXEC mode.

Configuration Examples for Ethernet Overhead Accounting

Example: Enabling Ethernet Overhead Accounting

The following configuration example shows how to enable Ethernet overhead accounting. In the example, the configuration of the policy map named ethernet_ovrh shapes class-default traffic at a rate of 200,000 kbps and enables overhead accounting with a user-defined value of 18. The ethernet_ovrh policy is attached to Gigabit Ethernet subinterface 1/0/0.100, thereby enabling overhead accounting on the subinterface.

```
Router# configure-terminal
Router(config)# policy-map ethernet_ovrh
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 200000 account user-defined 18
!
Router(config)# interface GigabitEthernet1/0/0.100
Router(config-subif)# service-policy output ethernet_ovrh
!
Router# show running-config | begin 1/0/0.100

interface GigabitEthernet1/0/0.100
encapsulation dot1Q 101
pppoe enable group group_pta
service-policy output ethernet_ovrh
```

Example: Verifying Ethernet Overhead Accounting with User-Defined Option

The following sample output for the policy map named ethernet_ovrh indicates that Ethernet overhead accounting is enabled for shaping and that the user-defined offset is 18 bytes. The sample output from the **show policy-map** command indicates that the ethernet_ovrh policy map is attached to the Gigabit Ethernet subinterface 1/0/0.100, enabling overhead accounting on the subinterface.

```
Router# show policy-map ethernet_ovrh

Policy Map ethernet_ovrh
Class class-default
Average Rate Traffic Shaping
cir 200000 (bps) account user-defined 18
Router# show policy-map interface GigabitEthernet1/0/0.100
GigabitEthernet1/0/0.100
Service-policy output: ethernet_ovrh
Class-map: class-default (match-any)
0 packets, 0 bytes
30 second offered rate 0 bps, drop rate 0 bps
Match: any
0 packets, 0 bytes
30 second rate 0 bps
Queueing
queue limit 8 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
shape (average) cir 200000, bc 800, be 800
target shape rate 200000
Overhead Accounting Enabled
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	Cisco IOS Quality of Service Solutions Command Reference

Related Topic	Document Title
Policing and shaping	"Policing and Shaping Overview" module
Class maps	"Applying QoS Features Using the MQC" module
Policy maps	"Applying QoS Features Using the MQC" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	—

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

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

Technical Assistance

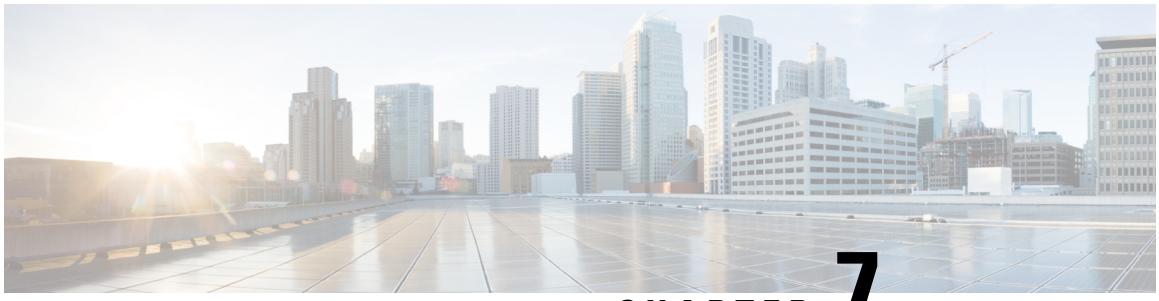
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Ethernet Overhead Accounting

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 7: Feature Information for Ethernet Overhead Accounting



CHAPTER 7

MQC Traffic Shaping Overhead Accounting for ATM

The MQC Traffic Shaping Overhead Accounting for ATM feature enables a broadband aggregation system (BRAS) to account for various encapsulation types when applying quality of service (QoS) functionality to packets. Typically, in Ethernet digital subscriber line (DSL) environments, the encapsulation from the router to the digital subscriber line access multiplexer (DSLAM) is Gigabit Ethernet and the encapsulation from the DSLAM to the customer premises equipment (CPE) is ATM. ATM overhead accounting enables the router to account for ATM encapsulation on the subscriber line and for the overhead added by cell segmentation. This functionality enables the service provider to prevent overruns at the subscriber line and ensures that the router executes QoS features on the actual bandwidth used by ATM packets.

Prerequisites for Traffic Shaping Overhead Accounting for ATM

Traffic classes must be configured using the **class-map** command.

Restrictions for Traffic Shaping Overhead Accounting for ATM

- The overhead accounting type or value used within a policy map and between the parent policy map and the child policy map (in a hierarchical policy map structure) must be consistent.
- You must attach a policy map that is configured with ATM overhead accounting to only an Ethernet interface (or an IP session on an Ethernet interface).
- Ethernet overhead accounting allows the automatic inclusion of downstream Ethernet frame headers in the shaped rate.
- If you enable overhead accounting on a child policy, you must enable overhead accounting on the parent policy.

- In a policy map, you must either enable overhead accounting for all classes in the policy or disable overhead accounting for all classes in the policy. You cannot enable overhead accounting for some classes and disable overhead accounting for other classes in the same policy.
- Overhead accounting is not reflected in any QoS counters (classification, policing, or queuing).
- Implicit ATM overhead accounting for policers are not supported.
- Implicit L2 overhead (ATM or otherwise) for policers are not supported for certain logical targets (tunnels) when the policy is applied to the logical target. The same limitation exists for queuing and scheduling overhead accounting.
- Police overhead cannot be configured on conditional policers (priority and rate), however, the priority queue it used will inherit the queueing overhead from parent shaper if configured.
- Police overhead is not added to the counters and are not reflected in statistics reported by the control plane.
- The overhead accounting type or value used by policing within a policy map and between the parent policy map and the child policy map (in a hierarchical policy map structure) must be consistent.
- The overhead accounting type or value used by queuing features within a policy map and between the parent policy map and the child policy map (in a hierarchical policy map structure) must be consistent.

Information About Traffic Shaping Overhead Accounting for ATM

Benefits of Traffic Shaping Overhead Accounting for ATM

The Traffic Shaping Overhead Accounting for ATM feature enables the broadband aggregation system (BRAS) to account for various encapsulation types when applying QoS to packets. Typically, in Ethernet digital subscriber line (DSL) environments, the encapsulation from the BRAS to the DSLAM is Gigabit Ethernet and the encapsulation from the DSLAM to the CPE is ATM. ATM overhead accounting enables the BRAS to account for ATM encapsulation on the subscriber line and for the overhead added by cell segmentation. This functionality enables the service provider to prevent overruns at the subscriber line and ensures that the router executes QoS features on the actual bandwidth used by ATM subscriber traffic.

BRAS and Encapsulation Types

Broadband aggregation system (BRAS) uses the encapsulation type that is configured for the DSLAM-CPE side to calculate the ATM overhead per packet.

DSLAM-CPE encapsulation types are based on Subnetwork Access Protocol (SNAP) and multiplexer (MUX) formats of ATM adaptation layer 5 (AAL5), followed by routed bridge (RBE), x-1483, x-dot1q-rbe, IP, PPP over Ethernet (PPPoE), or PPP over ATM (PPPoA) encapsulations. Because the DSLAM treats IP and PPPoE packets as payload, the BRAS does not account for IP and PPPoE encapsulations.

On the BRAS-DSLAM side, encapsulation is IEEE 802.1Q VLAN or Q-in-Q (qinq). However, because the DSLAM removes the BRAS-DSLAM encapsulation, the BRAS does not account for 802.1Q or qinq encapsulation.

AAL5 segmentation processing adds the additional overhead of the 5-byte cell headers, the AAL5 Common Part Convergence Sublayer (CPCS) padding, and the AAL5 trailer. For more information, see the ATM Overhead Calculation, on page 57.

Subscriber Line Encapsulation Types

The router supports the following subscriber line encapsulation types:

- snap-rbe
- mux-rbe
- snap-dot1q-rbe
- mux-dot1q-rbe
- snap-pppoa
- mux-pppoa
- snap-1483routed
- mux-1483routed
- snap-rbe-dot1q
- mux-rbe-dot1q

Note The encapsulation types listed above are for AAL5, qinq, and dot1q encapsulations. User-defined encapsulations with offsets based on the platform in use are also supported.

ATM Overhead Calculation

The Traffic Shaping Overhead Accounting for ATM feature prevents oversubscription of a subscriber line by accounting for the ATM encapsulation overhead at the BRAS. When calculating the ATM overhead, the Traffic Shaping Overhead Accounting for ATM feature considers the following:

- The encapsulation type used by the BRAS
- The CPCS trailer overhead
- The encapsulation type used between the DSLAM and the CPE

The offset size (a parameter used to calculate ATM overhead accounting) is calculated using the following formula:

Offset size in bytes = (CPCS trailer overhead) + (DSLAM to CPE) - (BRAS encapsulation type)

See the table below for the offset sizes, in bytes, derived from this formula.

This offset size, along with the packet size and packet assembler/disassembler (PAD) byte overhead in the CPCS, is used by the router to calculate the ATM overhead accounting rate.

Note A CPCS trailer overhead of 8 bytes corresponds to AAL5. A CPCS trailer overhead of 4 bytes corresponds to AAL3, but AAL3 is not supported.

Table 8: Offset Sizes, in Bytes, Used for ATM Overhead Calculation

Encapsulation Type in Use	BRAS	CPCS Trailer Overhead	DSLAM to CPE	Offset Size
dot1q mux-1483routed	18	8	3	-7
dot1q snap-1483routed	18	8	6	-4
dot1q mux-rbe	18	8	14	4
dot1q snap-rbe	18	8	24	14
dot1q mux-dot1q-rbe	18	8	18	8
dot1q snap-dot1q-rbe	18	8	28	18
qot1q mux-pppoa	18 + 6	8	2	-14
qot1q snap-pppoa	18 + 6	8	4	-12
qinq mux-1483routed	22	8	3	-11
qinq snap-1483routed	22	8	6	-8
qinq mux-rbe	22	8	14	0
qinq snap-rbe	22	8	24	10
qinq mux-dot1q-rbe	22	8	18	4
qing snap-dot1q-rbe	22	8	28	14
qinq mux-pppoa	22 + 6	8	2	-18
qinq snap-pppoa	22 + 6	8	4	-16

ATM Overhead Accounting and Hierarchical Policies

In hierarchical policies, you can enable ATM overhead accounting for shaping and bandwidth on parent policies and child policies. You are not required to enable ATM overhead accounting on a traffic class that does not contain the **bandwidth** or **shape** command. If you enable ATM overhead accounting on a child policy, then you must enable ATM overhead accounting on the parent policy. The parent and child classes must specify the same encapsulation type when ATM overhead accounting is enabled.

Overhead Accounting and Priority Queues

Overhead accounting configuration is supported for queuing features (shape, bandwidth and priority) and non-queuing feature (police) separately. However, priority queue can be integrated with policer. When overhead accounting is configured on a priority queue, through inheritance, it operates in the following fashion:

- Overhead accounting is added to (or subtracted from) the priority packet for queuing features in the hierarchy (for example, shape in the parent class).
- Overhead accounting is not added to the packet for priority rate enforcement (**priority** {*bandwidth-kbps* | **percent** *percentage*} [**burst**]). Although policing overhead accounting is supported, it does not apply to the conditional policer (rate enforcement is implemented through this conditional policer).

How to Configure Traffic Shaping Overhead Accounting for ATM

Configuring Traffic Shaping Overhead Accounting for ATM in a Hierarchical Policy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-map-name*
5. bandwidth {bandwidth-kbps | percent percentage | remaining percent percentage} account {{qinq | dot1q} {**aal5** | **aal3**} {subscriber-encapsulation}} | {user-defined offset [atm]}}
6. **bandwidth remaining ratio** *ratio* [**account** {**qinq** | **dot1q**} [**aal5**|**aal3**] {*subscriber-encapsulation* | **user-defined***offset*[atm]}]
7. **shape** [**average** |**peak**] mean-*rate*[*burst-size*] [*excess-burst-size*] account {{{qinq | dot1q} {aal5 | aal3} {subscriber-encapsulation}} | {user-defined offset [atm]}}
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map Business`	Creates or modifies the child policy and enters policy-map configuration mode. • Enter the policy map name. This is the name of the child policy.
Step 4	**class** *class-map-name* **Example:** `Router(config-pmap)# class video`	Assigns the traffic class that you specify for the policy map and enters policy-map class configuration mode. • Enter the traffic class name. This is the name of the previously configured class map.
Step 5	bandwidth {bandwidth-kbps | percent percentage | remaining percent percentage} account {{qinq | dot1q} {**aal5** | **aal3**} {subscriber-encapsulation}} | {user-defined offset [atm]}} **Example:** `Router(config-pmap-c)# bandwidth 8000 account dot1q aal5 snap-pppoa`	Enables Class-Based Weighted Fair Queueing (CBWFQ) on the basis of the keywords and arguments specified, such as the following: • *bandwidth-kbps* --Specifies or modifies the minimum bandwidth allocated for a class that belongs to a policy map. Valid values are from 8 to 2488320, which represents from 1 to 99 percent of the link bandwidth. • **percent** *percentage* --Specifies or modifies the minimum percentage of the link bandwidth allocated for a class that belongs to a policy map. Valid values are from 1 to 99. • **remaining percent** *percentage* --Specifies or modifies the minimum percentage of unused link bandwidth allocated for a class that belongs to a policy map. Valid values are from 1 to 99. • **account** --Enables ATM overhead accounting. • **qinq** --Specifies queue-in-queue encapsulation as the BRAS-DSLAM encapsulation type. • **dot1q** --Specifies IEEE 802.1Q VLAN encapsulation as the BRAS-DSLAM encapsulation type. • **aal5** --Specifies the ATM adaptation layer 5 that supports connection-oriented variable bit rate (VBR) services. • **aal3** --Specifies the ATM adaptation layer 5 that supports both connectionless and connection-oriented links. • *subscriber-encapsulation* --Specifies the encapsulation type at the subscriber line. For more information, see the Subscriber Line Encapsulation Types, on page 57. • **user-defined** --Specifies the offset size that the router uses when calculating the ATM overhead.

	Command or Action	Purpose
		• *offset* --Specifies the offset size when calculating ATM overhead. Valid values are from -63 to +63 bytes. • **atm** --(Optional) Applies the ATM cell tax in the ATM overhead calculation.
Step 6	**bandwidth remaining ratio** *ratio* [**account** {**qinq** | **dot1q**} [**aal5**|**aal3**] {*subscriber-encapsulation* | **user-defined***offset*[atm]}] **Example:** `Router(config-pmap-c)# bandwidth remaining ratio 10 account dot1q aal5 snap-pppo`	(Optional) Specifies the bandwidth-remaining ratio for the subinterface along with ATM accounting parameters: • *ratio* --Specifies the bandwidth-remaining ratio for the subinterface. Valid values are 1 to 100. The default value is 1. **Note** For the Cisco 7600 series router, valid values are from 1 to 10000. The default value is 1. • **account** --Enables ATM overhead accounting. • **qinq** --Specifies queue-in-queue encapsulation as the BRAS-DSLAM encapsulation type. • **dot1q** --Specifies IEEE 802.1Q VLAN encapsulation as the BRAS-DSLAM encapsulation type. • **aal5** --Specifies the ATM adaptation layer 5 that supports connection-oriented VBR services. • **aal3** --Specifies the ATM adaptation layer 5 that supports both connectionless and connection-oriented links. • *subscriber-encapsulation* --Specifies the encapsulation type at the subscriber line. For more information, see the Subscriber Line Encapsulation Types, on page 57. • **user-defined** --Specifies the offset size that the router uses when calculating the ATM overhead. • *offset* --Specifies the offset size, in bytes, when calculating ATM overhead. Valid values are from -63 to +63. • **atm** --(Optional) Applies the ATM cell tax in the ATM overhead calculation.
Step 7	**shape** [**average** |**peak**] mean-*rate*[*burst-size*] [*excess-burst-size*] account {{{qinq | dot1q} {aal5 | aal3} {subscriber-encapsulation}} | {user-defined offset [atm]}} **Example:** `Router(config-pmap-c)# shape 8000 account qinq aal5 snap-dot1q-rbe`	Shapes traffic to the indicated bit rate and enables ATM overhead accounting on the basis of the keywords and arguments specified, such as the following: • **average** --(Optional) The committed burst (Bc) that specifies the maximum number of bits sent out in each interval.

	Command or Action	Purpose
		• **peak** --(Optional) Specifies the maximum number of bits sent out in each interval (the Bc + excess burst [Be]). The Cisco 10000 router and the SIP400 (on the Cisco 7600 series router) do not support this option. • *mean-rate* --Also called committed information rate (CIR). Indicates the bit rate used to shape the traffic, in bits per second. • *burst-size* --(Optional) The number of bits in a measurement interval (Bc). • *excess-burst-size* --(Optional) The acceptable number of bits permitted to go over the Be. • **account** --Enables ATM overhead accounting. • **qinq** --Specifies queue-in-queue encapsulation as the BRAS-DSLAM encapsulation type. • **dot1q** --Specifies IEEE 802.1Q VLAN encapsulation as the BRAS-DSLAM encapsulation type. • **aal5** --The ATM adaptation layer 5 that supports connection-oriented variable bit rate (VBR) services. • **aal3** --Specifies the ATM Adaptation Layer 5 that supports both connectionless and connection-oriented links. You must specify either aal3 or aal5. • *subscriber-encapsulation* --Specifies the encapsulation type at the subscriber line. For more information, see the Subscriber Line Encapsulation Types, on page 57. • **user-defined** --Specifies the offset size that the router uses when calculating the ATM overhead. • *offset* --Specifies the offset size when calculating ATM overhead. Valid values are from -63 to +63 bytes. • **atm** --(Optional) Applies ATM cell tax in the ATM overhead calculation. Configuring both the *offset* and the **atm** options adjusts the packet size to the offset size and then adds ATM cell tax.
Step 8	**end** **Example:** `Router(config-pmap-c)# end`	Exits policy-map class configuration mode and returns to privileged EXEC mode.

Verifying the Configuration of Traffic Shaping Overhead Accounting for ATM

SUMMARY STEPS

1. **enable**
2. **show policy-map** [*policy-map-name*]
3. **show policy-map session**
4. **show running-config**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show policy-map** [*policy-map-name*] **Example:** `Router# show policy-map unit-test`	(Optional) Displays the configuration of all classes for a specified policy map or of all classes for all existing policy maps. • (Optional) Enter the policy map name.
Step 3	**show policy-map session** **Example:** `Router# show policy-map session`	(Optional) Displays the QoS policy map in effect for an IPoE/PPPoE session.
Step 4	**show running-config** **Example:** `Router# show running-config`	(Optional) Displays the contents of the currently running configuration file.
Step 5	**exit** **Example:** `Router# exit`	Exits privileged EXEC mode.

Configuration Examples for Traffic Shaping Overhead Accounting for ATM

Example Enabling Traffic Shaping Overhead Accounting for ATM

The following example shows how to enable ATM overhead accounting using a hierarchical policy map structure. The Child policy map has two classes: Business and Non-Business. The Business class has priority and is policed at 128,000 kbps. The Non-Business class has ATM overhead accounting enabled and has a bandwidth of 20 percent of the available bandwidth. The Parent policy map shapes the aggregate traffic to 256,000 kbps and enables ATM overhead accounting.

Notice that Layer 2 overhead accounting is not explicitly configured for the Business traffic class. If the class-default class of a parent policy has ATM overhead accounting enabled, you are not required to enable ATM overhead accounting on a child traffic class that does not contain the **bandwidth** or **shape** command. Therefore, in this example, the Business priority queue implicitly has ATM overhead accounting enabled because its parent class-default class has overhead accounting enabled.

```
policy-map Child
 class Business
  priority
  police 128000
 class Non-Business
  bandwidth percent 20 account dot1q aal5 snap-rbe-dot1q
  exit
 exit
policy-map Parent
 class class-default
  shape 256000 account dot1q aal5 snap-rbe-dot1q
  service-policy Child
```

In the following example, overhead accounting is enabled for bandwidth on the gaming and class-default class of the child policy map named subscriber_classes and on the class-default class of the parent policy map named subscriber_line. The voip and video classes do not have accounting explicitly enabled; these classes have ATM overhead accounting implicitly enabled because the parent policy has overhead accounting enabled. Notice that the features in the parent and child policies use the same encapsulation type.

```
policy-map subscriber_classes
 class voip
  priority level 1
  police 8000
 class video
  priority level 2
  police 8000
 class gaming
  bandwidth remaining percent 80 account dot1q aal5 snap-rbe-dot1q
 class class-default
  bandwidth remaining percent 20 account dot1q aal5 snap-rbe-dot1q
policy-map subscriber_line
 class class-default
  bandwidth remaining ratio 10 account dot1q aal5 snap-rbe-dot1q
  shape average 512 account aal5 dot1q snap-rbe-dot1q
  service policy subscriber_classes
```

Example Verifying Traffic Shaping Overhead Accounting for ATM

```
Router# show policy-map interface


Service-policy output:unit-test
Class-map: class-default (match-any)
100 packets, 1000 bytes
30 second offered rate 800 bps, drop rate 0 bps
Match: any
shape (average) cir 154400, bc 7720, be 7720
target shape rate 154400
overhead accounting: enabled
bandwidth 30% (463 kbps)
overhead accounting: disabled
queue limit 64 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(packets output/bytes output) 100/1000


Router# show policy-map session output

SSS session identifier 2 -
Service-policy output:  ATM_OH_POLICY
    Class-map: class-default (match-any)
      0 packets, 0 bytes
      30 second offered rate 0 bps, drop rate 0 bps
      Match: any
      Queueing
      queue limit 2500 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 0/0
      shape (average) cir 10000000, bc 40000, be 40000
      target shape rate 10000000
       Overhead Accounting Enabled
```

The following output from the **show running-config** command indicates that ATM overhead accounting is enabled for shaping. The BRAS-DSLAM encapsulation is dot1q and the subscriber line encapsulation is snap-rbe based on the AAL5 service.

```
subscriber policy recording rules limit 64
no mpls traffic-eng auto-bw timers frequency 0
call rsvp-sync
!
controller T1 2/0
framing sf
linecode ami
!
controller T1 2/1
framing sf
linecode ami
!
!
policy-map unit-test
class class-default
shape average percent 10 account dot1q aal5 snap-rbe
!
```

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC), hierarchical policies, policy maps	"Applying QoS Features Using the MQC" module
Policing and shaping traffic	"Policing and Shaping Overview" module

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	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

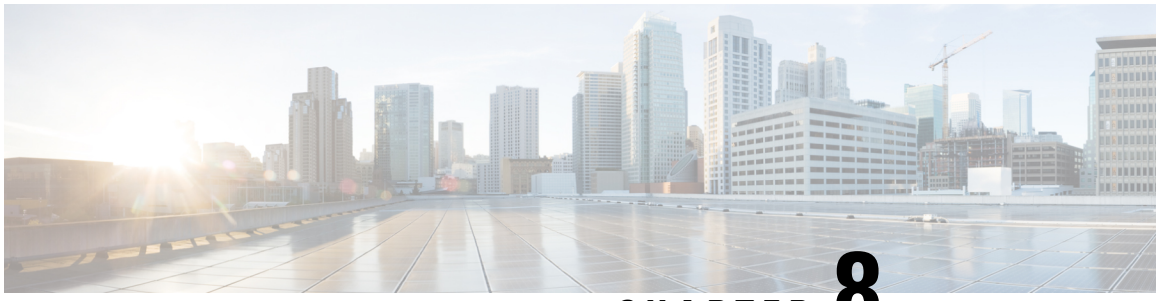
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for MQC Traffic Shaping Overhead Accounting for ATM

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 9: Feature Information for MQC Traffic Shaping Overhead Accounting for ATM



CHAPTER 8

QoS Policy Accounting

The QoS Policy Accounting feature helps you accurately account for traffic on your system. It also provides greater flexibility in assigning quality of service (QoS) configurations to subscribers. In addition, the QoS Accounting High Availability feature ensures that QoS accounting statistics persist, and that the RADIUS accounting billing server continues to report accounting counters during planned and unexpected Route Processor (RP) switchovers. This module describes how to configure QoS policy accounting, use subscriber templates, and activate subscriber accounting accuracy.

Prerequisites for QoS Policy Accounting

- PPP over Ethernet (PPPoE) or PPP over Ethernet over ATM (PPPoEoA) sessions are enabled.
- The RADIUS server is configured.
- Authentication, authorization, and accounting (AAA) is enabled.
- The subscriber's user profile on the RADIUS server has been created.
- A policy map is configured.
- A service template is configured.
- Traffic classes have been created.
- Stateful switchover (SSO) and In-service Software Upgrade (ISSU) prerequisites must be met. For more information, see the *Cisco IOS High Availability Configuration Guide*.

Restrictions for QoS Policy Accounting

- In system failover, the following occurs:

  - For QoS accounting configured statically at the policy map, QoS accounting statistics are reset to zero.
  - For QoS accounting configured dynamically using service templates, sessions no longer exist on the new active Route Processor (RP).

- Multicasting is not supported for QoS policy accounting services.
- The following QoS actions are not supported in service templates:
  - account
  - fair-queue
  - netflow-sampler
  - random-detect
- The following QoS filters are not supported in service templates:
  - atm
  - class-map
  - cos
  - destination-address
  - discard-class
  - fr-de
  - fr-dlci
  - input-interface
  - mpls
  - not
  - packet
  - source-address
  - vlan
- Service template definition lines may not exceed maximum configuration line length allowed by the Cisco IOS CLI. You may need to shorten shell variable names to stay within this limit.
- A template service activated on a session cannot be changed. Instead, you can deactivate it and activate a different template service.
- When a template service is active, a legacy complex parameterized string may not be used to change the QoS policy active on a session.
- IP address parameterization is supported only for IPv4 and only for named ACLs without remarks. IP addresses specified in the parameterized service activation are always added to the cloned ACL in this fixed pattern: "permit ip network mask any" and "permit ip any network mask".
- Service templates are supported only for PPP sessions and may not be activated on subinterfaces.
- Only one turbo button service can be active on a session at any given time. Turbo button service is any service that changes a QoS action other than “service-policy xxxx” (changing the child policy) in the class-default of the parent policy.
- Shell variables, QoS class map, and Access Control List (ACL) names may not have the following characters:
  - !
  - $

  - #
  - -
  - ,
  - >
  - <

- Service names are echoed back in the accounting records only for group accounting (when you use $_acctgrp in the service template).
- The IN/OUT QoS policy name active on a session is formed by concatenating the previously active QoS policy (or the static QoS policy specified in the last multiservice Change of Authorization (CoA) or Access-Accept).
- Two template services instantiated from the same service template may not be activated on the session at the same time. However, multiple template services instantiated from unrelated service templates can be active on a session at the same time.
- Template service support is available only for locally terminated PPP and PPP forwarded sessions on the Layer 2 Tunneling Protocol (L2TP) Access Concentrator (LAC).
- For PPP forwarded sessions on the LAC, to apply template services via Access-Accept, use the following configurations:
  - vpdn authen-before-forward.
  - Specify template services only in the user authorization profile (Access-Accept that is received after PPP authentication), not in the authentication profile.
- Only activate template services on the child policy under the parent class-default (only two levels) and on the parent policy (Turbo Button service).
- The default QoS policy can be only two levels deep (Parent + Child under class-default) and should not have a child policy configured under any class other than the class-default.
- A child policy should be configured under the default parent policy class-default in order for template services to be activated at the child level.
- Only rollback due to syntax error checking is supported.
- When multiple service activations or deactivations are included in a single CoA message, the failure of any operation (activation or deactivation) means that the CoA must roll back (undo) all previous operations to restore the session state to what it was before the CoA processing started. In other words, either all the operations must be processed successfully in a CoA or none at all. A CoA negative ACK (NACK) is sent to the RADIUS.
- For rollback to work during Access-Accept processing, subscriber service multiple-accept processing must be configured. The failure to process a service in an Access-Accept should roll back (undo) all previous services in the Access-Accept. The session will come up even if Access-Accept service processing fails.
- Errors originating in the platform or data plane will not trigger rollback which can result in an incomplete service.
- Do not modify a service template if its template services are in use or active on sessions. Use the **show subscriber policy ppm-shim-db** command to display which template services are in use.

Information About QoS Policy Accounting

RADIUS is a networking protocol that provides AAA management. Among other things, each RADIUS accounting message includes ingress and egress counters. The QoS Policy Accounting feature helps you resolve any inaccuracies between counters.

QoS Policy Accounting Feature in Groups

The QoS Policy Accounting feature collects and reports the following information to the RADIUS server per-session:

- Acct-Session-Id
- Ingress and egress packets/bytes/gigawords, packets, and bytes of successfully transmitted packets
- Parent-Session-Id
- Policy name and class or group name (if the QoS Policy Accounting feature is enabled on the group)
- Service name
- Username

When you enable the QoS Policy Accounting feature on a group and assign it a group name, this feature aggregates packets that meet the following criteria:

- Classified by traffic classes in the same group
- Included in the ingress or egress QoS policy applied on the same target

Separate Accounting Streams

If you do not assign a traffic class to a group, but instead assign it to an AAA method list, separate QoS policy accounting streams are created for each traffic class. Separate accounting streams allow you to differentiate between traffic that matches more than one class. Each unique target, direction, policy name, and class name has a unique RADIUS Acct-Session-Id value.

Service Templates

Service templates allow you to dynamically change QoS parameters without defining a new QoS policy on the CLI. You can change QoS policy when a session begins or any time after the session is established. Before you dynamically modify an active QoS deactivate the current service.

To understand service templates, learn the following terms:

- Service templates:
  - Are Cisco IOS shell functions
  - Have IN QoS policy-map definitions
  - Have OUT QoS policy-map definitions
  - Are programmatically invoked
  - Specify default values for shell variables

- Template services:
  - Are QoS service names with a parenthesis in them
  - Have a matching shell-map template definition
  - Are created dynamically during service template shell function execution
- IN Net effect policy map
- OUT Net effect policy map

The QoS Policy Accounting feature, describes how the Cisco IOS shell overrides default values of variables used in service template shell functions. QoS policy definitions inside a shell map may have shell variables in place of QoS action parameter values.

Using Service Templates

To create a service template, you write the service template in a text editor and you then copy the template to the CLI. The contents of a shell map block are treated as text.

When you define the service-template policy maps (policy map $_outgoing/$_incoming), there is no CLI help or prompts available. For example you cannot access the following CLI aids:

- Parser auto completion
- Command options
- Range help
- Syntax checking

Note There is no editor available to you in the CLI, if you make a mistake you must delete the entire service template and then configure it again from the start.

Verifying Service Templates

When you write a service template in a text editor you do not have a syntax checking facility. Therefore, before you activate your service template, you must verify its syntax. The following code sample shows how to verify the *voice-service1* service template. To verify your own template, replace *voice-service1* with your service template name.

```
(shell map voice-service1 police_rate=100000 prec_value=4 queue_size=1)
configure terminal
no policy-map test-svc_IN <-------- Removes previous service template verifications.
no policy-map test-svc_OUT <-------- Removes previous service template verifications.
no aaa-accounting group test_svc_GRP <-------- Removes previous service template
verifications.
end
trigger voice-service1 _incoming=test-svc_IN _outgoing=test-svc_OUT _acctgrp=test-svc_GRP
show policy-map test-svc-IN <--------
Ensure that the output matches the expected service template template service with default
 values.
show policy-map test-svc-OUT <--------
Ensure that the output matches the expected service template template service with default
 values.
```

Removing Service Templates

To remove a service template, at the command line enter:

```
no shell map voice-service1 police_rate=100000 prec_value=4 queue_size=1 in_h=class-default
 out_h=class-default
```

Where voice-service1 is the name of your service template.

Sample Service Templates

Service Template

This example shows a sample service template:

```
{
    configure terminal
    accounting group $_acctgrp list default
    policy-map $_outgoing
      class voip
       police $police_rate 60625 0 conform-action transmit exceed-action drop violate-action
 drop
 exit
        priority level 1
        queue-limit 8 packets
        set precedence $prec_value
        set cos 6
        aaa-accounting group $_acctgrp
      class voip-control
       police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop

        queue-limit $queue_size packets
        set precedence 6
        aaa-accounting group $_acctgrp
   policy-map $_incoming
      class voip
        police 200000 9216 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 5
        aaa-accounting group $_acctgrp
      class voip-control
       police 112000 21000 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 7
        aaa-accounting group $_acctgrp
}
```

Action Parameter Override

Action Parameter Override is a type of service template where shell variables are used in place of parameters for QoS actions such as police, shape, and bandwidth, configurations entered under a class in a QoS policy.

If you deactivate a template service, the system restores the previously active QoS policy. The QoS policy name may be different but is structurally and functionally identical to the QoS policy active before the template service was activated.

This example generates the service with the following parameters:

```
Reserved variable initialization before executing the service template shell function:
$_incoming = voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN
```

```
$_outgoing = voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT
$_acctgrp = aaa-accounting group
voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP list default
```

OUT QoS policy active on the session:

```
policy-map output_parent
 class class-default
   shape average 10000000
   service-policy output_child
policy-map output_child
class class-default
```

IN QoS policy active on the session:

```
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default
```

After you activate voice-service1(police_rate=200000,prec_value=5,queue_size=32) on the target session, this is the active OUT policy:

```
policy-map
output_parent$class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default

 class class-default
   shape average 10000000
   service-policy
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default
policy-map
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default

      class voip
        police 200000 60625 0 conform-action transmit exceed-action drop violate-action
drop
        priority level 1
        queue-limit 8 packets
        set precedence 5
        set cos 6
       aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP

      class voip-control
       police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop

        queue-limit 32 packets
        set precedence 6
       aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
class class-default
```

After you activate voice-service1(police_rate=200000,prec_value=5,queue_size=32) on the target session, this is the active IN policy:

```
policy-map
input_parent$class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default

 class class-default
   police cir 10000000 bc 312500 conform-action transmit exceed-action drop
   service-policy
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
```

```
policy-map
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
      class voip
        police 200000 9216 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 5
       aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP

      class voip-control
       police 112000 21000 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 7
       aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP

 class-default
```

Action Parameterization Default Parameters

Action Parameterization Default Paramters is a type of service template where shell variables are used in place of parameters for QoS actions such as police, shape, and bandwidth, configurations entered under a class in a QoS policy.

If you deactivate a template service, the system restores the previously active QoS policy. The QoS policy name maybe different but is structurally and functionally identical to the QoS policy active before the template service was activated.

OUT QoS policy active on the session:

```
policy-map output_parent
class class-default
   shape average 10000000
   service-policy output_child
policy-map output_child
class class-default
```

IN QoS policy active on the session:

```
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default
ip access-list extended voip-acl
 permit ip 10.1.1.0 0.0.0.255 any
ip access-list extended voip-control-acl
 permit ip 10.2.2.0 0.0.0.255 any
class-map match-any voip
 match access-group name voip-acl
!
class-map match-any voip-control
 match access-group name voip-control-acl
!
shell map voice-service1 police_rate=100000 prec_value=4 queue_size=1 in_h=class-default
out_h=class-default
{
    configure terminal
    accounting group $_acctgrp list default
    policy-map $_outgoing
      class voip
       police $police_rate 60625 0 conform-action transmit exceed-action drop violate-action
 drop
 exit
```

```
        priority level 1
        queue-limit 8 packets
        set precedence $prec_value
        set cos 6
        aaa-accounting group $_acctgrp
      class voip-control
       police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop

        queue-limit $queue_size packets
        set precedence 6
        aaa-accounting group $_acctgrp
   policy-map $_incoming
     class voip
       police 200000 9216 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 5
        aaa-accounting group $_acctgrp
      class voip-control
       police 112000 21000 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 7
        aaa-accounting group $_acctgrp
}
```

After you activate voice-service1 on the target session, this is the active OUT policy:

```
policy-map output_parent$class-default$voice-service1><_OUT$class-default class
  class-default
  shape average 10000000
  service-policy output_child$voice-service1><_OUT$class-default
policy-map output_child$voice-service1><_OUT$class-default
  class voip
   police 10000 60625 0 conform-action transmit exceed-action drop violate-action drop
   priority level 1
   queue-limit 8 packets
   set precedence 4
   set cos 6
   aaa-accounting group voice-service1><_GRP
  class voip-control
   police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop
   queue-limit 16 packets
   set precedence 6
   aaa-accounting group voice-service1><GRP
 class class-default
```

After you activate voice-service1 on the target session, this is the active IN policy:

```
policy-map input_parent$class-default$voice-service1><_IN$class-default
 class class-default
 police cir 10000000 bc 312500 conform-action transmit exceed-action drop
 service-policy input_child$voice-service1><_IN$class-default
policy-map input_child$voice-service1><_IN$class-default
 class voip
  police 200000 9216 0 conform-action transmit exceed-action transmit violate-action drop
  set precedence 5
  aaa-accounting group voice-service1><_GRP
 class voip-control
  police 112000 21000 0 conform-action transmit exceed-action transmit violate-action drop
  set precedence 7
  aaa-accounting group voice-service1><_GRP
 class-default
```

Class Name Override

Class name override is a type of service template where shell variables are used in place of parameters for QoS actions such as police, shape, and bandwidth, configurations entered under a class in a QoS policy. Shell variables may also be used in place of class names in service template policy definitions. Shell variables may completely substitute a class name or may be configured as a variable suffix with a constant prefix.

If you deactivate a template service, the system restores the previously active QoS policy. The QoS policy name may be different but is structurally and functionally identical to the QoS policy active before the template service was activated.

OUT QoS policy active on the session:

```
policy-map output_parent
 class class-default
   shape average 10000000
   service-policy output_child
policy-map output_child
class class-default
```

IN QoS policy active on the session:

```
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default
! Pre-configured ACLs/class-maps
ip access-list extended aol_classifier_acl         ! Locally pre-configured
 permit ip host 10.1.30.194 any
class-map match-all voice-control-aol_classifier_reference      ! Locally pre-configured
  match access-group name aol_classifier_acl
! Other pre-configured ACLs/classes here (e.g., voice-aol_classifier_reference,
voice-t_online, etc.)
! Service template:
shell map voice-aol-service1 prec_value=3 police_rate=100000 class_ref=t_online
in_h=class-default out_h=class-default
{
    configure terminal
    accounting group $_acctgrp list default
    policy-map $_outgoing
      class voice-control-$class_ref
       police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop

        queue-limit 16 packets
        set precedence 6
        aaa-accounting group $_acctgrp
      class voice-$class_ref
       police $poice_rate 60625 0 conform-action transmit exceed-action drop violate-action
 drop
        priority level 1
        queue-limit 8 packets
        set precedence $prec_value
        set cos 6
        aaa-accounting group $_acctgrp
   policy-map $_incoming
     class voice-control-$class_ref
      police 112000 21000 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 7
        aaa-accounting group $_acctgrp
      class voice-$class_ref
        police 200000 9216 0 conform-action transmit exceed-action transmit violate-action
```

```
 drop
        set precedence $prec_value
        aaa-accounting group $_acctgrp
}
```

After you activate voice-aol-service1(class_ref=aol_classifier_reference) on the target session, this is the active OUT policy:

```
policy-map
output_parent$class-default$voice-aol-service1<class_ref=aol_classifier_reference>_OUT$class-default

 class class-default
    shape average 10000000
   service-policy
output_child$voice-aol-service1<class_ref=aol_classifier_reference>_OUT$class-default
policy-map
output_child$voice-aol-service1<class_ref=aol_classifier_reference>_OUT$class-default
      class voice-control-aol_classifier_reference     ! Reference to pre-configured class
        police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop

        queue-limit 16 packets
        set precedence 6
        aaa-accounting group voice-aol-service1<class_ref=aol_classifier_reference>_GRP
      class voice-aol_classifier_reference     ! reference to pre-configured class
        police 100000 60625 0 conform-action transmit exceed-action drop violate-action
drop
        priority level 1
        queue-limit 8 packets
        set precedence 3
        set cos 6
        aaa-accounting group voice-aol-service1<class_ref=aol_classifier_reference>_GRP
class class-default
```

After you activate voice-aol-service1(class_ref=aol_classifier_reference) on the target session, this is the active IN policy:

```
policy-map
input_parent$class-default$voice-aol-service1<class_ref=aol_classifier_reference>_IN$class-default

 class class-default
    police cir 10000000 bc 312500 conform-action transmit exceed-action drop
   service-policy
input_child$voice-aol-service1<class_ref=aol_classifier_reference>_IN$class-default
policy-map input_child$voice-aol-service1<class_ref=aol_classifier_reference>_IN$class-default

      class voice-control-aol_classifier_reference     ! reference to pre-configured class
        police 112000 21000 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 7
        aaa-accounting group voice-aol-service1<class_ref=aol_classifier_reference>_GRP
      class voice-aol_classifier_reference     ! reference to pre-configured class
        police 200000 9216 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 3
        aaa-accounting group voice-aol-service1<class_ref=aol_classifier_reference>_GRP
       class-default
```

IP Address Parameterization

IP Address Parameterization is a type of Action Parameterization service template in which classifiers may be dynamically modified by adding more entries to ACLs. The entries to be added in an ACL are a list of IP addresses in a shell variable.

If you deactivate a template service, the system restores the previously active QoS policy. The QoS policy name may be different but is structurally and functionally identical to the QoS policy active before the template service was activated.

Note Classes must be predefined; they are not dynamically created.

OUT QoS policy active on the session:

```
policy-map output_parent
 class class-default
   shape average 10000000
   service-policy output_child
policy-map output_child
class class-default
```

IN QoS policy active on the session:

```
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default
! Base ACLs:
ip access-list extended IPOne-control-acl     ! Base ACL locally pre-configured
  permit ip any host 10.0.132.118
  permit ip host 10.0.132.118 any
  permit ip any host 10.1.245.122
  permit ip host 10.1.245.122 any
ip access-list extended IPOne-combined-acl      ! Base ACL pre-configured
 permit ip any 10.0.132.0 0.0.0.127
 permit ip 10.0.132.0 0.0.0.127 any
 permit ip any 10.1.245.64 0.0.0.63
 permit ip 10.1.245.64 0.0.0.63 any
! Base class-maps:
class-map match-any voice-control      ! Base class map pre-configured
  match access-list name IPOne-control-acl      !   Match on the base ACL
class-map match-any voice      ! base class-map pre-configured
  match access-list name IPOne-combined-acl      !   Match on the base ACL
! Service template:
shell map voice-toi  prec_value=3 police_rate=100000 ip_list=10.2.1.0/28,10.2.1.0/29
in_h=class-default out_h=class-default
{
    configure terminal
    ! Class-map templates:
    classmap-template voice-control  $ip_list
    classmap-template voice  $ip_list
    ! Service parameter templates:
    policy-map $_outgoing
      class voice-control-$ip_list     ! class names MUST end with -$ip_list
       police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop

        queue-limit 16 packets
        set precedence 6
        aaa-accounting group IPOne-aol
      class voice-$ip_list
       police $police_rate 60625 0 conform-action transmit exceed-action drop violate-action
 drop
        priority level 1
        queue-limit 8 packets
        set precedence $prec_value
```

```
        aaa-accounting group IPOne-aol
    policy-map $_incoming
      class voice-control-$ip_list
       police 112000 21000 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 7
        aaa-accounting group IPOne-aol
      class voice-$ip_list
        police 200000 9216 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence $prec_value
        aaa-accounting group IPOne-aol
```

After you activate voice-toi(ip_list=10.1.30.0/28,10.1.40.0/29) on the target session, this is the active OUT QoS policy :

```
policy-map output_parent$class-default$
voice-toi>ip_list=10.1.30.0/28,10.1.40.0/29<_OUT$class-default
 class class-default
    shape average 10000000
   service-policy output_child$voice-toi>ip_list=10.1.30.0/28,10.1.40.0/29<_OUT$class-default
policy-map output_child$voice-toi>ip_list=10.1.30.0/28,10.1.40.0/29<_OUT$class-default
      class voice-control-10.1.30.0/28,10.1.40.0/29
       police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop

        queue-limit 16 packets
        set precedence 6
        aaa-accounting group IPOne-aol
      class voice-10.1.30.0/28,10.1.40.0/29
        police 100000 60625 0 conform-action transmit exceed-action drop violate-action
drop
        priority level 1
        queue-limit 8 packets
        set precedence 3
        aaa-accounting group IPOne-aol
class class-default
```

After you activate voice-toi(ip_list=10.1.30.0/28,10.1.40.0/29) on the target session, this is the active IN QoS policy :

```
policy-map
input_parent$class-default$voice-toi>ip_list=10.1.30.0/28,10.1.40.0/29<_IN$class-default
 class class-default
    police cir 10000000 bc 312500 conform-action transmit exceed-action drop
   service-policy input_child$voice-toi>ip_list=10.1.30.0/28,10.1.40.0/29<_IN$class-default
policy-map input_child$voice-toi>ip_list=10.1.30.0/28,10.1.40.0/29<_IN$class-default
      class voice-control-10.1.30.0/28,10.1.40.0/29
       police 112000 21000 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 7
        aaa-accounting group IPOne-aol
      class voice-10.1.30.0/28,10.1.40.0/29
        police 200000 9216 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 3
        aaa-accounting group IPOne-aol
 class-default
```

Note The following configurations are dynamically created.

```
! Internally created ACLs:
ip access-list extended IPOne-control-acl-10.1.30.0/28,10.1.40.0/29
  permit ip any host 10.0.132.118
  permit ip host 10.0.132.118 any
  permit ip any host 10.1.245.122
  permit ip host 10.1.245.122 any
  permit ip 10.1.30.0 0.0.0.15 any  ! ACEs derived from $ip_list
  permit ip any  10.1.30.0 0.0.0.15
  permit ip 10.1.40.0 0.0.0.7 any
  permit ip any 10.1.40.0 0.0.0.7
ip access-list extended IPOne-combined-acl-10.1.30.0/28,10.1.40.0/29
  permit ip any 10.0.132.0 0.0.0.127
  permit ip 10.0.132.0 0.0.0.127 any
  permit ip any 10.1.245.64 0.0.0.63
  permit ip 10.1.245.64 0.0.0.63 any
  permit ip 10.1.30.0 0.0.0.15 any  ! ACEs derived from $ip_list
  permit ip any 10.1.30.0 0.0.0.15
  permit ip 10.1.40.0 0.0.0.7 any
  permit ip any 0.0.0.7 10.1.40.0
! internally created class-maps:
class-map match-any voice-control-10.1.30.0/28,10.1.40.0/29
   match access-group name IPOne-control-acl-10.1.30.0/28,10.1.40.0/29
class-map match-any voice-10.1.30.0/28,10.1.40.0/29
   match access-group name IPOne-combined-acl-10.1.30.0/28,10.1.40.0/29
```

Turbo Button Service

Turbo Button service is a type of Action Parameterization service template in which only policy parameters in the INPUT parent class-default and shape parameters in the OUT parent class-default can be dynamically modified.

This example shows how to create a service template for the Turbo Button service:

OUT QoS policy active on the session:

```
policy-map output_parent
 class class-default
   shape average 10000000
   service-policy output_child
policy-map output_child
class class-default
```

IN QoS policy active on the session:

```
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default
```

```
shell map turbo-button in_police_val=20000000 $out_shape=20000000
configure terminal
accounting group $_acctgrp list default
policy-map $_outgoing
class class-default
shape average $out_shape
aaa-accounting group $_acctgrp
policy-map $_incoming
class class-default
police $in_police_val
aaa-accounting group $_acctgrp
```

Turbo Button Activation

This example shows how to activate the Turbo Button service using the default values.

OUT QoS policy active on the session:

```
policy-map output_parent
 class class-default
   shape average 10000000
   service-policy output_child
policy-map output_child
class class-default
```

IN QoS policy active on the session:

```
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default
```

```
accounting group turbo-button>< list default

accounting group turbo-button>< list default
! Service outgoing:
 policy-map  turbo-button><_OUT
 class class-default
 shape average 20000000
 aaa-accounting group turbo-button>< list default
! Service incoming:
 policy-map  turbo-button><_IN
 class class-default
 police 20000000
aaa-accounting group turbo-button>< list default
```

After you activate the service on the target session, this is the active OUT policy:

```
policy-map output_parent$turbo-button><_OUT$
class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default

 class class-default
 shape average 20000000
aaa-accounting group turbo-button>< list default
service-policy
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default
policy-map
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default
class voip
police 200000 60625 0 conform-action transmit exceed-action drop violate-action drop
 priority level 1
 queue-limit 8 packets
 set precedence 5
 set cos 6

aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
 class voip-control
 police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop
 queue-limit 32 packets
 set precedence 6
aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
class class-default
```

After you activate the service on the target session, this is the active IN policy:

```
policy-map input_parent$turbo-button>
<_IN$class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
class class-default
   police cir 20000000 bc 312500 conform-action transmit exceed-action drop
   aaa-accounting group turbo-button>< list default

service-policy
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
policy-map
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default

   class voip
        police 200000 9216 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 5
       aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP

   class voip-control
        police 112000 21000 0 conform-action transmit exceed-action transmit violate-action
 drop
        set precedence 7
       aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
class-default
```

Turbo Button Deactivation

This example shows how to deactivate the Turbo Button service using the default values of VSA 252 0c turbo-button().

OUT QoS policy active on the session:

```
policy-map output_parent
 class class-default
   shape average 10000000
   service-policy output_child
policy-map output_child
class class-default
```

IN QoS policy active on the session:

```
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default
```

After you activate the service on the target session, this is the active OUT policy:

```
policy-map
output_parent$class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default
class class-default
  shape average 10000000
  service-policy
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default
policy-map
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default

  class voip
   police 200000 60625 0 conform-action transmit exceed-action drop violate-action drop
   priority level 1
   queue-limit 8 packets
   set precedence 5
   set cos 6
   aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
```

```
  class voip-control
   police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop
   queue-limit 32 packets
   set precedence 6
   aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP

class class-default
```

After you activate the service on the target session, this is the active IN policy:

```
policy-map
input_parent$class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default

   class class-default
   police cir 10000000 bc 312500 conform-action transmit exceed-action drop
   service-policy
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default

policy-map
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
  class voip
   police 200000 9216 0 conform-action transmit exceed-action transmit violate-action drop
   set precedence 5
   aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
  class voip-control
   police 112000 21000 0 conform-action transmit exceed-action transmit violate-action drop

   set precedence 7
   aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
  class-default
```

Turbo Button Override

This example shows how to activate the Turbo Button service using the default values of VSA 250 Aturbo-button(in_police_val=30000000, out_shape_val=30000000) (Activation from Access-Accept) or VSA 252 0b turbo-button(in_police_val=30000000, out_shape_val=30000000) (Activation from CoA).

OUT QoS policy active on the session:

```
policy-map output_parent
 class class-default
   shape average 10000000
   service-policy output_child
policy-map output_child
class class-default
```

IN QoS policy active on the session:

```
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default

accounting group turbo-button>in_police_val=30000000#out_shape_val=30000000 list default

! Service outgoing:
 policy-map turbo-button>in_police_val=30000000#out_shape_val=30000000<_OUT
 class class-default
  shape average 30000000
  accounting group turbo-button>in_police_val=30000000#out_shape_val=30000000
```

```
! Service incoming:
  policy-map turbo-button>in_police_val=30000000#out_shape_val=30000000<_IN
  class class-default
   police 30000000
   accounting group turbo-button>in_police_val=30000000#out_shape_val=30000000
```

After you activate the service on the target session, this is the active OUT policy:

```
policy-map output_parent$turbo-button>
in_police_val=30000000#out_shape_val=30000000<_OUT$class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default
class class-default
 shape average 20000000
 accounting group turbo-button>in_police_val=30000000#out_shape_val=30000000
 service-policy
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default

 policy-map
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default
class voip
 police 200000 60625 0 conform-action transmit exceed-action drop violate-action drop
 priority level 1
 queue-limit 8 packets
 set precedence 5
 set cos 6
 aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
class voip-control
 police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop
 queue-limit 32 packets
 aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
class class-default
```

After you activate the service on the target session, this is the active IN policy:

```
policy-map
input_parent$turbo-button>in_police_val=30000000#out_shape_val=30000000<_IN$class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
class class-default
 police cir 20000000 bc 312500 conform-action transmit exceed-action drop
 accounting group turbo-button>in_police_val=30000000#out_shape_val=30000000
 service-policy
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
policy-map
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
 class voip
  police 200000 9216 0 conform-action transmit exceed-action transmit violate-action drop
  set precedence 5
  aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
 class voip-control
  police 112000 21000 0 conform-action transmit exceed-action transmit violate-action drop
  set precedence 7
  aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
 class-default
```

Example Turbo Button Override Deactivation

This example shows how to deactivate the Turbo Button override using the default values of VSA 252 0c turbo-button (in_police_val=30000000, out_shape_val=30000000).

OUT QoS policy active on the session:

```
policy-map output_parent
 class class-default
   shape average 10000000
   service-policy output_child
```

```
policy-map output_child
class class-default
```

IN QoS policy active on the session:

```
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default
```

After you activate the service on the target session, this is the active OUT policy:

```
policy-map
output_parent$class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default


 class class-default
  shape average 10000000
  service-policy
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default
policy-map
output_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_OUT$class-default

  class voip
   police 200000 60625 0 conform-action transmit exceed-action drop violate-action drop
   priority level 1
   queue-limit 8 packets
   set precedence 5
   set cos 6
   aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
  class voip-control
   police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop
   queue-limit 32 packets
   set precedence 6
   aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
  class class-default
```

After you activate the service on the target session, this is the active IN policy:

```
policy-map
input_parent$class-default$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default

  class class-default
  police cir 10000000 bc 312500 conform-action transmit exceed-action drop
  service-policy
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
  policy-map
input_child$voice-service1>police_rate=200000#prec_value=5#queue_size=32<_IN$class-default
  class voip
   police 200000 9216 0 conform-action transmit exceed-action transmit violate-action drop
   set precedence 5
   aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
  class voip-control
   police 112000 21000 0 conform-action transmit exceed-action transmit violate-action drop

   set precedence 7
   aaa-accounting group voice-service1>police_rate=200000#prec_value=5#queue_size=32<_GRP
  class-default
```

Example Overriding Interim Accounting Interval

Overriding Interim Accounting Interval is a type of Action Parameterization service template in which you can use the shell variables in place of interim interval values in the accounting method list definition, allowing the account interim value to be dynamically modified.

This example shows how to do an accounting group override using the default values of: VSA 252 0b voice-service1(police_rate=200000,prec_value=5,acct_interval=600).

This example generates a service with the following parameters:

```
! Global AAA method list and accounting group parameters
aaa accounting network list-600
 action-type start-stop periodic interval 600
  accounting group voice-service1>policy_rate=200000#prec_value=5#acct_interval=600 <_GRP
list list-600
! OUT policy-map:
 policy-map voice-service1>policy_rate=200000#prec_value-5#acct_interval=600 <_OUT
  class voip
   police 200000 60625 0 conform-action transmit exceed-action drop violate-action drop
   priority level 1
   queue-limit 8 packets
   set precedence 5
   set cos 6
   aaa-accounting group voice-service1>policy_rate=200000#prec_value=5#acct_interval=600
<_GRP
  class voip-control
   police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop
   queue-limit 32 packets
   set precedence 6
   aaa-accounting group



OUT:
policy-map output_parent
 class class-default
   shape average 10000000
   service-policy output_child
policy-map output_child
class class-default
IN:
policy-map input_parent
 class class-default
   police 10000000
   service-policy input_child
policy-map input_child
 class-default
```

After you activate the service on the target session, this is the active OUT policy:

```
policy-map
output_parent$class-default$voice-service1>policy_rate=200000#prev_value=5#acct_interval=600
 <_OUT$class-default
 class class-default
 shape average 10000000
 service-policy output_child$voice-service1>policy_rate=200000#prev_value=5#acct_interval=600
 <_OUT$class-default
 policy-map output_child$voice-service1>policy_rate=200000#prev_value=5#acct_interval=600
<_OUT$class-default
 class voip
  police 200000 60625 0 conform-action transmit exceed-action drop violate-action drop
  priority level 1
  queue-limit 8 packets
  set precedence 5
```

```
  set cos 6
  aaa-accounting group voice-service1>policy_rate=200000#prec_value=5#acct_interval=600
<_GRP
  class voip-control
  police 112000 1000 0 conform-action transmit exceed-action drop violate-action drop
  queue-limit 32 packets
  set precedence 6
  aaa-accounting group voice-service1>policy_rate=200000#prec_value=5#acct_interval=600
<_GRP
 class class-default
```

After you activate the service on the target session, this is the active IN policy:

```
policy-map
input_parent$class-default$voice-service1>policy_rate=200000#prec_value-5#acct_interval=600
 <_IN$class-default
class class-default
 police cir 10000000 bc 312500 conform-action transmit exceed-action drop
 service-policy input_child$voice-service1>policy_rate=200000#prec_value=5#acct_interval=600
 <_IN$class-default
policy-map input_child$voice-service1>policy_rate=200000#prec_value=5#acct_interval=600
<_IN$class-default
 class voip
  police 200000 9216 0 conform-action transmit exceed-action transmit violate-action drop
  set precedence 5
  aaa-accounting group voice-service1>policy_rate=200000#prec_value=5#acct_interval=600
<_GRP
 class voip-control
  police 112000 21000 0 conform-action transmit exceed-action transmit violate-action drop
  set precedence 7
  aaa-accounting group voice-service1>policy_rate=200000#prec_value=5#acct_interval=600
<_GRP
 class class-default
```

Subscriber Accounting Accuracy

The Subscriber Accounting Accuracy feature guarantees that the I/O packet/byte statistics in the Accounting-Stop record are accurate to within one second.

Subscriber accounting data is sent to authentication, authorization, and accounting (AAA) servers during the following events:

- Configured intervals during the lifetime of the session or service
- Service logoff
- Session tear down

Use the **subscriber accounting accuracy** *milliseconds* command to set the value for the Subscriber Accounting Accuracy feature.

Change of Authorization (CoA) ACK Ordering

CoA ACK ordering sends a CoA-ACK for each CoA event before a QoS accounting record is sent for that CoA. A CoA may contain activation or deactivation of single or multiple services.

If a service fails to install on a session the following happens:

- The entire CoA fails.
- The Policy Manager sends a CoA-NAK to the RADIUS server.
- The previous service configuration is restored

If one or more services install before a failure is detected the following happens:

- The entire CoA fails.
- Services are backed out.
- The Policy Manager sends a CoA-NAK to the RADIUS server.
- The previous service configuration is restored.

Multiservice CoAs can compose up of either of the following:

- QoS services—The Policy Manager combines the services into one net-effect policy map. Only one QoS policy is applied to the session for all services. If the policy fails to install, the system restores the session to use the previous policy map. In effect the session is restored to the state prior to the CoA.
- QoS and Intelligent Services Gateway (ISG) services—The Policy Manager applies the ISG service first, then the QoS service. If the QoS policy fails to install, the system restores the session to the previous policy map. Both the ISG and QoS service are rolled back to the previous state.

For multiservice CoA only one CoA-ACK is sent when all services successfully install.

Change of Authorization Rollback

The CoA Rollback feature restores QoS policy accounting to its state before the CoAs were issued. CoA Rollback also properly acknowledges the RADIUS server using a CoA-NAK.

The CoA Rollback feature applies to syntax mistakes and policy install failures such as admission control and resource allocation failure.

If CoA fails, the system sends a CoA-NAK and does not send QoS accounting records. The accounting record for existing services keeps previous counters and continues to count new packets.

QoS Accounting High Availability

When QoS accounting is enabled in a class the policy accounting feature supports three types of events:

- Start—Indicates a new accounting flow. The start record contains statistics and attributes specific to this flow.
- Interim—Indicates how often flow statistics are reported.
- Stop—Indicates the end of an accounting flow. The stop record also contains statistics and attributes specific to this flow.

The policy accounting feature collects the statistics for the accounting flows and sends the information to the RADIUS accounting billing server.

The QoS accounting high availability feature ensures that the start, interim, and stop accounting records are not affected if a planned or unexpected failover occurs. When a planned or unexpected failover occurs the

QoS accounting HA feature ensures that the RP switchover occurs without interrupting the flow of information to the RADIUS accounting billing server. The feature also ensures that all QoS services on all active sessions continue without any interruption and that the service accounting counters persist across the RP switchover.

Persistence of Policy Accounting States

To ensure that start, stop, and interim accounting is not affected by a stateful switchover (SSO) or an in-service software upgrade (ISSU), the Policy Manager synchronizes all QoS services and parameterized CoA functionality with the standby RP at the time of the failover. In addition, the dynamic QoS configurations and the polling interval are synchronized between the active and standby RPs.

To synchronize a parameterized CoA event to a standby RP, the Policy Manager performs the following functions:

- Manages the CoA replay to synchronize provisioning events on the standby RP.
- Uses the same service template on both the active and standby RP.
- Creates the same policy map and class map names to apply to the session on both the active ad standby RP.
- Uses predefined QoS policy maps and class maps during service template activation.

Persistence of Policy Accounting Counters

The QoS Accounting HA feature ensures that the policy accounting counters persist across an SSO or failover. After a switchover occurs, the standby RP becomes the active RP and accumulates the statistics from the previously active RP. If the newly active RP receives a periodic update after the switchover it generates an interim record using the statistics it accumulated plus the values from the periodic update. If the newly active RP does not receive a periodic update after the switchover, it generates the interim record using only the statistics it accumulated from the previously active RP.

For more information on SSOs and ISSUs, see the *Cisco IOS High Availability Configuration Guide*.

How to Use QoS Policy Accounting

To use QoS Policy Accounting you must assign a group or AAA method list to a traffic class, then you configure the service template for policy accounting, and finally you activate the subscriber accounting accuracy functionality.

Note By default, QoS Policy Accounting is not assigned to traffic classes.

Assigning a Group or AAA Method List to a Traffic Class

Before you begin

Ensure the group or AAA method list already exists. If you try to add an undefined group or AAA method list to a traffic class, you will receive an error message.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **aaa authentication ppp** *list-name method1*
4. **aaa accounting network** *methodlist-name*
5. **action-type start-stop**
6. **periodic interval** *minutes*
7. **accounting group group_name** *list list-name*
8. **policy-map** *policy-map-name*
9. **class class-default**
10. **accounting aaa list** *list-name* [*group-name*]
11. **end**
12. **show policy-map session**
13. **show accounting group** *group-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**aaa authentication ppp** *list-name method1* **Example:** `Router(config)# aaa authentication ppp group radius`	Specifies a valid AAA authentication method. • Group RADIUS enables global RADIUS authentication.
Step 4	**aaa accounting network** *methodlist-name* **Example:** `Router(config)# aaa accounting network list1`	Enables AAA of services when you use RADIUS. • The algorithm determining the interim interval for a class or group uses the method list specified here.
Step 5	**action-type start-stop** **Example:** `Router(config)# action-type start-stop`	Sends a start accounting notice at the beginning of a process and a stop accounting notice at the end of a process.
Step 6	**periodic interval** *minutes* **Example:**	Adds the interim interval value (1 to 71,582 minutes) in the method list, if specified.

	Command or Action	Purpose
	`Router(config)# periodic interval 1`	• If you do not define an interim interval, the global value defined by AAA is used. • If the method list disables interim updates, the accounting flows using the method list do not generate an interim update.
Step 7	**accounting group group_name** *list list-name* **Example:** `Router(config)# accounting group group_name AAAmethodlist AAAmethodlist1`	Sets properties in the AAA method list. • You can make per-session changes to existing traffic classes by temporarily overwriting properties in the groups or AAA method lists to which they are assigned. This allows you to provide dynamic customized QoS configuration to each subscriber.
Step 8	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map p1`	Creates a policy map.
Step 9	**class class-default** **Example:** `Router(config)# class class-default`	Creates a traffic class.
Step 10	**accounting aaa list** *list-name* [*group-name*] **Example:** `Router(config)# accounting aaa list AAAmethodlist1`	Assigns the traffic class to a group or an AAA method list. • This example shows the QoS Policy Accounting feature enabled for instances of a traffic class using list AAAmethodlist1 with no group.
Step 11	**end** **Example:** `Router(config)# end`	Exits interface configuration mode and returns to privileged EXEC mode.
Step 12	**show policy-map session** **Example:** `Router# show policy-map session`	(Optional) Displays QoS Policy Accounting feature information for traffic classes with a group or an AAA method list.
Step 13	**show accounting group** *group-name* **Example:** `Router# show accounting group acc-group1`	(Optional) Displays all group-to-method list associations. • Enter a group name to view information specific to that group.

Activating Subscriber Accounting Accuracy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **subscriber accounting accuracy** *milliseconds*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**subscriber accounting accuracy** *milliseconds* **Example:** `Device(config)# subscriber accounting accuracy 1000`	Sets the value for the Subscriber Accounting Accuracy feature.
Step 4	**end** **Example:** `Device(config)# end`	Enters privileged EXEC mode.

Troubleshooting Service Templates

To troubleshoot any service template issues, you can display usage information for all template service policy maps on your router.

SUMMARY STEPS

1. **enable**
2. **show subscriber policy ppm-shim-db**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**show subscriber policy ppm-shim-db** **Example:** `Router(config)# show subscriber policy ppm-shim-db`	Displays reference counts (usage) of all template service policy-maps and Net Effect policy-maps on the router.

Configuration Examples for QoS Policy Accounting

Example: Using the QoS Policy Accounting Feature in Groups

The following example shows grouping:

```
policy-map my-policy
class voip
police
aaa-accounting group premium-services
class voip-control
police
aaa-accounting group premium-services
```

Example: Generating Separate Accounting Streams

The following example shows two classifiers called class voip and class voip-control. The classifiers are assigned to one policy associated with one target. This configuration generates two separate QoS policy accounting streams.

```
policy-map my-policy
class voip
police 200000
accounting aaa list AAA-LIST
class voip-control
police 100000
accounting aaa list AAA-LIST
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands	*Cisco IOS QoS Command Reference*

Related Topic	Document Title
Cisco IOS High Availability	*Cisco IOS High Availability Configuration Guide*

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2866	RADIUS Accounting

Technical Assistance

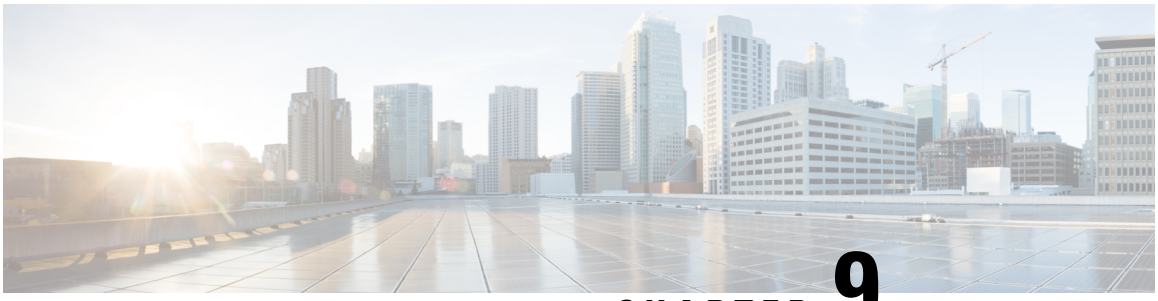
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for the QoS Policy Accounting Feature

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 10: Feature Information for the QoS Policy Accounting Feature



CHAPTER 9

PPP Session Queueing on ATM VCs

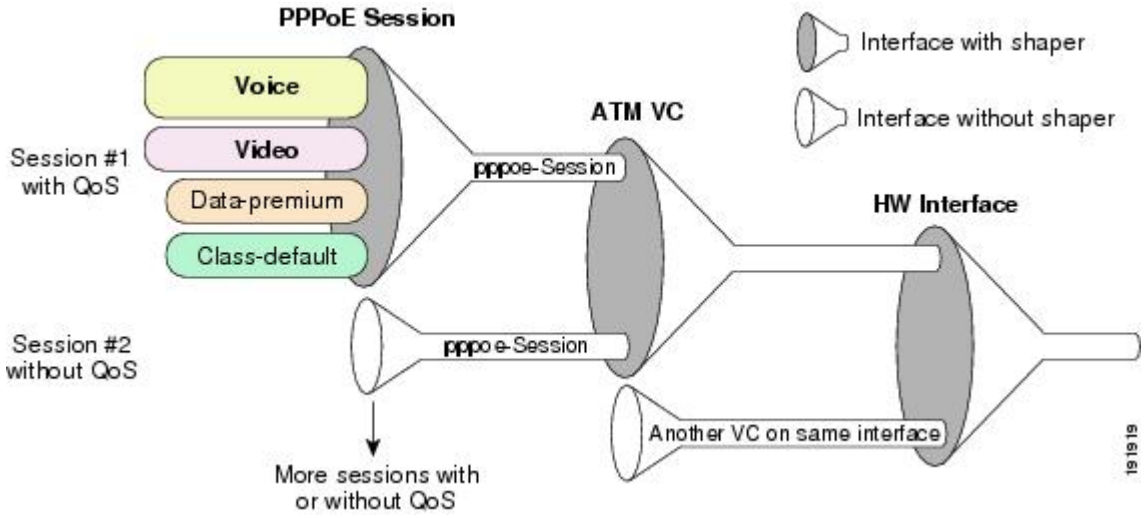
The PPP Session Queueing on ATM VCs feature enables you to shape and queue PPP over Ethernet over ATM (PPPoEoA) sessions to a user-specified rate. Multiple sessions can exist on any ATM VC and have Quality of Service (QoS) policies applied, or some of the sessions might have QoS policies. The router shapes the sum of allbandwidth used for PPPoEoA traffic on a VC so that the subscriber's connection to the Digital Subscriber Line Access Multiplexer (DSLAM) does not become congested. Queueing-related functionality provides different levels of service to the various applications that run over the PPPoEoA session.

A nested, two-level hierarchical service policy is used to configure session shaping directly on the router using the modular quality of service command-line interface (MQC). The hierarchical policy consists of the following:

- Child policy--Defines QoS actions using QoS commands such as the priority, bandwidth, and police commands.
- Parent policy--Contains only the class-default class with the shape or bandwidth remaining ratio command configured, or with both commands configured:
  - shape command--Shapes the session traffic to the specified bit rate, according to a specific algorithm.
  - bandwidth remaining ratio command--Specifies a ratio value that the router uses to determine how much unused bandwidth to allocate to the session during congestion.

Note The PPP Session Queueing on ATM VCs feature works with both PPP terminated aggregation (PTA) and L2TP access concentrator (LAC) configurations.

The figure below illustrates PPP session Queueing on ATM VCs.



Prerequisites for PPP Session Queueing on ATM VCs

- PPPoEoA sessions must be enabled.
- Create traffic classes using the class-map command and specify the match criteria used to classify traffic.
- For dynamic PPPoEoA session queueing using RADIUS, you must:
  - Enable authentication, authorization, and accounting (AAA) on the router
  - Configure the RADIUS server for dynamic QoS
  - Create the subscriber's user profile on the RADIUS server

Restrictions for PPP Session Queueing on ATM VCs

- You cannot configure PPP session queueing on unshaped VCs--VCs without a specified peak cell rate (PCR) or sustained cell rate (SCR).
- VCs with session queueing polices cannot be part of a shaped virtual path (VP).
- If the same ATM category (for example, shaped unspecified bit rate (UBR)) contains both high and low bandwidth VCs, the SAR mechanism can cause low throughput for high bandwidth VCs. The workaround is to use different ATM classes for low and high bandwidth VCs. For example, configure low bandwidth VCs as shaped UBR and high bandwidth VCs as variable bit rate-nonreal-time (VBR-nrt) or constant bit rate (CBR).

- The CLASS-BASED QOS MIB does not include statistics for service policies applied to sessions.
- RADIUS accounting does not include queueing statistics.

Information About PPP Session Queueing on ATM VCs

Dynamically Applying QoS Policies to PPP Sessions on ATM VCs

The router allows you to dynamically apply QoS policy maps to PPPoEoA sessions using RADIUS. Although the actual configuration of the QoS policies occurs on the router, you can configure the following attribute-value (AV) pairs on RADIUS to specify the name of the policy map to dynamically apply to the session:

```
"ip:sub-qos-policy-in=<name of the QoS policy in ingress direction>"
"ip:sub-qos-policy-out=<name of egress policy>"
```

You define the AV pairs in one of the following RADIUS profiles:

- User profile--The user profile on the RADIUS server contains an entry that identifies the policy map name applicable to the user. The policy map name is the service that RADIUS downloads to the router after a session is authorized.
- Service profile--The service profile on the RADIUS server specifies a session identifier and an AV pair. The session identifier might be, for example, the IP address of the session. The AV pair defines the service (policy map name) to which the user belongs.

After receiving a service-logon request from the policy server, RADIUS sends a change of authorization (CoA) request to the router to activate the service for the subscriber, who is already logged in. If the authorization succeeds, the router downloads the name of the policy map from RADIUS using the ip:sub-qos-policy-in[out]= AV-pair and applies the QoS policy to the PPPoEoA session. Because the service policy contains queueing-related actions, the router sets up the appropriate class queues.

Note Although the router also supports the RADIUS vendor specific attribute (VSA) 38, Cisco-Policy-Down and Cisco-Policy-Up, we recommend that you use the ip:sub-qos-policy-in[out]= AV pairs for QoS policy definitions.

PPP Session Queueing Inheritance

PPP Sessions either inherit queues from their parent interface or they have their own queues. Each PPPoEoA session for which session queueing is configured has its own set of queues.

The table below describes the queues to which the router directs session traffic.

Table 11: PPP Session Queue Inheritance

Queueing Policy	Queue Used for Session Traffic
No policy	VC default queue

Queueing Policy	Queue Used for Session Traffic
Applied to the VC	VC queues
Applied to the session	Session queues

Interfaces Supporting PPP Session Queueing

The router supports PPP session queueing on shaped ATM VCs for outbound traffic only.

The router does not support PPP session queueing on inbound ATM interfaces.

Mixed Configurations and Queueing

A mixed configuration is one in which all sessions do not have QoS applied to them. On some VCs, the queueing policy is applied at the VC level, and on other VCs the queueing policies are applied on the sessions. Some sessions have no policy applied at all. As a result, the router uses the hierarchical queueing framework (HQF) to direct traffic in the following ways:

- If no queueing policy is applied at the VC or session level, the router sends all traffic on the VC to the default queue, including traffic from sessions on the VC that have a policing-only policy applied or no policy applied.
- If a queueing policy is applied at the VC level, but not at the session level, the router sends traffic to the queues associated with the queueing policy on the VC.
- If queueing policies are applied to some sessions on a VC but not to other sessions, the router sends the traffic with a policing-only policy or with no policy applied to the VC's default queue. The router sends traffic with queueing policies to the queues associated with the queueing policy applied to the session.

Bandwidth Mode and ATM Port Oversubscription

An ATM port can operate in reserved bandwidth mode or shared bandwidth mode.

When a port is not oversubscribed (the sum of the bandwidths of all VCs on the port is less than the port bandwidth), the port operates in reserved bandwidth mode--a specific amount of bandwidth is reserved for each VC on the port. If a VC does not use all of its allocated bandwidth, the unused bandwidth is not shared among the VCs on the port.

When the ATM port is oversubscribed (the sum of the bandwidths of all VCs on the port is greater than the port bandwidth), the port operates in shared bandwidth mode. In this mode, any unused bandwidth is available for reuse by the other VCs on the port, up to the VC's respective shape rate--traffic on a VC cannot exceed the shape rate of that VC.

Oversubscription at the Session Level

Oversubscription at the session level occurs after session traffic shaping and when the aggregate session traffic exceeds the subinterface shape rate. After all priority traffic is accounted for, the router distributes the remaining bandwidth on the VC to the sessions according to the value specified in the bandwidth remaining ratio command configured in the parent policy of the policy applied to the sessions. If the bandwidth remaining ratio command is not specified in the parent policy, the router uses a default ratio of 1.

How to Configure PPP Session Queueing on ATM VCs

Configuring PPP Session Queueing Using a Virtual Template

A virtual templat e is a logical interface whose configuration can specify generic configuration information for a specific purpose, user-specific configuration information, and router-dependent information. You configure a virtual template on an interface and apply QoS policy maps to the virtual template. The virtual template inherits the QoS features specified in the policy map. When the router establishes sessions on an interface, the router applies the QoS features specified in the virtual template configuration to the virtual access interfaces (VAIs) created for the sessions, including the QoS features specified in the policy map attached to the virtual template.

A broadband aggregation group (bba-group) configured on an ATM interface points to the virtual template the router uses to apply QoS policies to sessions. When a session arrives on an ATM interface, the router creates a virtual access interface (VAI) for the session and applies the policies associated with the virtual template to the sessions.

To configure PPPoEoA session queueing using a virtual template, perform the following configuration tasks:

Configuring an Hierarchical QoS Policy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-map-name*
5. priority level level
6. **police** *bps* [*burst-normal burst-max*] [**conform-action** *action]* [**exceed-action** *action]* **violate-action** *action*
7. set cos value
8. bandwidth remaining ratio
9. exit
10. **policy-map** *policy-map-name*
11. **class** *class-default*
12. bandwidth remaining ratio
13. **shape** [**average**] *mean-rate*[*burst-size*] [*excess-burst-size*]
14. **service-policy** *policy-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy-map-name`	Creates or modifies the child policy. Enters policy-map configuration mode. policy-map-name is the name of the child policy map.
Step 4	**class** *class-map-name* **Example:** `Router(config-pmap)# class class-map-name`	Assigns the traffic class you specify to the policy map. Enters policy-map class configuration mode. class-map-name is the name of a previously configured class map and is the traffic class for which you want to define QoS actions. Repeat Steps 2 through 6 for each traffic class you want to include in the child policy map.
Step 5	priority level level **Example:** `Router(config-pmap-c)# priority level level`	(Optional) Defines multiple levels of a strict priority service model. When you enable a traffic class with a specific level of priority service, the implication is a single priority queue associated with all traffic enabled with the specified level of priority service. level is a number that indicates a specific priority level. Valid values are from 1 (high priority) to 4 (low priority). Default: 1
Step 6	**police** *bps* [*burst-normal burst-max*] [**conform-action** *action]* [**exceed-action** *action]* **violate-action** *action* **Example:** `Router(config-pmap-c)# police bps [burst-normal] [burst-max] [conform-action action] [exceed-action action] [violate-action action]`	(Optional) Configures traffic policing. bps is the average rate in bits per second. Valid values are 8000 to 200000000. (Optional) burst-normal is the normal burst size in bytes. Valid values are 1000 to 51200000. The default normal burst size is 1500 bytes. (Optional) burst-max is the excess burst size in bytes. Valid values are 1000 to 51200000. (Optional) conform-action action indicates the action to take on packets that conform to the rate limit. (Optional) exceed-action action indicates the action to take on packets that exceed the rate limit. (Optional) violate-action action indicates the action to take on packets that violate the normal and maximum burst sizes.
Step 7	set cos value **Example:**	(Optional) Sets the Layer 2 class of service (CoS) value of an outgoing packet.

	Command or Action	Purpose
	`Router(config-pmap-c)# set cos value`	value is a specific IEEE 802.1Q CoS value from 0 to 7.
Step 8	bandwidth remaining ratio **Example:** `Router(config-pmap-c)# bandwidth remaining ratio`	(Optional) Specifies a bandwidth-remaining ratio for class-level or subinterface-level queues to be used during congestion to determine the amount of excess bandwidth (unused by priority traffic) to allocate to non-priority queues. ratio specifies the relative weight of this subinterface or queue with respect to other subinterfaces or queues. Valid values are from 1 to 1000.
Step 9	exit **Example:** `Router(config-pmap-c)# exit`	Exits policy-map class configuration mode.
Step 10	**policy-map** *policy-map-name* **Example:** `Router(config-pmap)# policy-map policy-map-name`	Creates or modifies the parent policy. policy-map-name is the name of the parent policy map.
Step 11	**class** *class-default* **Example:** `Router(config-pmap)# class class-default`	Configures or modifies the parent class-default class. You can configure only the class-default class in a parent policy. Do not configure any other traffic class.
Step 12	bandwidth remaining ratio **Example:** `Router(config-pmap-c)# bandwidth remaining ratio`	(Optional) Specifies a bandwidth-remaining ratio for class-level or subinterface-level queues to be used during congestion to determine the amount of excess bandwidth (unused by priority traffic) to allocate to non-priority queues. ratio specifies the relative weight of this subinterface or queue with respect to other subinterfaces or queues. Valid values are from 1 to 1000.
Step 13	**shape** [**average**] *mean-rate*[*burst-size*] [*excess-burst-size*] **Example:** `Router(config-pmap-c)# shape [average] mean-rate [burst-size] [excess-burst-size]`	Shapes traffic to the indicated bit rate and enables ATM overhead accounting. (Optional) average is the committed burst (Bc) that specifies the maximum number of bits sent out in each interval. This option is only supported on the PRE3. mean-rate is also called committed information rate (CIR). Indicates the bit rate used to shape the traffic, in bits per second. When this command is used with backward explicit congestion notification (BECN) approximation, the bit rate is the upper bound of the range of bit rates that are permitted.

	Command or Action	Purpose
		(Optional) burst-size is the number of bits in a measurement interval (Bc). (Optional) excess-burst-size is the acceptable number of bits permitted to go over the Be.
Step 14	**service-policy** *policy-map-name* **Example:** `Router(config-pmap-c)# service-policy policy-map-name`	Applies the child policy to the parent class-default class. policy-map-name is the name of the child policy map configured in step 1.

Example

The following example shows how to configure a hierarchical QoS policy. In the example, the child-policy configures QoS features for two traffic classes: Premium and Silver. Premium traffic has priority and is policed at 40 percent. The router sets the IP precedence of Premium traffic to precedence level 3. Silver traffic is policed at 80000 bps and IP precedence level 3 is set. The child-policy is applied to the Parent policy class-default class, which shapes traffic to 200,000 Kbps.

```
Router(config)# policy-map child-policy
Router(config-pmap)# class Premium
Router(config-pmap-c)# priority
Router(config-pmap-c)# police percent 40
Router(config-pmap-c)# set ip precedence 3
Router(config-pmap-c)# class Silver
Router(config-pmap-c)# police 80000 10000 conform-action transmit exceed-action drop
Router(config-pmap-c)# set ip precedence 5
Router(config-pmap-c)# exit
Router(config-pmap)# policy-map Parent
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape 200000
Router(config-pmap-c)# service-policy output child-policy
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)#
```

Associating the Hierarchical Policy Map with a Virtual Template

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface virtual-template template-** *number*
4. **service-policy {input | output} policy-map-name**
5. exit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface virtual-template template-** *number* **Example:** `Router(config)# interface virtual-template template-number`	Creates a virtual template and enters interface configuration mode. template-number is the number you assign to the virtual template interface to identify it. Valid values are from 1 to 200. You can configure up to 200 virtual template interfaces on the router.
Step 4	**service-policy {input | output} policy-map-name** **Example:** `Router(config-if)# service-policy {input | output} policy-map-name`	Attaches the policy map you specify to the virtual template interface in the inbound or outbound direction that you specify. input specifies to apply the policy map to inbound traffic. output specifies to apply the policy map to outbound traffic. policy-map-name is the name of a previously configured policy map.
Step 5	exit **Example:** `Router(config-if)# exit`	Exits interface configuration mode.

Example

The following example shows how to associate a policy map with a virtual template. In this example, the policy map named Parent is associated with the virtual template named VirtualTemplate1.

```
Router(config)# interface virtual-template1
Router(config-if)# service-policy output Parent
Router(config-if)# exit
Router(config)#
```

Applying the Virtual Template to an ATM Subinterface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **bba-group pppoe group-name**
4. **virtual-template template-number**
5. exit
6. interface atm number.subinterface [point-to-point]
7. pvc [name] vpi/vci
8. protocol pppoe group group-name
9. exit
10. exit

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**bba-group pppoe group-name** **Example:** `Router(config)# bba-group pppoe group-name`	Creates a PPP over Ethernet (PPPoE) profile. Enters BBA group configuration mode. group-name is the name of the PPPoE profile.
Step 4	**virtual-template template-number** **Example:** `Router(config-bba-grp)# virtual-template template-number`	Associates a BBA group to the virtual template to be used for cloning virtual access interfaces. template-number is the identifying number of the virtual template.
Step 5	exit **Example:** `Router(config-bba-grp)# exit`	Exits BBA group configuration mode.
Step 6	interface atm number.subinterface [point-to-point] **Example:**	Creates or modifies a subinterface. Enters subinterface configuration mode. atm is the interface type.

	Command or Action	Purpose
	`Router(config)# interface atm number.subinterface [point-to-point]`	number is the slot, module, and port number of the interface (for example 1/0/0). .subinterface is the number of the subinterface (for example, 1/0/0.1). (Optional) point-to-point indicates that the subinterface connects directly with another subinterface.
Step 7	pvc [name] vpi/vci **Example:** `Router(config-subif) pvc [name] vpi/vci`	Creates or modifies an ATM permanent virtual circuit (PVC). Enters ATM virtual circuit configuration mode. (Optional) name identifies the PVC and can contain up to 15 characters. vpi/ specifies the ATM network virtual path identifier (VPI) for this PVC. You must specify the slash. Valid values are from 0 to 255. The router treats a value that is outside the range of valid values as the connection ID. The default value is 0. **Note** The arguments vpi and vci cannot both be set to 0; if one is 0, the other cannot be 0. vci specifies the ATM network virtual channel identifier (VCI) for this PVC. Valid values are from 0 to 1 less than the maximum value set for this interface by the atm vc-per-vp command. A value that is out of range causes an " unrecognized command" error message. The VCI value has local significance only and, therefore, is unique only on a single link, not throughout the ATM network. Typically, lower values from 0 to 31 are reserved for specific traffic (for example, F4 OAM, SVC signaling, ILMI, and so on) and should not be used.
Step 8	protocol pppoe group group-name **Example:** `Router(config-atm-vc)# protocol pppoe group group-name`	Enables PPP over Ethernet (PPPoE) sessions to be established on permanent virtual circuits (PVCs). group specifies a PPPoE profile (bba-group) to be used by PPPoE sessions on the interface. group-name is the name of the PPPoE profile (bba-group) to be used by PPPoE sessions on the interface. The group group-name points to the bba-group to be used for applying a virtual template interface with QoS policies to sessions.
Step 9	exit **Example:** `Router(config-atm-vc)# exit`	Exits ATM virtual circuit configuration mode.

	Command or Action	Purpose
Step 10	exit **Example:** `Router(config-subif)# exit`	Exits subinterface configuration mode.

Examples

The following example shows how to associate a virtual template interface with an ATM interface and apply the policies in the virtual template to the sessions on the interface. In the example, the service policy named Parent is applied to the Virtual-Template 8, which is associated with the bba-group named pppoeoa-group. The bba-group is applied to PVC 101/210 on ATM subinterface 4/0/1.10.

```
bba-group pppoe pppoeoa-group
Virtual-Template 8
interface ATM4/0/1.10 point-to-point
pvc 101/210
vbr-nrt 4000 2000 50
no dbs enable
encapsulation aal5snap
protocol pppoe group pppoeoa-group
!
interface Virtual-Template8
ip unnumbered Loopback5555
no logging event link-status
peer default ip address pool pool-1
ppp authentication chap
service-policy output Parent
```

Configuring PPP Session Queueing Using Radius

To configure PPPoEoA session queueing using RADIUS, perform the following configuration tasks:

Configuring the Policy Map

The router allows you to use RADIUS to apply QoS policy maps to PPPoEoA sessions.

Adding the Cisco QoS AV Pairs to the RADIUS Profile

Cisco attribute-value (AV) pairs are vendor-specific attributes (VSAs) that allow vendors such as Cisco to support their own extended attributes. RADIUS attribute 26 is a Cisco VSA used to communicate vendor-specific information between the router and the RADIUS server.

The RADIUS user profile contains an entry for each user that the RADIUS server authenticates. Each entry establishes an attribute the user can access. When configuring PPPoEoA session queueing using RADIUS, enter the following Cisco AV-pair in the appropriate user profile:

```
Cisco-AVPair = "ip:sub-qos-policy-out=<name of egress policy>"
```

The Cisco AV-pair identifies the policy map the router is to use when applying QoS features to a PPPoEoA session. After receiving a service-logon request from the policy server, RADIUS sends a change of authorization

(CoA) request to the router to activate the service for the user, who is already logged in. If the authorization succeeds, the router downloads the name of the policy map from RADIUS using the Cisco AV-pair and applies the QoS policy to the session.

Note Although the router also supports the RADIUS vendor specific attribute (VSA) 38, Cisco-Policy-Down and Cisco-Policy-Up, we recommend that you use the above attribute for QoS policy definitions.

Verifying PPP Session Queueing on ATM VCs

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **show policy-map [interface interface]**
4. show policy-map session [uid uid-number] [input | output [class class-name]]
5. **show running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**show policy-map [interface interface]** **Example:** `Router# show policy-map [interface interface]`	Displays information about the policy map attached to the interface you specify. If you do not specify an interface, it displays information about all of the policy maps configured on the router. interface interface is the interface type and number (for example, atm 4/0/0).
Step 4	show policy-map session [uid uid-number] [input | output [class class-name]] **Example:** `Router# show policy-map session [uid uid-number] [input | output [class class-name]]`	Displays the QoS policy map in effect for subscriber sessions. (Optional) uid defines a unique session ID. (Optional) uid-number is a unique session ID. Valid values are from 1 to 65535. (Optional) input displays the upstream traffic of the unique session.

	Command or Action	Purpose
		(Optional) output displays the downstream traffic of the unique session. (Optional) class identifies the class that is part of the QoS policy-map definition. (Optional) class-name provides a class name that is part of the QoS policy-map definition.
Step 5	**show running-config** **Example:** `Router# show running-config`	Displays the running configuration on the router. The output shows the AAA setup and the configuration of the policy map, ATM VCs, PPPoEoA, dynamic bandwidth selection, virtual template, and RADIUS server.

Configuration Examples for PPP Session Queueing on ATM VCs

Example Configuring PPP Session Queueing on ATM VCs

The following example shows how to configure PPPoEoA session queueing. In the example, a hierarchical QoS policy named pm_hier2_0_2 is associated with Virtual-Template555, which is applied to the broadband aggregation group named pppoeoa-group.

```
bba-group pppoe pppoeoa-group
Virtual-Template 555
!
policy-map pm_hier2_child_0_2
class cm_0
priority level 1
police percent 5 2 ms 0 ms conform-action transmit exceed-action drop violate-action drop
queue-limit 77 packets
class cm_1
shape average percent 80
bandwidth remaining ratio 80
class class-default
shape average percent 50
bandwidth remaining ratio 20
policy-map pm_hier2_0_2
class class-default
shape average percent 100
bandwidth remaining ratio 100
service-policy pm_hier_child_0_2
interface ATM2/0/7.5555 point-to-point
pvc 1/5555
vbr-nrt 4000 2000 50
no dbs enable
encapsulation aal5snap
protocol pppoe group pppoeoa-group
!
!
interface Virtual-Template555
ip unnumbered Loopback5555
no logging event link-status
peer default ip address pool pool-1
```

```
ppp authentication chap
service-policy output pm_hier2_0_2
```

Example Configuring and Applying an Hierarchical Policy Map

The example belo shows how to configure a hierarchical policy and apply it to a virtual template. The example contains a child policy map named child1 with QoS features defined for the gold and bronze traffic classes. The child1 policy is applied to the parent policy map, which is shaped to 512000 bps. The hierarchical policy is applied to the virtual template named virtual-template 1.

```
Router(config)# policy-map child1
Router(config-pmap)# class gold
Router(config-pmap-c)# priority
Router(config-pmap-c)# police percent 40
Router(config-pmap-c)# class bronze
Router(config-pmap-c)# police 8000
Router(config-pmap-c)# exit
Router(config-pmap)# policy-map parent
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape 512000
Router(config-pmap-c)# service-policy child1
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# interface virtual-template 1
Router(config-if)# service-policy output parent
```

Example Setting Up RADIUS for PPP Session Queueing on ATM VCs

This section shows how to define the Cisco AV pairs used to download the policy map name to the router. The first three lines of a subscriber's sample user profile contain the user password, service type, and protocol type. This information is entered into the subscriber's user profile when the user profile is first created. The last line is an example of the Cisco QoS AV-pair added to the user profile. The policy map name downloaded to the router is p23.

```
userid Password = "cisco"
Service-Type = Framed,
Framed-Protocol = PPP,
cisco-avpair = "sub-qos-policy-out=p23"
```

Example Verifying PPP Session Queueing on ATM VCs

Displaying PPP Session Information--show pxf cpu queue session Command

Use the `show pppoe session` command to display the sessions established on the router. In the example below, one session is active with a session ID (SID) of 6.

```
Router# show pppoe session
1 session in LOCALLY_TERMINATED (PTA) State
1 session total
Uniq ID PPPoE RemMAC Port VT VA State
 SID LocMAC VA-st Type
 14 6 0009.b68d.bb37 ATM2/0/7.5555 555 Vi3.1 PTA
   0009.b68d.bc37 VC: 1/5555    UP
```

Displaying PPP Session Information--show policy-map session Command

Use the `show policy-map session` command to display QoS policy map statistics for traffic in the downstream direction. The example below also shows the policy map configurations.

```
Router# show pppoe session
1 session in LOCALLY_TERMINATED (PTA) State
1 session total
Uniq ID PPPoE RemMAC Port VT VA State
 SID LocMAC VA-st Type
 14 6 0009.b68d.bb37 ATM2/0/7.5555 555  Vi3.1  PTA
 0009.b68d.bc37 VC: 1/5555 UP
Router#
Router#
Router# show policy-map session uid 14
SSS session identifier 14 -
 Service-policy output: pm_hier2_0_2
Class-map: class-default (match-any)
0 packets, 0 bytes
30 second offered rate 0 bps, drop rate 0 bps
Match: any
0 packets, 0 bytes
30 second rate 0 bps
Queueing
queue limit 50 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
shape (average) cir 2000000, bc 8000, be 8000
target shape rate 2000000
bandwidth remaining ratio 100
 Service-policy : pm_hier2_child_0_2
queue stats for all priority classes:
Queueing
priority level 1
queue limit 77 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
Class-map: cm_0 (match-any)
0 packets, 0 bytes
30 second offered rate 0 bps, drop rate 0 bps
Match: ip precedence 0
0 packets, 0 bytes
30 second rate 0 bps
Priority: 0% (0 kbps), burst bytes 4470, b/w exceed drops: 0
Priority Level: 1
Police:
104000 bps, 1536 limit, 0 extended limit
conformed 0 packets, 0 bytes; action: transmit
exceeded 0 packets, 0 bytes; action: drop
violated 0 packets, 0 bytes; action: drop
Class-map: cm_1 (match-any)
0 packets, 0 bytes
30 second offered rate 0 bps, drop rate 0 bps
Match: ip precedence 1
0 packets, 0 bytes
30 second rate 0 bps
Queueing
queue limit 237 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
shape (average) cir 1600000, bc 6400, be 6400
target shape rate 1600000
bandwidth remaining ratio 80
```

```
Class-map: class-default (match-any)
0 packets, 0 bytes
30 second offered rate 0 bps, drop rate 0 bps
Match: any
0 packets, 0 bytes
30 second rate 0 bps
Queueing
queue limit 77 packets
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts output/bytes output) 0/0
shape (average) cir 1000000, bc 4000, be 4000
target shape rate 1000000
bandwidth remaining ratio 20
Router# show policy-map pm_hier2_0_2
Policy Map pm_hier2_0_2
Class class-default
Average Rate Traffic Shaping
cir 100%
bandwidth remaining ratio 100
service-policy pm_hier2_child_0_2
Router# show policy-map pm_hier2_child_0_2
Policy Map pm_hier2_child_0_2
Class cm_0
priority level 1
police percent 5 2 ms 0 ms conform-action transmit exceed-action drop violate-action drop
queue-limit 77 packets
Class cm_1
Average Rate Traffic Shaping
cir 80%
bandwidth remaining ratio 80
Class class-default
Average Rate Traffic Shaping
cir 50%
bandwidth remaining ratio 20
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands	*Cisco IOS QoS Command Reference*

Technical Assistance

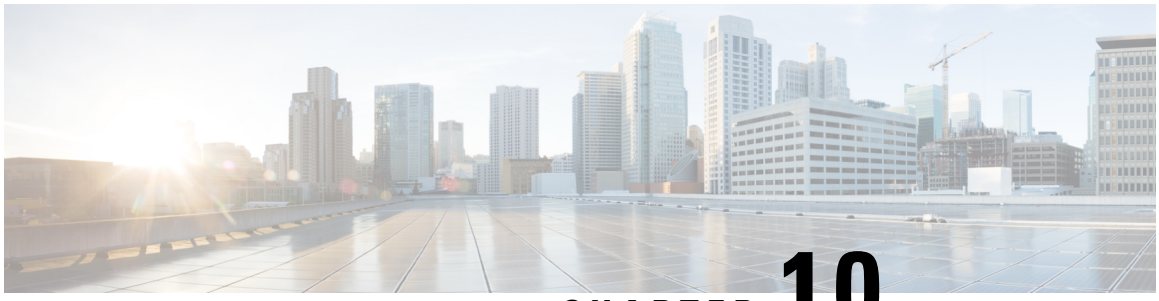
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for PPP Session Queueing on ATM VCs

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 12: Feature Information for PPP Session Queueing on ATM VCs



CHAPTER 10

VP/VC Shaping for PPPoEoA/PPPoA

This feature adds support for ATM VP shaping for VCs with underlying broadband sessions. Per VC and per VP traffic shaping controls or modifies the flow of traffic on an interface. Traffic shaping limits throughput by buffering excess traffic instead of dropping packets. It ensures that traffic from one VC does not adversely impact another VC, thus preventing loss of data. Providing traffic shaping on a per VC and per VP basis allows flexibility and control over every VC and VP configured.

The VP and VC Shaping for PPPoEoA and PPPoA feature is supported for the following ATM traffic service categories:

- Variable bit rate Non-Real-Time (VBR-nRT)
- Unspecified bit rate (UBR)

Prerequisites for VP/VC Shaping for PPPoEoA/PPPoA

- Dynamic changes to VP shaper rate should be enabled.
- The ATM VC create-on-demand functionality (with the VP shaper configured) should be enabled.
- PPP over Ethernet over ATM (PPPoEoA) sessions must be enabled.

Restrictions for VP/VC Shaping for PPPoEoA/PPPoA

- All the VCs parented by a given VP with shaping applied must be of the same type. For example, if a VP shaper is applied to virtual path identifier (VPI) 10, all the virtual circuit identifiers (VCIs) with a VP of 10 must be vbr-nrt or all must be ubr+.
- The **atm pvp rate** command cannot be added or removed if any of the VCs on that ATM interface that are in VP are in the active state. This is not supported in a nonbroadband configuration.

- Configuration of Modular QoS CLI (MQC) policy maps on VPs is not supported. Only configuration of the VP rate using the **atm pvp** command is supported.
- Quality of Service (QoS) on the VP and VC session is supported.
- The sum of the VC shaper rates can oversubscribe the VP shaper rate configured.
- The sum of all the VP shaper rates can oversubscribe the physical rate of the ATM interface.
- VP shapers are supported for any combination of VCs with or without broadband sessions. They may or may not have queuing QoS policies attached.
- On a given ATM interface, there may be mixed VPs with and without shapers.
- When there are multiple VCs in a VP, class-of-service change is not allowed.
- When there is only one VC in a VP, class-of-service change is allowed.
- IP sessions and the existing Intelligent Services Gateway (ISG) on ATM functionality are supported.

Configuring VP/VC Shaping for PPPoEoA/PPPoA

Before you begin

Before you configure VP/VC shaping for PPoEoA/PPPoA, ensure that you configure the ATM interface and define the attributes for each session. A broadband aggregation group (bba-group) configured on an ATM interface points to the virtual template the router will use to apply QoS policies to the sessions.

To configure VP/VC shaping for PPPoEoA/PPPoA on an ATM interface, perform the following configuration task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface atm** *slot/module/port*
4. **mac-address** *mac-address*
5. **no ip address**
6. **atm clock internal**
7. **atm oam flush**
8. **no atm ilmi-keepalive**
9. **exit**
10. **bba-group pppoe** {*group-name* | **global**}
11. **virtual-template** *template-number*
12. **sessions per-vc limit** *per-vc-limit* [**threshold** *threshold-value*]
13. **sessions per-mac limit** *per-mac-limit*
14. **sessions per-vlan limit** *per-vlan-limit*
15. **sessions per-vc throttle** *per-vc-throttle*
16. **exit**
17. **interface atm** *slot/subslot/port* [*subinterface*][**point-to-point** | **multipoint**]

18. **atm pvp** *vpi [peak-rate]*
19. **pvc** *vpi/vci*
20. **vbr-nrt** *output-pcr output-scr[output-maxburstsize]*
21. **dbs enable** [**aggregated** | **maximum**]
22. **encapsulation aal5snap**
23. **protocol pppoe group** {*group-name* | **global**}
24. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables the privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.
Step 3	**interface atm** *slot/module/port* **Example:** `Router(config)# interface atm slot/module/port`	Creates or modifies an ATM interface. Enters the interface configuration mode. Here: *slot/module/port* is the interface number.
Step 4	**mac-address** *mac-address* **Example:** `Router(config-if)# mac-address mac-address`	Specifies the mac address for an interface.
Step 5	**no ip address** **Example:** `Router(config-if)# no ip address`	Disables IP processing on the interface by removing its IP address.
Step 6	**atm clock internal** **Example:** `Router(config-if)#atm clock internal`	Synchronizes the timer between two back-to-back ATM interfaces.
Step 7	**atm oam flush** **Example:** `Router(config-if)# atm oam flush`	Drops all the current and future Operation, Administration, and Maintenance (OAM) cells received on the ATM interface.
Step 8	**no atm ilmi-keepalive** **Example:**	Disables the Interim Local Management Interface (ILMI) keepalives.

	Command or Action	Purpose
	`Router(config-if)# no atm ilmi-keepalive`	
Step 9	**exit** **Example:** `Router(config-if)# exit`	Exits the interface configuration mode.
Step 10	**bba-group pppoe** {*group-name* | **global**} **Example:** `Router(config)# bba-group pppoe group-name`	Defines a PPPoE profile, and enters the BBA group configuration mode. The **global** keyword creates a profile that serves as the default profile for any PPPoE port that is not assigned a specific profile.
Step 11	**virtual-template** *template-number* **Example:** `Router(config-bba-group)# virtual-template template-number`	Specifies which virtual template will be used to clone virtual access interfaces.
Step 12	**sessions per-vc limit** *per-vc-limit* [**threshold** *threshold-value*] **Example:** `Router(config-bba-group)# sessions per-vc limit per-vc-limit`	Specifies the maximum number of PPPoE sessions that can be established over an ATM permanent virtual circuit (PVC)
Step 13	**sessions per-mac limit** *per-mac-limit* **Example:** `Router(config-bba-group)# sessions per-mac limit per-mac limit`	Sets the maximum number of PPPoE sessions permitted per MAC address in a PPPoE profile.
Step 14	**sessions per-vlan limit** *per-vlan-limit* **Example:** `Router(config-bba-group)# sessions per-vlan limit per-vlan-limit`	Specifies the maximum number of PPPoE sessions permitted per VLAN in a PPPoE profile.
Step 15	**sessions per-vc throttle** *per-vc-throttle* **Example:** `Router(config-bba-group)# sessions per-vc throttle per-vc-throttle`	Configures PPPoE connection throttling, which limits the number of PPPoE session requests that can be made from a VC.
Step 16	**exit** **Example:** `Router(config-bba-group)# exit`	Exits the BBA group configuration mode and returns to the global configuration mode.

	Command or Action	Purpose
Step 17	**interface atm** *slot/subslot/port* [*subinterface*][**point-to-point** | **multipoint**] **Example:** `Router(config)# interface atm slot/subslot/port multipoint`	Configures the ATM interface and enters the subinterface configuration mode.
Step 18	**atm pvp** *vpi [peak-rate]* **Example:** `Router(config-subif)# atm pvp vpi[peak-rate]`	Creates a permanent virtual path (PVP) used to multiplex (or bundle) one or more VCs.
Step 19	**pvc** *vpi/vci* **Example:** `Router(config-subif)# atm pvp vpi[peak-rate]`	Creates or assigns a name to an ATM PVC and enters ATM virtual circuit configuration mode.
Step 20	**vbr-nrt** *output-pcr output-scr[output-maxburstsize]* **Example:** `Router(config-if-atm-vc)# vbr-nrt output-pcr output-scr [output-maxburstsize]`	Configures the VBR-nRT QoS and specifies output peak cell rate (PCR), output sustainable cell rate (SCR), and output maximum burst cell size for an ATM PVC, PVC range, switched virtual circuit (SVC), VC class, or VC bundle member.
Step 21	**dbs enable** [**aggregated** | **maximum**] **Example:** `Router(config-if-atm-vc)# dbs enable`	Applies the Dynamic Subscriber Bandwidth Selection QoS parameters.
Step 22	**encapsulation aal5snap** **Example:** `Router(config-if-atm-vc)# encapsulation aal5snap`	Configures the ATM adaptation layer (AAL) and encapsulation type for an ATM VC.
Step 23	**protocol pppoe group** {*group-name* | **global**} **Example:** `Router(config-if-atm-vc)# protocol pppoe group group-name`	Enables PPPoE sessions to be established on PVCs. **group** specifies a PPPoE profile (bba-group) to be used by the PPPoE sessions on the interface. *group-name* is the name of the PPPoE profile (bba-group) to be used by the PPPoE sessions on the interface. **group** *group-name* points to the bba-group to be used for applying a virtual template interface with QoS policies to sessions.
Step 24	**end** **Example:** `Router(config-if-atm-vc)# end`	Ends the session and returns to the privileged EXEC mode.

Example

The following example shows how to configure VP/VC shaping for PPPoEoA/PPPoA:

```
Router(config)#interface ATM1/0/0
Router(config-if)#mac-address 0000.b001.0001
Router(config-if)#no ip address
Router(config-if)#atm clock INTERNAL
Router(config-if)#atm oam flush
Router(config-if)#no atm ilmi-keepalive
Router(config-if)#exit
Router(config)#bba-group pppoe group_basic
Router(config-bba-group)#virtual-template 2
Router(config-bba-group)#sessions per-vc limit 1
Router(config-bba-group)#sessions per-mac limit 1
Router(config-bba-group)#sessions per-vlan limit 1
Router(config-bba-group)#sessions per-vc throttle 1 2 3
Router(config-bba-group)#exit
Router(config)#interface ATM1/0/0.64001 multipoint
Router(config-subif)#atm pvp 1 50000
Router(config-subif)#pvc 1/32
Router(config-if-atm-vc)#vbr-nrt 40000 40000 1
Router(config-if-atm-vc)#dbs enable
Router(config-if-atm-vc)#encapsulation aal5snap
Router(config-if-atm-vc)#protocol pppoe group group_1
Router(config-if-atm-vc)#end
```

Configuration Examples for VP/VC Shaping for PPPoEoA/PPPoA

Example: Configuring VP/VC Shaping for PPPoEoA/PPPoA

The following example shows how to configure VP/VC shaping for PPPoEoA/PPPoA:

```
interface ATM1/0/0
mac-address 0000.b001.0001
no ip address
atm clock INTERNAL
atm oam flush
no atm ilmi-keepalive
!
bba-group pppoe group_basic
virtual-template 2
sessions per-vc limit 1
sessions per-mac limit 1
sessions per-vlan limit 1
sessions per-vc throttle 1 2 3
!
interface ATM1/0/0.1 multipoint
atm pvp 1 1000
pvc 1/10000
 vbr-nrt 500 500 1
 dbs enable
 encapsulation aal5snap
 protocol pppoe group group_basic
```

Example: Verifying VP/VC Shaping for PPPoEoA/PPPoA

The following example shows how to display configuration of a particular PVC.

```
Router# Show ATM pvc
Keys:  A = ATM1/0/0, B = ATM1/0/1, C = ATM1/0/2,
          VCD /                                             Peak Av/Min Burst
Interface  Name          VPI   VCI Type    Encaps     SC     Kbps   Kbps Cells St
A.64001    1             1     3 PVC     F4-OAM     UBR   50000               UP
A.64001    2             1     4 PVC     F4-OAM     UBR   50000               UP
A.64001    11            1    32 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    12            1    33 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    13            1    34 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    14            1    35 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    15            1    36 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    16            1    37 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    17            1    38 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    18            1    39 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    19            1    40 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    20            1    41 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    3             2     3 PVC     F4-OAM     UBR   50000               UP
A.64001    4             2     4 PVC     F4-OAM     UBR   50000               UP
A.64001    21            2    32 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    22            2    33 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    23            2    34 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    24            2    35 PVC     SNAP       VBR   40000  40000      1 UP
```

The following example shows how to display configuration of the traffic parameters for a PVC.

```
Router# Show ATM vc
Keys:  A = ATM1/0/0, B = ATM1/0/1, C = ATM1/0/2,
Codes: DN - DOWN, IN - INACTIVE

          VCD /                                             Peak Av/Min Burst
Interface  Name          VPI   VCI Type    Encaps     SC     Kbps   Kbps Cells St
A.64001    1             1     3 PVC     F4-OAM     UBR   50000               UP
A.64001    2             1     4 PVC     F4-OAM     UBR   50000               UP
A.64001    11            1    32 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    12            1    33 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    13            1    34 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    14            1    35 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    15            1    36 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    16            1    37 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    17            1    38 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    18            1    39 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    19            1    40 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    20            1    41 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    3             2     3 PVC     F4-OAM     UBR   50000               UP
A.64001    4             2     4 PVC     F4-OAM     UBR   50000               UP
A.64001    21            2    32 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    22            2    33 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    23            2    34 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    24            2    35 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    25            2    36 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    26            2    37 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    27            2    38 PVC     SNAP       VBR   40000  40000      1 UP
A.64001    28            2    39 PVC     SNAP       VBR   40000  40000      1 UP
```

The following example shows how to display configuration for VP mode cell relay.

```
Router# Show ATM vp
Keys:  A = ATM1/0/0, B = ATM1/0/1, C = ATM1/0/2,
```

```
                                    Data  CES    Peak     CES Avg/Min Burst     MCR
Interface     VPI SC                VCs  VCs    Kbps     Kbps    Kbps Cells    Kbps    CDVT  Status
A.64001         1 VBR-NRT            10    0   50000       0     N/A   N/A     N/A     N/A  ACTIVE

A.64001         2 VBR-NRT            10    0   50000       0     N/A   N/A     N/A     N/A  ACTIVE

A.64001         3 VBR-NRT            10    0   50000       0     N/A   N/A     N/A     N/A  ACTIVE

A.64001         4 VBR-NRT            10    0   50000       0     N/A   N/A     N/A     N/A  ACTIVE

A.64001         5 VBR-NRT            10    0   50000       0     N/A   N/A     N/A     N/A  ACTIVE

B.64001         6 VBR-NRT            10    0   40000       0     N/A   N/A     N/A     N/A  ACTIVE

B.64001         7 VBR-NRT            10    0   40000       0     N/A   N/A     N/A     N/A  ACTIVE

B.64001         8 VBR-NRT            10    0   40000       0     N/A   N/A     N/A     N/A  ACTIVE

B.64001         9 VBR-NRT            10    0   40000       0     N/A   N/A     N/A     N/A  ACTIVE

B.64001        10 VBR-NRT            10    0   40000       0     N/A   N/A     N/A     N/A  ACTIVE

C.64001        11 VBR-NRT            10    0   30000       0     N/A   N/A     N/A     N/A  ACTIVE

C.64001        12 VBR-NRT            10    0   30000       0     N/A   N/A     N/A     N/A  ACTIVE

C.64001        13 VBR-NRT            10    0   30000       0     N/A   N/A     N/A     N/A  ACTIVE

C.64001        14 VBR-NRT            10    0   30000       0     N/A   N/A     N/A     N/A  ACTIVE

C.64001        15 VBR-NRT            10    0   30000       0     N/A   N/A     N/A     N/A  ACTIVE
```

Additional References

Related Documents

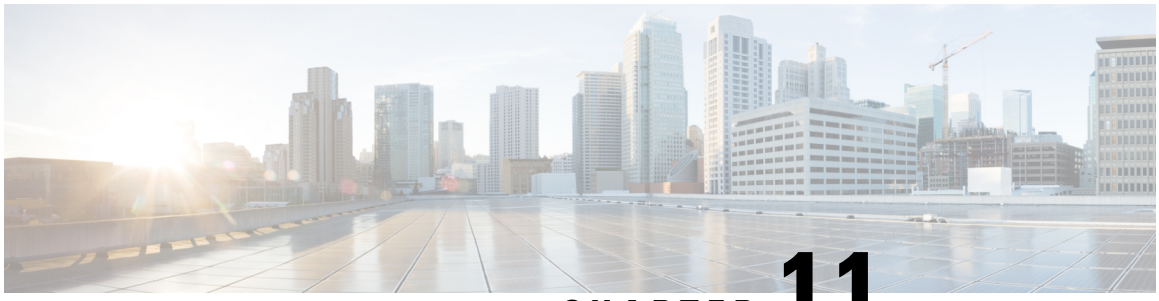
Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands	*Cisco IOS QoS Command Reference*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for VP/VC Shaping for PPPoEoA/PPPoA

Table 13: Feature Information for VP/VC Shaping for PPPoEoA/PPPoA



CHAPTER 11

Hierarchical Color-Aware Policing

The Hierarchical Color-Aware Policing feature provides two levels of policing where the policer ordering is evaluated from child to parent, and there is preferential treatment of certain traffic at the parent level.

- Reverse the order of dataplane policing in hierarchical policies so that they are evaluated from child to parent. In prior releases, the policies are evaluated from parent to child.
- Limited support for color-aware policing (RFC 2697 and RFC 2698) within Quality of Service (QoS) policies.

Prerequisites for Hierarchical Color-Aware Policing

You must already be familiar with relevant features and technologies including modular QoS CLI (MQC) and the master control processor (MCP) software and hardware architecture. The Additional References, on page 135 section provides pointers to relevant feature and technology documents.

Restrictions for Hierarchical Color-Aware Policing

The following restrictions apply to the Hierarchical Color-Aware Policing feature:

- Color-aware class maps support only QoS group matching.
- Only one filter (one match statement) per color-aware class is supported.
- Color-aware statistics are not supported, only existing policer statistics.
- Color-aware class map removal (using the **no class-map***class-map-name* command) is not allowed while the class map is being referenced in a color-aware policer. It must be removed from all color-aware policers (using either the **no conform-color***class-map-name* or **no exceed-color***class-map-name* command first).

- Hierarchical policer evaluation is permanently reversed (not configurable) to support child-to-parent ordering.

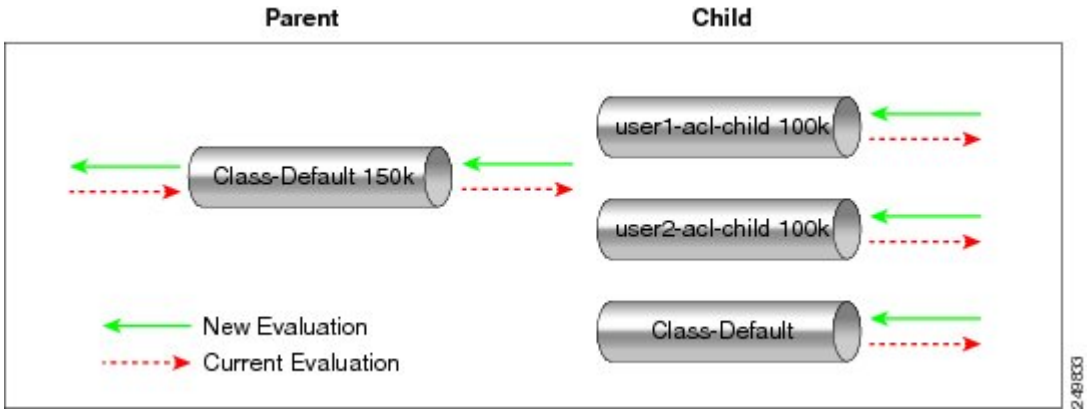
Information About Hierarchical Color-Aware Policing

Hierarchical Order Policing

With the introduction of the Hierarchical Color-Aware Policing feature, the evaluation order is reversed so that policers are evaluated from child to parent in QoS policies. This ordering is a permanent change to the default behavior and is not configurable. The reverse order policer functionality is shared for both ingress and egress directions.

The following sample configuration for a simple two-level policer would result in the changed behavior shown in the figure below:

```
policy-map child
 class user1
  police 100k
 class user2
  police 100k
policy-map parent
 class class-default
  police 150k
  service-policy child
```



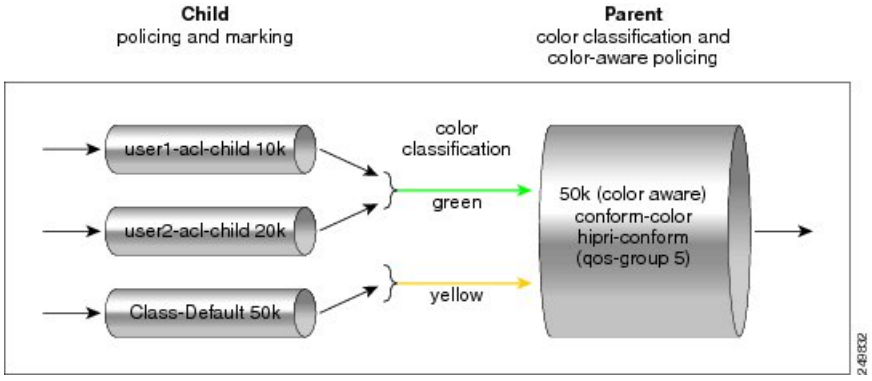
Limited Color-Aware Policing

The following sample configuration for a simple two-level color-aware policer would result in the changed behavior shown in the figure below:

```
ip access-list extended user1-acl
 permit ip host 192.168.1.1 any
 permit ip host 192.168.1.2 any
ip access-list extended user2-acl
 permit ip host 192.168.2.1 any
 permit ip host 192.168.2.2 any
class-map match-all user1-acl-child
```

```
  match access-group name user1-acl
class-map match-all user2-acl-child
  match access-group name user2-acl
class-map match-all hipri-conform
 match qos-group 5
policy-map child-policy
 class user1-acl-child
   police 10000 bc 1500
    conform-action set-qos-transmit 5
class user2-acl-child
   police 20000 bc 1500
    conform-action set-qos-transmit 5
class class-default
   police 50000 bc 1500
policy-map parent-policy
 class class-default
  police 50000 bc 3000
   confirm-action transmit
   exceed-action transmit
   violate-action drop
   conform-color hipri-conform
  service-policy child-policy
```

Figure 2: Simple Two-Level Color-Aware Policer

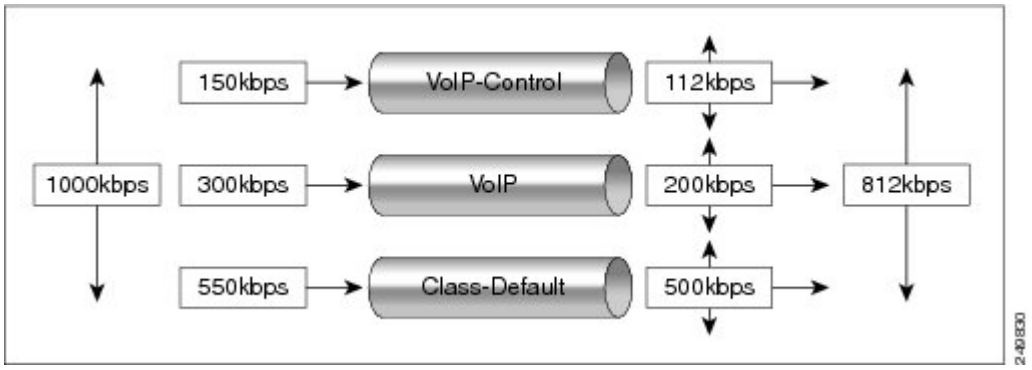


Note To avoid drops at the parent level for "conformed" child traffic, the parent policer must have a rate and burst that are equal to or greater than the sum of the child conform rates and burst sizes. There is no check for inappropriate (parent-to-child) rates and burst sizes in code. You must be aware of this limitation and configure appropriately. In the following example, explicit marking actions are supported in conjunction with color-aware policing and operate similarly color-aware policer marking actions. If these marking actions ("set qos-group," for example) are present in the child policies, the resulting bit values are evaluated by the parent color-aware policer (same as for child policer marking actions): 50k >= 10k (user1-acl-child) + 20k (user2-acl-child)

Policing Traffic in Child Classes and Parent Classes

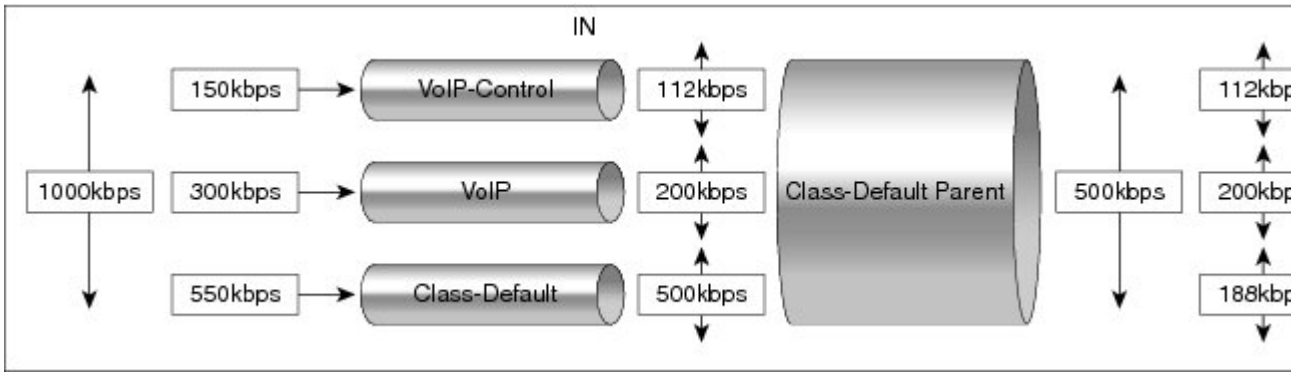
Prior to the release of the Hierarchical Color-Aware Policing feature, policing and marking were typically used as input QoS options. For example, a voice customer was limited to 112 kb/s for voice control and 200 kb/s for voice traffic. The class-default class has no policer. The only limit is the physical bandwidth of the xDSL connection. As shown in the figure below, a customer could send up to 1000 kb/s. However, this involved sending more voice and voice-control packets, which required policing the traffic for both classes.

Figure 3: Policing Traffic in Child Classes



As shown in the figure below, it is important to control the overall input bandwidth. The important requirement is that the premium traffic in the overall limit is not affected. In the figure below, voice and voice-control packets are not dropped in the overall limit. Only packets from the child class-default class are dropped to fulfill the limit.

Figure 4: Policing Traffic in Parent Classes



The first classes function the same way. Voice and voice-control are policed to the allowed level and the class-default class is not affected. In the next level, the overall bandwidth is forced to 500 kb/s and must only drop packets from the class-default class. Voice and voice-control must remain unaffected.

The order of policer execution is as follows:

1. Police the traffic in the child classes, as shown in the figure above, police VoIP-Control class to 112 kb/s, police VoIP class to 200 kb/s, and police class-default to 500 kb/s.
2. Police the traffic in the class default of the parent policy map, but only drop the traffic from the child class default, and do not drop the remaining child classes. As shown in the figure above, 112 kb/s VoIP-Control and 200 kb/s VoIP traffic are unaffected at the parent policer, but 500 kb/s class default from the child is policed to 188kb/s to meet the overall police policy of 500 kb/s at the parent level.

How to Configure Hierarchical Color-Aware Policing

Configuring the Hierarchical Color-Aware Policing Feature

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default** [**fragment** *fragment-class-name*]} [**insert-before** *class-name*] [**service-fragment** *fragment-class-name*]
5. **police** [**cir** *cir*][**bc** *conform-burst*] [**pir** *pir*][**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]][**conform-color hipri-conform**]
6. **service-policy** *policy-map-name*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map parent-policy`	Enters policy-map configuration mode and creates a policy map.
Step 4	**class** {*class-name* | **class-default** [**fragment** *fragment-class-name*]} [**insert-before** *class-name*] [**service-fragment** *fragment-class-name*] **Example:** `Router(config-pmap)# class class-default`	Enters policy-map class configuration mode. • Specifies the name of the class whose policy you want to create or change or specifies the default class (commonly known as the class-default class) before you configure its policy. Repeat this command as many times as necessary to specify the child or parent classes that you are creating or modifying: • *class name* --Name of the class to be configured or whose policy is to be modified. The class name is used for both the class map and to configure a policy for the class in the policy map.

	Command or Action	Purpose
		• **class-default** --Specifies the default class so that you can configure or modify its policy. • **fragment** *fragment-class-name* --(Optional) Specifies the default traffic class as a fragment, and names the fragment traffic class. • **insert-before** *class-name* --(Optional) Adds a class map between any two existing class maps. Inserting a new class map between two existing class maps provides more flexibility when modifying existing policy map configurations. Without this option, the class map is appended to the policy map. **Note** This keyword is supported only on flexible packet matching (FPM) policies. • **service-fragment** *fragment-class-name* --(Optional) Specifies that the class is classifying a collection of fragments. The fragments being classified by this class must all share the same fragment class name.

Step 5	**police** [**cir** *cir*][**bc** *conform-burst*] [**pir** *pir*][**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]][**conform-color hipri-conform**]
	Example:
	`Router(config-pmap-c)# police 50000 bc 3000`
	`Router(config-pmap-c-police)# exceed-action transmit`
	Example:
	`Router(config-pmap-c-police)# violate-action drop`
	Example:
	`Router(config-pmap-c-police)# conform-color hipri-conform`
		Configures traffic policing and specifies multiple actions applied to packets marked as conforming to, exceeding, or violating a specific rate.
		• Enters policy-map class police configuration mode. Use one line per action that you want to specify:
		• **cir** --Committed information rate. Indicates that the CIR will be used for policing traffic.
		• **conform-action** --(Optional) Action to take on packets when the rate is less than the conform burst.
		• **exceed-action** --(Optional) Action to take on packets whose rate is within the conform and conform plus exceed burst.
		• **violate-action** --(Optional) Action to take on packets whose rate exceeds the conform plus exceed burst. You must specify the exceed action before you specify the violate action.
		• **conform-color** --(Optional) Enables color-aware policing (on the policer being configured) and assigns the class map to be used for conform color determination. The **hipri-conform** keyword is the class map (previously configured via the **class-map** command) to be used.

	Command or Action	Purpose
Step 6	**service-policy** *policy-map-name* **Example:** `Router(config-pmap-c-police)# service-policy child-policy`	Specifies a service policy as a QoS policy within a policy map (called a hierarchical service policy). • *policy-map-name* --Name of the predefined policy map to be used as a QoS policy. The name can be a maximum of 40 alphanumeric characters.
Step 7	**end** **Example:** `Router(config-pmap-c-police)# end`	Exits the current configuration mode.

Example

The following is a sample configuration for the Hierarchical Color-Aware Policing feature, showing the reverse order for policing:

```
class-map match-all user1-acl-child
match access-group name user1-acl
class-map match-all user2-acl-child
match access-group name user2-acl
class-map match-all hipri-conform
match qos-group 5
policy-map child-policy
class user1-acl-child
police 10000 bc 1500
conform-action set-qos-transmit 5
class user2-acl-child
police 20000 bc 1500
conform-action set-qos-transmit 5
class class-default
police 50000 bc 1500
policy-map parent-policy
class class-default
police 50000 bc 3000
exceed-action transmit
violate-action drop
conform-color hipri-conform
service-policy child-policy
```

Configuration Examples for Hierarchical Color-Aware Policing

Example Enable the Hierarchical Color-Aware Policing Feature

The following example shows a sample configuration that enables the Hierarchical Color-Aware Policing feature:

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
```

```
Router(config)# ip access-list extended user1-acl
Router(config-ext-nacl)# permit ip host 192.168.1.1 any
Router(config-ext-nacl)# permit ip host 192.168.1.2 any
Router(config-ext-nacl)# ip access-list extended user2-acl
Router(config-ext-nacl)# permit ip host 192.168.2.1 any
Router(config-ext-nacl)# permit ip host 192.168.2.2 any
Router(config-ext-nacl)# exit
Router(config)# class-map match-all user1-acl-child
Router(config-cmap)# match access-group name user1-acl
Router(config-cmap)# class-map match-all user2-acl-child
Router(config-cmap)# match access-group name user2-acl
Router(config-cmap)# class-map match-all hipri-conform
Router(config-cmap)# match qos-group 5
Router(config-cmap)# exit
Router(config)# policy-map child-policy
Router(config-pmap)# class user1-acl-child
Router(config-pmap-c)# police cir 10000 bc 1500
Router(config-pmap-c-police)# class user2-acl-child
Router(config-pmap-c)# police cir 20000 bc 1500
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# policy-map parent-policy
Router(config-pmap)# class class-default
Router(config-pmap-c)# police cir 50000 bc 3000
Router(config-pmap-c-police)# exceed-action transmit
Router(config-pmap-c-police)# violate-action drop
Router(config-pmap-c-police)# conform-color hipri-conform
Router(config-pmap-c-police)# service-policy child-policy
```

Example Disallowing Multiple Entries in Class Map

The following example shows a rejected attempt to configure multiple entries in a class map:

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# class-map hipri-conform
Router(config-cmap)# match qos-group 5
Router(config-cmap)# match qos-group 6
Only one match statement is supported for color-aware policing
Router(config-cmap)# no match qos-group 6
```

Example Disallowing the Removal of an Active Color-Aware Class Map

The following example shows that an active color-aware class map cannot be disallowed:

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# no class-map hipri-conform
Class-map hipri-conform is being used
```

Example Dismantling a Configuration of the Hierarchical Color-Aware Policing Feature

The following example shows how to dismantle the configuration of the Hierarchical Color-Aware Policing feature:

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# no policy-map parent-policy
Router(config)# no policy-map child-policy
Router(config)# no class-map hipri-conform
Router(config)# no class-map user1-acl-child
Router(config)# no class-map user2-acl-child
```

Example Enabling Hierarchical Color-Aware Policing

The following example shows how to enable Hierarchical Color-Aware Policing:

```
Router# configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# ip access-list extended user1-acl
Router(config-ext-nacl)# permit ip host 192.168.1.1 any
Router(config-ext-nacl)# permit ip host 192.168.1.2 any
Router(config-ext-nacl)# ip access-list extended user2-acl
Router(config-ext-nacl)# permit ip host 192.168.2.1 any
Router(config-ext-nacl)# permit ip host 192.168.2.2 any
Router(config-ext-nacl)# class-map match-all user1-acl-child
Router(config-cmap)# match access-group name user1-acl
Router(config-cmap)# class-map match-all user2-acl-child
Router(config-cmap)# match access-group name user2-acl
Router(config-cmap)# class-map match-all hipri-conform
Router(config-cmap)# match qos-group 5
Router(config-cmap)# policy-map child-policy
Router(config-pmap)# class user1-acl-child
Router(config-pmap-c)# police 10000 bc 1500
Router(config-pmap-c-police)# conform-action set-qos-transmit 5
Router(config-pmap-c-police)# class user2-acl-child
Router(config-pmap-c)# police 20000 bc 1500
Router(config-pmap-c-police)# conform-action set-qos-transmit 5
Router(config-pmap-c-police)# class class-default
Router(config-pmap-c)# police 50000 bc 1500
Router(config-pmap-c-police)# policy-map parent-policy
Router(config-pmap)# class class-default
Router(config-pmap-c)# police 50000 bc 3000
Router(config-pmap-c-police)# exceed-action transmit
Router(config-pmap-c-police)# violate-action drop
Router(config-pmap-c-police)# conform-color hipri-conform
Router(config-pmap-c-police)# service-policy child-policy
Router(config-pmap-c)# end
Router#
*Sep 16 12:31:11.536: %SYS-5-CONFIG_I: Configured from console by console
Router# show class-map
Class Map match-all user1-acl-child (id 4)
Match access-group name user1-acl
Class Map match-all user2-acl-child (id 5)
Match access-group name user2-acl
Class Map match-any class-default (id 0)
Match any
```

```
Class Map match-all hipri-conform (id 3)
Match qos-group 5
Router# show policy-map
Policy Map parent-policy
Class class-default
police cir 50000 bc 3000 be 3000
conform-color hipri-conform
conform-action transmit
exceed-action transmit
violate-action drop
service-policy child-policy
Policy Map police
Class prec1
priority level 1 20000 (kb/s)
Class prec2
bandwidth 20000 (kb/s)
Class class-default
bandwidth 20000 (kb/s)
Policy Map child-policy
Class user1-acl-child
police cir 10000 bc 1500
conform-action set-qos-transmit 5
exceed-action drop
Class user2-acl-child
police cir 20000 bc 1500
conform-action set-qos-transmit 5
exceed-action drop
Class class-default
police cir 50000 bc 1500
conform-action transmit
exceed-action drop
```

Example Applying show Command with Hierarchical Color-Aware Policing

The following is sample output from the **show policy-map interface** command when a policy with hierarchical color-aware policing is applied:

```
Router# show policy-map interface
GigabitEthernet0/0/0
Service-policy input: parent-policy
Class-map: class-default (match-any)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: any
police:
cir 50000 bps, bc 3000 bytes, be 3000 bytes
conformed 0 packets, 0 bytes; actions:
transmit
exceeded 0 packets, 0 bytes; actions:
transmit
violated 0 packets, 0 bytes; actions:
drop
No color-aware policing statistics available
conformed 0000 bps, exceed 0000 bps, violate 0000 bps
Service-policy : child-policy
Class-map: user1-acl-child (match-all)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: access-group name user1-acl
police:
cir 10000 bps, bc 1500 bytes
conformed 0 packets, 0 bytes; actions:
```

```
set-qos-transmit 5
exceeded 0 packets, 0 bytes; actions:
drop
conformed 0000 bps, exceed 0000 bps
Class-map: user2-acl-child (match-all)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: access-group name user2-acl
police:
cir 20000 bps, bc 1500 bytes
conformed 0 packets, 0 bytes; actions:
set-qos-transmit 5
exceeded 0 packets, 0 bytes; actions:
drop
conformed 0000 bps, exceed 0000 bps
Class-map: class-default (match-any)
0 packets, 0 bytes
5 minute offered rate 0000 bps, drop rate 0000 bps
Match: any
police:
cir 50000 bps, bc 1500 bytes
conformed 0 packets, 0 bytes; actions:
transmit
exceeded 0 packets, 0 bytes; actions:
drop
conformed 0000 bps, exceed 0000 bps
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Quality of Service commands	*Cisco IOS Quality of Service Solutions Command Reference*
Quality of Service configuration information	*Cisco IOS QoS Configuration Guide, Cisco IOS XE Release 3S*

Standards

Standard	Title
No new or modified standards are supported by this feature.	--

MIBs

MIB	MIBs Link
• CISCO-CLASS-BASED-QOS-MIB • CISCO-CLASS-BASED-QOS-CAPABILITY-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2697	*A Single Rate Three Color Marker*
RFC 2698	*A Two Rate Three Color Marker*

Technical Assistance

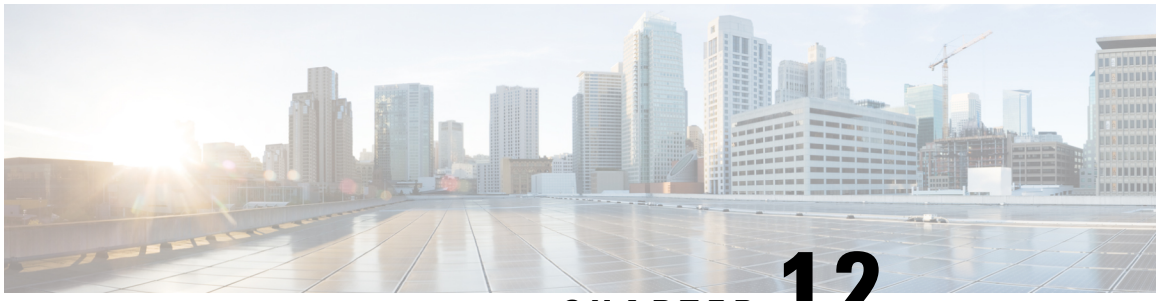
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Hierarchical Color-Aware Policing

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 14: Feature Information for Hierarchical Color-Aware Policing



CHAPTER 12

IPv6 QoS: MQC Traffic Policing

Configuration or command usage for policing are the same in IPv6 environments as for IPv4 environments.

Information About IPv6 QoS: MQC Traffic Policing

Implementation Strategy for QoS for IPv6

IPv6 packets are forwarded by paths that are different from those for IPv4. QoS features supported for IPv6 environments include packet classification, queueing, traffic shaping, weighted random early detection (WRED), class-based packet marking, and policing of IPv6 packets. These features are available at both the process switching and Cisco Express Forwarding switching paths of IPv6.

All of the QoS features available for IPv6 environments are managed from the modular QoS command-line interface (MQC). The MQC allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach those traffic policies to interfaces.

To implement QoS in networks running IPv6, follow the same steps that you would follow to implement QoS in networks running only IPv4. At a very high level, the basic steps for implementing QoS are as follows:

- Know which applications in your network need QoS.
- Understand the characteristics of the applications so that you can make decisions about which QoS features would be appropriate.
- Know your network topology so that you know how link layer header sizes are affected by changes and forwarding.
- Create classes based on the criteria you establish for your network. In particular, if the same network is also carrying IPv4 traffic along with IPv6, decide if you want to treat both of them the same way or treat them separately and specify match criteria accordingly. If you want to treat them the same, use match statements such as **match precedence**, **match dscp**. If you want to treat them separately, add match criteria such as **match protocol ip** and **match protocol ipv6** in a match-all class map.
- Create a policy to mark each class.
- Work from the edge toward the core in applying QoS features.

- Build the policy to treat the traffic.
- Apply the policy.

Traffic Policing in IPv6 Environments

Congestion management for IPv6 is similar to IPv4, and the commands used to configure queueing and traffic shaping features for IPv6 environments are the same commands as those used for IPv4. Traffic shaping allows you to limit the packet dequeue rate by holding additional packets in the queues and forwarding them as specified by parameters configured for traffic shaping features. Traffic shaping uses flow-based queueing by default. CBWFQ can be used to classify and prioritize the packets. Class-based policer and generic traffic shaping (GTS) or Frame Relay traffic shaping (FRTS) can be used for conditioning and policing traffic.

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC), hierarchical policies, policy maps	"Applying QoS Features Using the MQC" module
Policing and shaping traffic	"Policing and Shaping Overview" module

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	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

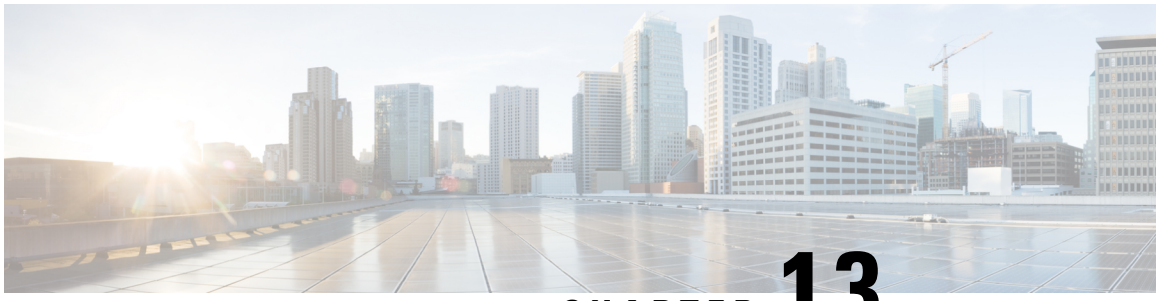
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for IPv6 QoS: MQC Traffic Policing

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 15: Feature Information for IPv6 QoS: MQC Traffic Policing



CHAPTER 13

Traffic Policing

This feature module describes the Traffic Policing feature. The Traffic Policing feature performs the following functions:

- Limits the input or output transmission rate of a class of traffic based on user-defined criteria.
- Marks packets by setting the ATM Cell Loss Priority (CLP) bit, Frame Relay Discard Eligibility (DE) bit, IP precedence value, IP differentiated services code point (DSCP) value, MPLS experimental value, and Quality of Service (QoS) group.

Traffic policing allows you to control the maximum rate of traffic that is transmitted or received on an interface. The Traffic Policing feature is applied when a service-policy containing the feature is attached to an interface. A service-policy (traffic policy) is configured using the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC).

Restrictions for Traffic Policing

- Traffic policing can be configured on an interface or a subinterface.
- Traffic policing is not supported on the EtherChannel interfaces.

Benefits

Bandwidth Management Through Rate Limiting

Traffic policing allows you to control the maximum rate of traffic transmitted or received on an interface. Traffic policing is often configured on interfaces at the edge of a network to limit traffic into or out of the network. In most Traffic Policing configurations, traffic that falls within the rate parameters is transmitted, whereas traffic that exceeds the parameters is dropped or transmitted with a different priority.

Packet Marking

Packet marking allows you to partition your network into multiple priority levels or classes of service (CoS). A packet is marked and these markings can be used to identify and classify traffic for downstream devices. In some cases, such as ATM Cell Loss Priority (CLP) marking or Frame Relay Discard Eligibility (DE) marking, the marking is used to classify traffic.

- Use traffic policing to set the IP precedence or DSCP values for packets entering the network. Networking devices within your network can then use the adjusted IP precedence values to determine how the traffic should be treated. For example, the Weighted Random Early Detection (WRED) feature uses the IP precedence values to determine the probability that a packet will be dropped.
- Use traffic policing to assign packets to a QoS group. The router uses the QoS group to determine how to prioritize packets within the router.

Traffic can be marked without using the Traffic Policing feature. If you want to mark traffic but do not want to use Traffic Policing, see the "Marking Network Traffic" module.

Packet Prioritization for Frame Relay Frames

The Traffic Policing feature allows users to mark the Frame Relay DE bit of the Frame Relay frame. The Frame Relay DE bit is one bit and, therefore, can be set to either 0 or 1. In congested environments, frames with the DE bit set to 1 are discarded before frames with the DE bit set to 0.

Packet Prioritization for ATM Cells

The Traffic Policing feature allows users to mark the ATM CLP bit in ATM cells. The ATM CLP bit is used to prioritize packets in ATM networks. The ATM CLP bit is one bit and, therefore, can be set to either 0 or 1. In congested environments, cells with the ATM CLP bit set to 1 are discarded before cells with the ATM CLP bit set to 0.

How to Configure Traffic Policing

Configuring Traffic Policing

Command	Purpose
Router(config-pmap-c)# **police** *bps burst-normal burst-max* **conform-action** *action* **exceed-action** *action* **violate-action** *action*	Specifies a maximum bandwidth usage by a traffic class. **Note** The Traffic Policing feature works with a token bucket mechanism. There are currently two types of token bucket algorithms: a single token bucket algorithm and a two token bucket algorithm. A single token bucket system is used when the **violate-action** option is not specified, and a two token bucket system is used when the **violate-action** option is specified.

Monitoring and Maintaining Traffic Policing

Command	Purpose
Router# **show policy-map**	Displays all configured policy maps.
Router# **show policy-map** *policy-map-name*	Displays the user-specified policy map.
Router# **show policy-map interface**	Displays statistics and configurations of all input and output policies that are attached to an interface.

Configuration Examples for Traffic Policing

Example Configuring a Service Policy That Includes Traffic Policing

The following configuration shows how to define a traffic class (with the **class-map**command) and associate that traffic class with a traffic policy (with the **policy-map** command). Traffic policing is applied in the traffic policy. The **service-policy** command is then used to attach the traffic policy to the interface.

In this particular example, traffic policing is configured with the Committed Information Rate (CIR) at 8000 bits per second, the normal burst size at 2000 bytes, and the excess burst size at 4000 bytes. Packets coming into FastEthernet interface 1/1/1 are evaluated by the token bucket algorithm to analyze whether packets conform exceed, or violate the specified parameters. Packets that conform are transmitted, packets that exceed are assigned a QoS group value of 4 and are transmitted, and packets that violate are dropped.

```
Router(config)# class-map acgroup2
Router(config-cmap)# match access-group 2
Router(config-cmap)# exit
Router(config)# policy-map police
Router(config-pmap)# class acgroup2
Router(config-pmap-c)# police 8000 2000 4000 conform-action transmit exceed-action
set-qos-transmit 4 violate-action drop
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# interface fastethernet1/1/1
Router(config-if)# service-policy input police
Router(config-if)# end
```

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Conceptual information about policing and shaping	"Policing and Shaping Overview" module

Related Topic	Document Title
MQC	"Applying QoS Features Using the MQC" module
Marking network traffic	"Marking Network Traffic" module
IPv6 Traffic Policing	“IPv6 QoS: MQC Traffic Policing” module in the *QoS: Policing and Shaping Configuration Guide*.

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
• CISCO-CLASS-BASED-QOS-MIB • CISCO-CLASS-BASED-QOS-CAPABILITY-MIB	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2697	*A Single Rate Three Color Marker*

Technical Assistance

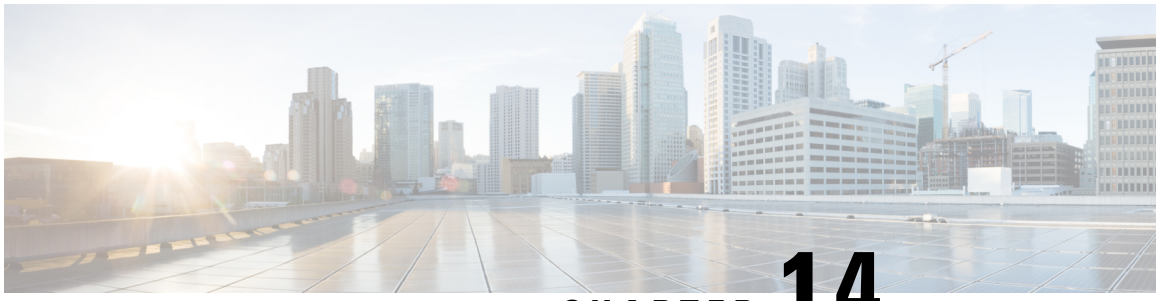
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Traffic Policing

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 16: Feature Information for Traffic Policing



CHAPTER 14

Policer Enhancement Multiple Actions

This document describes the Policer Enhancement Multiple Actions feature and includes the following sections:

Feature Overview

This feature further extends the functionality of the Cisco IOS XE single-rate policer and the Two-Rate Policer feature. The Traffic Policing and Two-Rate Policer features are traffic policing mechanisms that allow you to control the maximum rate of traffic sent or received on an interface. Both of these traffic policing mechanisms mark packets as either conforming to, exceeding, or violating a specified rate. After a packet is marked, you can specify an action to be taken on the packet based on that marking.

With both the Traffic Policing feature and the Two-Rate Policer feature, you can specify only one conform action, one exceed action, and one violate action. Now with the new Policer Enhancement Multiple Actions feature, you can specify multiple conform, exceed, and violate actions for the marked packets.

You specify the multiple actions by using the *action* argument of the **police** command. The resulting actions are listed in the table below.

Table 17: police Command Action Arguments

Specified Action	Result
drop	Drops the packet.
set-clp-transmit	Sets the ATM Cell Loss Priority (CLP) bit from 0 to 1 on the ATM cell and transmits the packet.
set-cos-transmit	Sets the Class of Service (CoS) value and transmits the packet.
set-discard-class-transmit	Sets the discard-class value and transmits the packet.

Specified Action	Result
set-dscp-transmit *new-dscp*	Sets the IP differentiated services code point (DSCP) value and transmits the packet with the ATM CLP bit set to 1.
set-frde-transmit	Sets the Frame Relay Discard Eligibility (DE) bit from 0 to 1 on the Frame Relay frame and transmits the packet.
set-mpls-exp-transmit	Sets the Multiprotocol Label Switching (MPLS) experimental (EXP) bits from 0 to 7 and transmits the packet.
set-mpls-exp-imposition-transmit	Sets the MPLS EXP bits from 0 to 7 at tag imposition and transmits the packet.
set-prec-transmit *new-prec*	Sets the IP Precedence level and transmits the packet.
set-qos-transmit *new-qos*	Sets the Quality of Service (QoS) group value and transmits the packet.
transmit	Transmits the packet.

Benefits

Before this feature, you could specify only *one* marking action for a packet, in addition to transmitting the packet. This feature provides enhanced flexibility by allowing you to specify *multiple* marking actions for a packet, as required. For example, if you know the packet will be transmitted through both a TCP/IP and a Frame Relay environment, you can change the DSCP value of the exceeding or violating packet, and also set the Frame Relay Discard Eligibility (DE) bit from 0 to 1 to indicate lower priority.

Restrictions

The **shape** (percent) command, when used in "child" (nested) policy maps, is not supported on the Cisco 7500, the Cisco 7200, or lower series routers. Therefore, the **shape** (percent) command cannot be configured for use in nested policy maps on these routers.

Related Features and Technologies

- Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC)
- Class-Based Weighted Fair Queueing (CBWFQ)
- Class-Based Packet Marking
- Traffic Policing
- Two-Rate Policing

Related Documents

- "Applying QoS Features Using the MQC" module
- "Configuring Weighted Fair Queueing" module

- "Marking Network Traffic" module
- "Policing and Shaping Overview" module
- "Traffic Policing" module
- "Two-Rate Policer" module
- "Policer Enhancements-Multiple Actions" module
- "Cisco Express Forwarding Overview" module
- Cisco IOS Quality of Service Solutions Command Reference
- Cisco IOS Switching Services Command Reference
- RFC 2697, *A Single Rate Three Color Marker*
- RFC 2698, *A Two Rate Three Color Marker*

Supported Standards MIBs and RFCs

Standards

None

MIBs

- CISCO-CLASS-BASED-QOS-MIB
- CISCO-CLASS-BASED-QOS-CAPABILITY-MIB

To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL:

http://tools.cisco.com/ITDIT/MIBS/servlet/index

If Cisco MIB Locator does not support the MIB information that you need, you can also obtain a list of supported MIBs and download MIBs from the Cisco MIBs page at the following URL:

http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml

To access Cisco MIB Locator, you must have an account on Cisco.com. If you have forgotten or lost your account information, send a blank e-mail to cco-locksmith@cisco.com. An automatic check will verify that your e-mail address is registered with Cisco.com. If the check is successful, account details with a new random password will be e-mailed to you. Qualified users can establish an account on Cisco.com by following the directions found at this URL:

http://www.cisco.com/register

RFCs

None

Prerequisites

- On a Cisco 7500 series router, CEF or dCEF must be configured on the interface before you can use the Policer Enhancement -- Multiple Actions feature.
- To configure the Policer Enhancement -- Multiple Actions feature, a traffic class and a service policy must be created, and the service policy must be attached to a specified interface.

Configuration Tasks

Configuring Multiple Policer Actions

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map-name*
2. Router(config-pmap)# **class** *class-default*
3. Router(config-pmap-c)# **police** {**cir** *cir*}[**bc** *conform-burst*]{**pir** *pir*} [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map-name*	Creates a policy map. Enters policy-map configuration mode.
Step 2	Router(config-pmap)# **class** *class-default*	Specifies the default traffic class for a service policy. Enters policy-map class configuration mode.
Step 3	Router(config-pmap-c)# **police** {**cir** *cir*}[**bc** *conform-burst*]{**pir** *pir*} [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]]	Configures traffic policing and specifies multiple actions applied to packets marked as conforming to, exceeding, or violating a specific rate. Use one line per action that you want to specify. Enters policy-map class police configuration mode.

Verifying the Multiple Policer Actions Configuration

Command	Purpose
`Router# show policy-map interface`	Displays statistics and configurations of all input and output policies attached to an interface.

Troubleshooting Tips

- Check the interface type. Verify that your interface is not listed as a nonsupported interface.
- For input traffic policing on a Cisco 7500 series router, verify that Cisco Express Forwarding or Distributed Cisco Express Forwarding is configured on the interface on which traffic policing is configured.
- For output traffic policing on a Cisco 7500 series router, ensure that the incoming traffic is Cisco Express Forwarding-switched or Distributed Cisco Express Forwarding-switched. Traffic policing cannot be used on the switching path unless Cisco Express Forwarding or Distributed Cisco Express Forwarding switching is enabled.

Monitoring and Maintaining the Multiple Policer Actions

Command	Purpose
`Router# show policy-map`	Displays all configured policy maps.
`Router# show policy-map` *policy-map-name*	Displays the user-specified policy map.
`Router# show policy-map interface`	Displays statistics and configurations of all input and output policies that are attached to an interface.

Configuration Examples

Example Multiple Actions in a Two-Rate Policer

In the following example, a policy map called police is configured to use a two-rate policer to police traffic leaving an interface. Two rates, a committed information rate (CIR) of 1 Mbps and a peak information rate (PIR) of 2 Mbps, have been specified.

```
Router(config)# policy-map police
Router(config-pmap)# class class-default
Router(config-pmap-c)# police cir 1000000 pir 2000000

Router(config-pmap-c-police)# conform-action transmit
Router(config-pmap-c-police)# exceed-action set-prec-transmit 4
Router(config-pmap-c-police)# exceed-action set-frde
Router(config-pmap-c-police)# violate-action set-prec-transmit 2
Router(config-pmap-c-police)# violate-action set-frde-transmit

Router(config-pmap-c-police)# end
```

The following actions will be performed on packets associated with the policy map called police:

- All packets marked as conforming to these rates (that is, packets conforming to the CIR) will be transmitted unaltered.

- All packets marked as exceeding these rates (that is, packets exceeding the CIR but not exceeding the PIR) will be assigned an IP Precedence level of 4, the DE bit will be set to 1, and then transmitted.
- All packets marked as violating the rate (that is, exceeding the PIR) will be assigned an IP Precedence level of 2, the DE bit will be set to 1, and then transmitted.

Example Verifying the Multiple Policer Actions

The following sample output of the **show policy-map**command displays the configuration for a service policy called police. In this service policy, multiple actions for packets marked as exceeding the specified CIR rate have been configured. For those packets, the IP Precedence level is set to 4, the DE bit is set to 1, and the packet is transmitted. Multiple actions for packets marked as violating the specified PIR rate have also been configured. For those packets, the IP Precedence level is set to 2, the DE bit is set to 1, and the packet is transmitted.

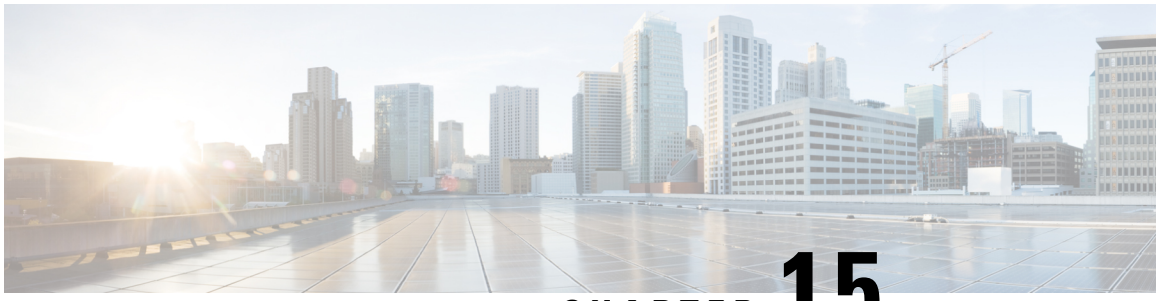
```
Router# show policy-map police
  Policy Map police
    Class class-default
     police cir 1000000 bc 31250 pir 2000000 be 31250
       conform-action transmit
       exceed-action set-prec-transmit 4
       exceed-action set-frde-transmit
       violate-action set-prec-transmit 2
       violate-action set-frde-transmit
```

Feature Information for Policer Enhancement Multiple Actions

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

For more information about the platform support and Cisco software image support, use the Cisco Feature Navigator. To access the Cisco Feature Navigator, go to www.cisco.com/go/cfn. You do not need an account on Cisco.com to use this site.

Table 18: Feature Information for QoS for dVTI



CHAPTER 15

Control Plane Policing

The Control Plane Policing feature allows you to configure a quality of service (QoS) filter that manages the traffic flow of control plane packets to protect the control plane of Cisco IOS XE routers and switches against reconnaissance and denial-of-service (DoS) attacks. In this way, the control plane (CP) can help maintain packet forwarding and protocol states despite an attack or heavy traffic load on the router or switch.

Restrictions for Control Plane Policing

Output Rate-Limiting Support

Output rate-limiting is performed in silent (packet discard) mode. Silent mode enables a router to silently discard packets using policy maps applied to output control plane traffic with the **service-policy output** command. For more information, see the “Output Rate-Limiting and Silent Mode Operation” section.

MQC Restrictions

The Control Plane Policing feature requires the Modular QoS CLI (MQC) to configure packet classification, packet marking, and traffic policing. All restrictions that apply when you use the MQC to configure traffic policing also apply when you configure control plane policing. Only two MQC commands are supported in policy maps—**police** and **set**.

Match Criteria Support and Restrictions

The following classification (match) criteria are supported:

- Standard and extended IP access control lists (ACLs).

- In class-map configuration mode, match criteria specified by the following commands:
  - **match dscp**
  - **match ip dscp**
  - **match ip precedence**
  - **match precedence**
  - **match protocol arp**
  - **match protocol ipv6**
  - **match protocol pppoe**

Note The **match protocol pppoe** command matches all PPPoE data packets that are sent to the control plane.

- **match protocol pppoe-discovery**

Note The **match protocol pppoe-discovery** command matches all PPPoE control packets that are sent to the control plane.

- **match qos-group**

Note The **match input-interface** command is not supported.

Note Features that require Network-Based Application Recognition (NBAR) classification may not work well at the control plane level.

Information About Control Plane Policing

Benefits of Control Plane Policing

Configuring the Control Plane Policing feature on your Cisco router or switch provides the following benefits:

- Protection against DoS attacks at infrastructure routers and switches
- QoS control for packets that are destined to the control plane of Cisco routers or switches
- Ease of configuration for control plane policies
- Better platform reliability and availability

Control Plane Terms to Understand

The following terms are used for the Control Plane Policing feature:

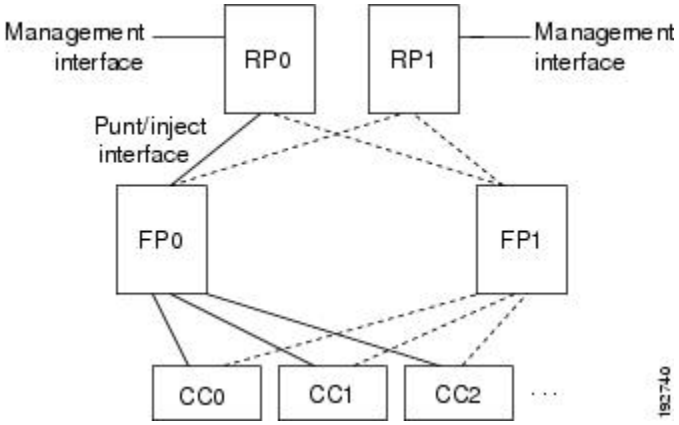
- Control plane—A collection of processes that run at the process level on the Route Processor (RP). These processes collectively provide high-level control for most Cisco IOS XE functions. The traffic sent to or sent by the control plane is called control traffic.
- Forwarding plane—A device that is responsible for high-speed forwarding of IP packets. Its logic is kept simple so that it can be implemented by hardware to do fast packet-forwarding. It punts packets that require complex processing (for example, packets with IP options) to the RP for the control plane to process them.

Control Plane Policing Overview

To protect the control plane on a router from DoS attacks and to provide fine-control over the traffic to or from the control plane, the Control Plane Policing feature treats the control plane as a separate entity with its own interface for ingress (input) and egress (output) traffic. This interface is called the punt/inject interface, and it is similar to a physical interface on the router. Along this interface, packets are punted from the forwarding plane to the RP (in the input direction) and injected from the RP to the forwarding plane (in the output direction). A set of quality of service (QoS) rules can be applied on this interface in order to achieve CoPP.

These QoS rules are applied only after the packet has been determined to have the control plane as its destination or when a packet exits from the control plane. You can configure a service policy (QoS policy map) to prevent unwanted packets from progressing after a specified rate limit has been reached; for example, a system administrator can limit all TCP/SYN packets that are destined for the control plane to a maximum rate of 1 megabit per second.

Figure 5: Abstract Illustration of a Device with Dual RPs and Dual Forwarding Panes



The figure above provides an abstract illustration of a device with dual RPs and dual forwarding planes. Only one RP and one forwarding plane are active at any time. The other RP and forwarding plane are in stand-by mode and do not receive traffic from the carrier card (CC). Packets destined to the control plane come in through the carrier card and then go through the active forwarding plane before being punted to the active RP. When an input QoS policy map is configured on the control plane, the active forwarding plane performs the QoS action (for example, a transmit, drop, or set action) before punting packets to the active RP in order to achieve the best protection of the control plane in the active RP.

On the other hand, packets exiting the control plane are injected to the active forwarding plane, and then go out through the carrier card. When an output QoS policy map is configured on the control plane, the active forwarding plane performs the QoS action after receiving the injected packets from the RP. This process saves the valuable CPU resource in the RP.

Note As shown in "Control Plane Policing Overview" section, the management interface is directly connected to the RP, so all traffic through the management interface to or from the control-plane is not subject to the CoPP function performed by the forwarding plane.

In high-availability (HA) mode, when an RP switchover happens, the active forwarding plane forwards traffic to the new active RP along the new punt/inject interface. The active forwarding plane continues to perform the CoPP function before punting traffic to the new active RP. When a forwarding plane switchover happens, the new active forwarding plane receives traffic from the carrier card and performs the CoPP function before punting traffic to the active RP.

Note The handles some traditional control traffic in the forwarding plane directly to reduce the load on the control plane. One example is the IP Internet Control Message Protocol (ICMP) echo-request packet sent to this router. When a router receives such packets, the packets are handled directly in the forwarding plane without being punted to the RP. In order to be consistent with other Cisco routers and to provide the same capability to control such packets using CoPP, the router extends the CoPP function on such packets, even though the packets are not punted to the RP. Customers can still use the CoPP function to rate-limit or to mark such packets.

Output Rate-Limiting and Silent Mode Operation

A router is automatically enabled to silently discard packets when you configure output policing on control plane traffic using the **service-policy output** *policy-map-name* command.

Rate-limiting (policing) of output traffic from the control plane is performed in silent mode. In silent mode, a router that is running Cisco IOS XE software operates without sending any system messages. If a packet that is exiting the control plane is discarded for output policing, you do not receive an error message.

How to Use Control Plane Policing

Defining Control Plane Services

Perform this task to define control plane services, such as packet rate control and silent packet discard for the active RP.

Before you begin

Before you enter control-plane configuration mode to attach an existing QoS policy to the control plane, you must first create the policy using MQC to define a class map and policy map for control plane traffic.

Note

- Platform-specific restrictions, if any, are checked when the service policy is applied to the control plane interface.
- Output policing does not provide any performance benefits. It simply controls the information that is leaving the device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **control-plane**
4. **service-policy** {**input** | **output** *policy-map-name*}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**control-plane** **Example:** `Device(config)# control-plane`	Enters control-plane configuration mode (which is a prerequisite for defining control plane services).
Step 4	**service-policy** {**input** | **output** *policy-map-name*} **Example:** `Device(config-cp)# service-policy input control-plane-policy`	Attaches a QoS service policy to the control plane. • **input**—Applies the specified service policy to packets received on the control plane. • **output**—Applies the specified service policy to packets transmitted from the control plane and enables the device to silently discard packets. • *policy-map-name*—Name of a service policy map (created using the **policy-map** command) to be attached.
Step 5	**end** **Example:** `Device(config-cp)# end`	(Optional) Returns to privileged EXEC mode.

Verifying Control Plane Services

SUMMARY STEPS

1. **enable**
2. **show policy-map control-plane** [**all**] [**input** [**class** *class-name*] | **output** [*class class-name*]]
3. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show policy-map control-plane** [**all**] [**input** [**class** *class-name*]	**output** [*class class-name*]] **Example:** `Device# show policy-map control-plane all`	Displays information about the control plane. • **all**—(Optional) Displays service policy information about all QoS policies used on the CP. • **input**—(Optional) Displays statistics for the attached input policy. • **output**—(Optional) Displays statistics for the attached output policy. • **class** *class-name*—(Optional) Specifies the name of the traffic class whose configuration and statistics are displayed.
Step 3	**exit** **Example:** `Device# exit`	(Optional) Exits privileged EXEC mode.

Examples

The following example shows that the policy map TEST is associated with the control plane. This policy map polices traffic that matches the class map TEST, while allowing all other traffic (that matches the class map "class-default") to go through as is.

```
Device# show policy-map control-plane

Control Plane
Service-policy input:TEST
Class-map:TEST (match-all)
      20 packets, 11280 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:access-group 101
      police:
        8000 bps, 1500 limit, 1500 extended limit
        conformed 15 packets, 6210 bytes; action:transmit
```

```
        exceeded 5 packets, 5070 bytes; action:drop
        violated 0 packets, 0 bytes; action:drop
        conformed 0 bps, exceed 0 bps, violate 0 bps
Class-map:class-default (match-any)
      105325 packets, 11415151 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:any
```

Configuring Control Plane Policing to Mitigate Denial-of-Service Attacks

Apply control plane policing (CoPP) to RSVP packets to mitigate denial of service (DoS) attacks.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **access-list** *access-list-number* **permit** *protocol* {**any | host** {*address* | *name*}} {**any | host** {*address* | *name*}}
4. **access-list** *access-list-number* **permit** *protocol* {**tcd | udp**} {**any | host** {*source-addr* | *name*}} **eq** *port number* {**any | host** {*source-addr* | *name*}} **eq** *port number*
5. **class-map** *class-map-name*
6. **match access-group** *access-list-index*
7. **exit**
8. **policy-map** *policy-map-name*
9. **class** *class-map-name*
10. **police rate** *units* **pps**
11. **conform-action** *action*
12. **exit**
13. **exit**
14. **control plane** [**host | transit | cef-exception**]
15. **service-policy** {**input** | **output**} *policy-map-name*
16. **exit**
17. **exit**
18. **show control-plane** {**aggregate** | **cef-exception** | **counters** | **features** | **host** | **transit**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**access-list** *access-list-number* **permit** *protocol* {**any** | **host** {*address* | *name*}} {**any** | **host** {*address* | *name*}} **Example:** `Device(config)# access-list 140 permit 46 any any`	Configures an access list for filtering frames by protocol type.
Step 4	**access-list** *access-list-number* **permit** *protocol* {**tcd** | **udp**} {**any** | **host** {*source-addr* | *name*}} **eq** *port number* {**any** | **host** {*source-addr* | *name*}} **eq** *port number* **Example:** `Device(config)# access-list 141 permit udp any eq 1699 any eq 1698`	Configures an access list for filtering frames by UDP protocol and matches only packets with a given port number.
Step 5	**class-map** *class-map-name* **Example:** `Device(config)# class-map match-any MyClassMap`	Creates a class-map and enters QoS class-map configuration mode.
Step 6	**match access-group** *access-list-index* **Example:** `Device(config-cmap)# match access-group 140`	Specifies access groups to apply to an identity policy. The range of valid values is 1-2799.
Step 7	**exit** **Example:** `Device(config-cmap)# exit`	Exits QoS class-map configuration mode and returns to global configuration mode.
Step 8	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map Policy1`	Specifies a service policy and enters QoS policy-map configuration mode.
Step 9	**class** *class-map-name* **Example:** `Device(config-pmap-)# class MyClassMap`	Enters QoS policy-map class configuration more
Step 10	**police rate** *units* **pps** **Example:** `Device(config-pmap-c)# police rate 10 pps`	Polices traffic destined for the control plane at a specified rate.
Step 11	**conform-action** *action* **Example:** `Device(config-pmap-c-police)# conform-action transmit`	(Optional) Specifies the action to take on packets that conform to the police rate limit and enters policy-map class police configuration mode.
Step 12	**exit** **Example:** `Device(config-pmap-c-police)# exit`	Exits policy-map class police configuration mode

	Command or Action	Purpose
Step 13	**exit** **Example:** `Device(config-pmap-)# exit`	Exits policy-map class configuration mode
Step 14	**control plane** [**host** | **transit** | **cef-exception**] **Example:** `Device(config)# control-plane`	Associates or modifies attributes (such as a service policy) that are associated with the control plane of the device and enters control plane configuration mode.
Step 15	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Device(config-cp)# service-policy input Policy1`	Attaches a policy map to a control plane.
Step 16	**exit** **Example:** `Device(config-cp)# exit`	Exits control plane configuration mode and returns to global configuration mode.
Step 17	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode returns to privileged EXEC mode.
Step 18	**show control-plane** {**aggregate** | **cef-exception** | **counters** | **features** | **host** | **transit**} **Example:** `Device# show control-plane features`	Displays the configured control plane features

Configuration Examples for Control Plane Policing

Example: Configuring Control Plane Policing on Input Telnet Traffic

The following example shows how to apply a QoS policy for aggregate control plane services to Telnet traffic that is received on the control plane. Trusted hosts with source addresses 10.1.1.1 and 10.1.1.2 forward Telnet packets to the control plane without constraint while allowing all remaining Telnet packets to be policed at the specified rate.

```
! Allow 10.1.1.1 trusted host traffic.
Device(config)# access-list 140 deny tcp host 10.1.1.1 any eq telnet

! Allow 10.1.1.2 trusted host traffic.
Device(config)# access-list 140 deny tcp host 10.1.1.2 any eq telnet

! Rate-limit all other Telnet traffic.
Device(config)# access-list 140 permit tcp any any eq telnet
! Define class-map "telnet-class."
Device(config)# class-map telnet-class

Device(config-cmap)# match access-group 140
```

```
Device(config-cmap)# exit
Device(config)# policy-map control-plane-in
Device(config-pmap)# class telnet-class
Device(config-pmap-c)# police 80000 conform transmit exceed drop
Device(config-pmap-c)# exit
Device(config-pmap)# exit
! Define aggregate control plane service for the active route processor.
Device(config)# control-plane
Device(config-cp)# service-policy input control-plane-in
Device(config-cp)# end
```

Example: Configuring Control Plane Policing on Output ICMP Traffic

The following example shows how to apply a QoS policy for aggregate control plane services to Telnet traffic transmitted from the control plane. Trusted networks with source addresses 10.0.0.0 and 10.0.0.1 receive Internet Control Management Protocol (ICMP) port-unreachable responses without constraint while allowing all remaining ICMP port-unreachable responses to be dropped.

```
! Allow 10.0.0.0 trusted network traffic.
Device(config)# access-list 141 deny icmp 10.0.0.0 0.0.0.255 any port-unreachable

! Allow
10.0.0.1
 trusted network traffic.
Device(config)# access-list 141 deny icmp 10.0.0.1 0.0.0.255 any port-unreachable

! Rate-limit all other ICMP traffic.
Device(config)# access-list 141 permit icmp any any port-unreachable
Device(config)# class-map icmp-class

Device(config-cmap)# match access-group 141
Device(config-cmap)# exit
Device(config)# policy-map control-plane-out
! Drop all traffic that matches the class "icmp-class."
Device(config-pmap)# class icmp-class
Device(config-pmap-c)# drop
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# control-plane
! Define aggregate control plane service for the active route processor.
Device(config-cp)# service-policy output control-plane-out
Device(config-cp)# end
```

Example: Marking Output Control Plane Packets

The following example shows how to apply a QoS policy on the control plane to mark all egress IPv6 echo-request packets with IPv6 precedence 6.

```
! Match all IPv6 Echo Requests
Device(config)# ipv6 access-list coppacl-ipv6-icmp-request
Device(config-ipv6-acl)# permit icmp any any echo-request
Device(config-ipv6-acl)# exit
Device(config)# class-map match-all coppclass-ipv6-icmp-request
Device(config-cmap)# match access-group name coppacl-ipv6-icmp-request
Device(config-cmap)# exit
! Set all egress IPv6 Echo Requests with precedence 6
Device(config)# policy-map copp-policy
```

```
Device(config-pmap)# class coppclass-ipv6-icmp-request
Device(config-pmap-c)# set precedence 6
Device(config-pmap-c)# exit
Device(config-pmap)# exit
! Define control plane service for the active route processor.
Device(config)# control-plane
Device(config-cp)# service-policy output copp-policy
Device(config-cp)# end
```

Example: Configuring Control Plane Policing to Mitigate Denial-of-Service Attacks

The following example shows how to configure control plane policing (CoPP) to police RSVP packets at a specified rate and displays configured CoPP features.

```
Device> enable
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# access-list 140 permit 46 any any
Device(config)# access-list 141 permit adp any eq 1699 any eq 1698
Device(config)# class-map match-any MyClassMap
Device(config-cmap)# match access-group 140
Device(config-cmap)# match access-group 141
Device(config-cmap)# exit
Device(config)# policy-map Policy1
Device(config-pmap)# class MyClassMap
Device(config-pmap-c)# police rate 10 pps
Device(config-pmap-c-police)# conform-action transmit
Device(config-pmap-c-police)# exceed-action drop
Device(config-pmap-c-police)# exit
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# control-plane
Device(config-cp)# service-policy input Policy1
Device(config-cp)#
*Sep 14 08:07:39.898: %CP-5-FEATURE: Control-plane Policing feature enabled on Control plane
 aggregate path
Device(config-cp)#
Device(config-c p)# exit
Device(config)# exit
Device#
*Sep 14 08:09:04.154: %SYS-5-CONFIG_I: Configured from console by console
Device# show control-plane features
Total 1 features configured

Control plane aggregate path features :

------------------------------------------------------------
Control-plane Policing activated Sep 14 2012 08:0

------------------------------------------------------------
```

Enabling QoS Policing and Matching for PPPoE Traffic on the Input Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos punt-path-matching**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**platform qos punt-path-matching** **Example:** `Device(config)# platform qos punt-path-matching`	Enables QoS policing and matching for PPPoE traffic on the input interface.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Returns to privileged EXEC mode.

Disabling QoS Policing and Matching for PPPoE Traffic on the Input Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no platform qos punt-path-matching**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no platform qos punt-path-matching** **Example:** `Device(config)# no platform qos punt-path-matching`	Disables QoS policing and matching for PPPoE traffic on the input interface.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Returns to privileged EXEC mode.

Example: Configuring PPPoE and PPPoE Discovery Packets on the Input Interface and Control Plane

The following example shows how to configure PPPoE and PPPoE discovery packets on the input interface and control plane:

```
Device#configure terminal
Device(config)#class-map pppoed
Device(config-cmap)#match protocol pppoe-discovery
Device(config-cmap)#class-map pppoe
Device(config-cmap)#match protocol pppoe
Device(config-cmap)#policy-map pppoe-input
Device(config-pmap)#class pppoed

Device(config-pmap-c)#police 10000
Device(config-pmap-c-police)#class pppoe
Device(config-pmap-c)#police 10000
Device(config-pmap-c-police)#int g0/0/0.100
Device(config-subif)#service-p input pppoe-input

Device(config-subif)#end

Device#show platform hardware qfp active feature qos config global

Punt-Path-Matching are: enabled
```

Additional References for Control Plane Policing

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	Cisco IOS Quality of Service Solutions Command Reference
QoS features overview	"Quality of Service Overview" module
MQC	"Applying QoS Features Using the MQC" module
Security features overview	"Security Overview" module

MIBs

MIB	MIBs Link
CISCO-CLASS-BASED-QOS-MIB	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

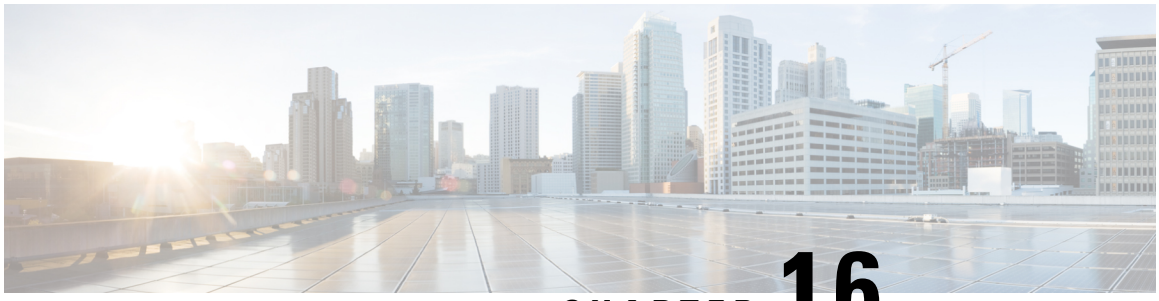
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Control Plane Policing

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 19: Feature Information for Control Plane Policing



CHAPTER 16

Management Plane Protection

First Published: February 27, 2006

Last Updated: February 27, 2006

The Management Plane Protection (MPP) feature in Cisco IOS software provides the capability to restrict the interfaces on which network management packets are allowed to enter a device. The MPP feature allows a network operator to designate one or more router interfaces as management interfaces. Device management traffic is permitted to enter a device only through these management interfaces. After MPP is enabled, no interfaces except designated management interfaces will accept network management traffic destined to the device.

Restricting management packets to designated interfaces provides greater control over management of a device, providing more security for that device. Other benefits include improved performance for data packets on nonmanagement interfaces, support for network scalability, need for fewer access control lists (ACLs) to restrict access to a device, and management packet floods on switching and routing interfaces are prevented from reaching the CPU.

Finding Feature Information in This Module

Your Cisco IOS software release may not support all of the features documented in this module. For a list of the releases in which a feature is supported, see Feature Information for Management Plane Protection, on page 174.

Finding Support Information for Platforms and Cisco IOS Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support. Access Cisco Feature Navigator at http://www.cisco.com/go/fn . You must have an account on Cisco.com. If you do not have an account or have forgotten your username or password, click **Cancel** at the login dialog box and follow the instructions that appear.

Prerequisites for Management Plane Protection

- You must enable IP Cisco Express Forwarding before you configure a management interface.
- You must explicitly enable the protocols that you wish to allow under the control-pane configuration mode. To do so, execute the following command:

```
control-plane host
management-interface <interface> allow <protocol>
```

Restrictions for Management Plane Protection

- Management Plane Protection is not a stateful feature.
- This feature only supports well-known ports. For example, port 443 for HTTPS. To see a list of common, well-known ports, see Well-Known Ports.
- Out-of-band management interfaces (also called dedicated management interfaces) are not supported. An out-of-band management interface is a dedicated Cisco IOS physical or logical interface that processes management traffic only.
- Loopback and virtual interfaces not associated to physical interfaces are not supported.
- Fallback and standby management interfaces are not supported.
- Hardware-switched and distributed platforms are not supported.
- Secure Copy (SCP) is supported under the Secure Shell (SSH) Protocol and not directly configurable in the command-line interface (CLI).
- Uninformed management stations lose access to the router through nondesignated management interfaces when the Management Plane Protection feature is enabled.
- This feature supports only IPv4 traffic. IPv6 traffic is neither blocked nor denied.

Information About Management Plane Protection

Before you enable the Management Plane Protection feature, you should understand the following concepts:

In-Band Management Interface

An in-band management interface is a Cisco IOS physical or logical interface that processes management as well as data-forwarding packets. Loopback interfaces commonly are used as the primary port for network management packets. External applications communicating with a networking device direct network management requests to the loopback port. An in-band management interface is also called a shared management interface.

Control Plane Protection Overview

A control plane is a collection of processes that run at the process level on a route processor and collectively provide high-level control for most Cisco IOS software functions. All traffic directly or indirectly destined to a router is handled by the control plane.

Control Plane Policing (CoPP) is a Cisco IOS control-plane feature that offers rate limiting of all control-plane traffic. CoPP allows you to configure a quality of service (QoS) filter that manages the traffic flow of control plane packets. This QoS filter helps to protect the control plane of Cisco IOS routers and switches against denial-of-service (DoS) attacks and helps to maintain packet forwarding and protocol states during an attack or during heavy traffic loads.

Control Plane Protection is a framework that encompasses all policing and protection features in the control plane. The Control Plane Protection feature extends the policing functionality of the CoPP feature by allowing finer policing granularity. Control Plane Protection also includes a traffic classifier, which intercepts control-plane traffic and classifies it in control-plane categories. Management Plane Protection operates within the Control Plane Protection infrastructure.

For more information about the Control Plane Policing feature in Cisco IOS software, see the Control Plane Policing module.

For more information about the Control Plane Protection feature in Cisco IOS software, see the Control Plane Protection module .

Management Plane

The management plane is the logical path of all traffic related to the management of a routing platform. One of three planes in a communication architecture that is structured in layers and planes, the management plane performs management functions for a network and coordinates functions among all the planes (management, control, data). The management plane also is used to manage a device through its connection to the network.

Examples of protocols processed in the management plane are Simple Network Management Protocol (SNMP), Telnet, HTTP, Secure HTTP (HTTPS), and SSH. These management protocols are used for monitoring and for CLI access. Restricting access to devices to internal sources (trusted networks) is critical.

Management Plane Protection Feature

The MPP feature in Cisco IOS software provides the capability to restrict the interfaces on which network management packets are allowed to enter a device. The MPP feature allows a network operator to designate one or more router interfaces as management interfaces. Device management traffic is permitted to enter a device through these management interfaces. After MPP is enabled, no interfaces except designated management interfaces will accept network management traffic destined to the device. Restricting management packets to designated interfaces provides greater control over management of a device.

The MPP feature is disabled by default. When you enable the feature, you must designate one or more interfaces as management interfaces and configure the management protocols that will be allowed on those interfaces. The feature does not provide a default management interface. Using a single CLI command, you can configure, modify, or delete a management interface.When you configure a management interface, no interfaces except that management interface will accept network management packets destined to the device. When the last configured interface is deleted, the feature turns itself off.

Following are the management protocols that the MPP feature supports. These management protocols are also the only protocols affected when MPP is enabled.

- Blocks Extensible Exchange Protocol (BEEP)
- FTP
- HTTP
- HTTPS
- SSH, v1 and v2
- SNMP, all versions
- Telnet
- TFTP

Cisco IOS features enabled on management interfaces remain available when the MPP feature is enabled. Nonmanagement packets such as routing and Address Resolution Protocol (ARP) messages for in-band management interfaces are not affected.

This feature generates a syslog for the following events:

- When the feature is enabled or disabled
- When a management interface fails.

For example, a failure will occur when the management interface cannot successfully receive or process packets destined for the control plane for reasons other than resource exhaustion.

Benefits of the Management Plane Protection Feature

Implementing the MPP feature provides the following benefits:

- Greater access control for managing a device than allowing management protocols on all interfaces
- Improved performance for data packets on nonmanagement interfaces
- Support for network scalability
- Simplifies the task of using per-interface ACLs to restrict management access to the device
- Fewer ACLs needed to restrict access to the device
- Management packet floods on switching and routing interfaces are prevented from reaching the CPU

How to Configure a Device for Management Plane Protection

This section contains the following task:

Configuring a Device for Management Plane Protection

Perform this task to configure a device that you have just added to your network or a device already operating in your network. This task shows how to configure MPP where SSH and SNMP are allowed to access the router only through the FastEthernet 0/0 interface.

Before you begin

- IP Cisco Express Forwarding must be enabled before a management interface can be configured.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **control-plane host**
4. **management-interface** *interface* **allow** *protocols*
5. **Ctrl z**
6. **show management-interface** [*interface* | **protocol** *protocol-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**control-plane host** **Example:** `Router(config)# control-plane host`	Enters control-plane host configuration mode.

Step 4	**management-interface** *interface* **allow** *protocols*
	Example:
	`Router(config-cp-host)# management-interface FastEthernet 0/0 allow ssh snmp`
		Configures an interface to be a management interface, which will accept management protocols, and specifies which management protocols are allowed.
		• *interface*—Name of the interface that you are designating as a management interface. You can also configure a virtual template interface.
		• *protocols*—Management protocols you want to allow on the designated management interface.
		  • BEEP
		  • FTP
		  • HTTP
		  • HTTPS
		  • SSH, v1 and v2
		  • SNMP, all versions
		  • Telnet
		  • TFTP

Step 5	**Ctrl z** **Example:** `Router(config-cp-host)# Ctrl z`	Returns to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show management-interface** [*interface* | **protocol** *protocol-name*] **Example:** `Router# show management-interface FastEthernet 0/0`	Displays information about the management interface such as type of interface, protocols enabled on the interface, and number of packets dropped and processed. *interface*—(Optional) Interface for which you want to view information. **protocol**—(Optional) Indicates that a protocol is specified. *protocol-name*—(Optional) Protocol for which you want to view information

Examples

The configuration in this example shows MPP configured to allow SSH and SNMP to access the router only through the FastEthernet 0/0 interface. This configuration results in all protocols in the remaining subset of supported management protocols to be dropped on all interfaces unless explicitly permitted. BEEP, FTP, HTTP, HTTPS, Telnet, and TFTP will not be permitted to access the router through any interfaces, including FastEthernet 0/0. Additionally, SNMP and SSH will be dropped on all interfaces except FastEthernet 0/0, where it is explicitly allowed.

To allow other supported management protocols to access the router, you must explicitly allow these protocols by adding them to the protocol list for the FastEthernet 0/0 interface or enabling additional management interfaces and protocols.

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# control-plane host
Router(config-cp-host)# management-interface FastEthernet 0/0 allow ssh snmp
Router(config-cp-host)#
.Aug  2 15:25:32.846: %CP-5-FEATURE: Management-Interface feature enabled on Control plane
 host path
Router(config-cp-host)#
```

The following is output from the **show management-interface** command issued after configuring MPP in the previous example. The **show management-interface** command is useful for verifying your configuration.

```
Router# show management-interface
Management interface FastEthernet0/0
        Protocol        Packets processed
             ssh                0
            snmp                0
Router#
```

Configuration Examples for Management Plane Protection

This section provides the following configuration example:

Configuring Management Plane Protection on Gigabit Ethernet Interfaces: Example

The following example shows how to configure MPP where only SSH, SNMP, and HTTP are allowed to access the router through the Gigabit Ethernet 0/3 interface and only HTTP is allowed to access the router through the Gigabit Ethernet 0/2 interface.

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# control-plane host
Router(config-cp-host)# management-interface GigabitEthernet 0/3 allow http ssh snmp

Router(config-cp-host)#
.Aug  2 17:00:24.511: %CP-5-FEATURE: Management-Interface feature enabled on Control plane
 host path
Router(config-cp-host)# management-interface GigabitEthernet 0/2 allow http
Router(config-cp-host)#
```

The following is output from the **show management-interface** command issued after configuring MPP in the previous example. The **show management-interface** command is useful for verifying your configuration.

```
Router# show management-interface

Management interface GigabitEthernet0/2
        Protocol        Packets processed
            http                0
Management interface GigabitEthernet0/3
        Protocol        Packets processed
            http                0
             ssh                0
            snmp                0
```

Additional References for Management Plane Protection

The following sections provide references related to Management Plane Protection.

Related Documents

Related Topic	Document Title
Network management	Cisco IOS Network Management Configuration Guide
Network security	Cisco IOS Security Configuration Guide
Control Plane Policing	Control Plane Policing module
Control Plane Protection	Control Plane Protection module

RFCs

RFC	Title
RFC 3871	Operational Security Requirements for Large Internet Service Provider (ISP) IP Network Infrastructure

Technical Assistance

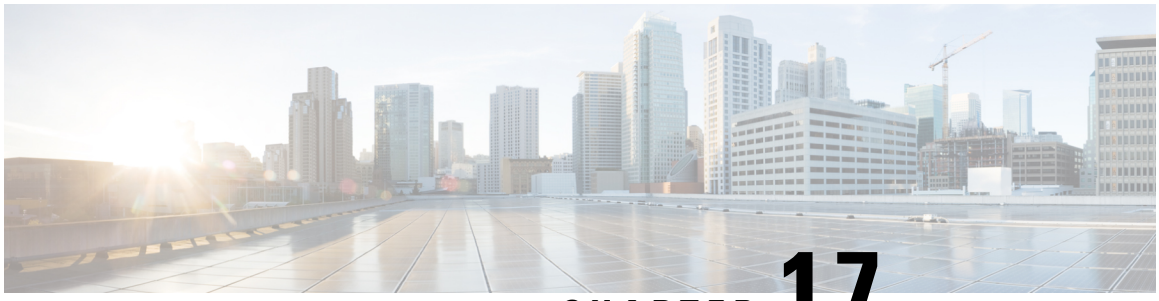
Description	Link
The Cisco Technical Support and Documentation website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Feature Information for Management Plane Protection

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Feature Information for Management Plane Protection



CHAPTER 17

Class-Based Policing

Class-based policing allows you to control the maximum rate of traffic that is transmitted or received on an interface. Class-based policing is often configured on interfaces at the edge of a network to limit traffic into or out of the network.

Information About Class-Based Policing

Class-Based Policing Functionality

The Class-Based Policing feature performs the following functions:

- Limits the input or output transmission rate of a class of traffic based on user-defined criteria.
- Marks packets by setting the ATM Cell Loss Priority (CLP) bit, Frame Relay Discard Eligibility (DE) bit, IP precedence value, IP differentiated services code point (DSCP) value, MPLS experimental value, and quality of service (QoS) group.

Class-based policing allows you to control the maximum rate of traffic transmitted or received on an interface. The Class-Based Policing feature is applied when you attach a traffic policy that contains the class-based policing configuration to an interface.

The Class-Based Policing feature works with a token bucket mechanism. There are currently two types of token bucket algorithms: a single token bucket algorithm and a two-token bucket algorithm. A single token bucket system is used when the **violate-action** option is not specified, and a two-token bucket system is used when the **violate-action** option is specified.

Benefits of Class-Based Policing

Bandwidth Management Through Rate Limiting

Class-based policing allows you to control the maximum rate of traffic transmitted or received on an interface. Class-based policing is often configured on interfaces at the edge of a network to limit traffic into or out of the network. In most class-based policing configurations, traffic that falls within the rate parameters is transmitted, whereas traffic that exceeds the parameters is dropped or transmitted with a different priority.

Packet Marking

Packet marking allows you to partition your network into multiple priority levels or classes of service (CoS). A packet is marked and these markings can be used to identify and classify traffic for downstream devices.

- Use class-based policing to set the IP precedence or DSCP values for packets entering the network. Networking devices within your network can then use the adjusted IP precedence values to determine how the traffic should be treated.
- Use class-based policing to assign packets to a QoS group. The router uses the QoS group to determine how to prioritize packets.

Traffic can be marked without using the Class-Based Policing feature. If you want to mark traffic but do not want to use class-based policing, see the "Marking Network Traffic" module.

Restrictions for Class-Based Policing

Class-based policing can be configured on an interface or a subinterface, but it is not supported on EtherChannel or tunnel interfaces.

How to Configure Class-Based Policing

Configuring a Traffic Policing Service Policy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** [**match-all** | **match-any**] *class-map-name*
4. **match ip precedence** *precedence-value*
5. **exit**
6. **policy-map** *policy-map-name*
7. **class** {*class-name* | **class-default**}
8. **police** *bps burst-normal burst-max* **conform-action** *action* **exceed-action** *action* **violate-action** *action*
9. **exit**
10. **exit**

11. **interface** *interface-type interface-number*
12. **service-policy** {**input** | **output**} *policy-map-name*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** [**match-all** | **match-any**] *class-map-name* **Example:** `Router(config)# class-map match-any MATCH_PREC`	Specifies the name of the class map to be created and enters QoS class map configuration mode. • The class map defines the criteria to use to differentiate the traffic. For example, you can use the class map to differentiate voice traffic from data traffic, based on a series of match criteria defined using the **match** command. **Note** If the **match-all** or **match-any** keyword is not specified, traffic must match all the match criteria to be classified as part of the traffic class.
Step 4	**match ip precedence** *precedence-value* **Example:** `Router(config-cmap)# match ip precedence 0`	Enables packet matching on the basis of the IP precedence values you specify. **Note** You can enter up to four matching criteria, as number abbreviation (0 to 7) or criteria names (critical, flash, and so on), in a single match statement.
Step 5	**exit** **Example:** `Router(config-cmap)# exit`	Returns to global configuration mode.
Step 6	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map POLICE-SETTING`	Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy, and enters QoS policy-map configuration mode.

	Command or Action	Purpose
Step 7	**class** {*class-name* | **class-default**} **Example:** `Router(config-pmap)# class MATCH_PREC`	Specifies the name of the class whose policy you want to create or change or specifies the default class (commonly known as the class-default class) before you configure its policy, and enters policy-map class configuration mode.
Step 8	**police** *bps burst-normal burst-max* **conform-action** *action* **exceed-action** *action* **violate-action** *action* **Example:** `Router(config-pmap-c)# police 8000 1000 1000 conform-action transmit exceed-action set-qos-transmit 1 violate-action drop`	Configures traffic policing according to burst sizes and any optional actions specified.
Step 9	**exit** **Example:** `Router(config-pmap-c)# exit`	(Optional) Exits policy-map class configuration mode.
Step 10	**exit** **Example:** `Router(config-pmap)# exit`	(Optional) Exits QoS policy-map configuration mode.
Step 11	**interface** *interface-type interface-number* **Example:** `Router(config)# interface GigabitEthernet 0/0/1`	Configures an interface type and enters interface configuration mode. • Enter the interface type and interface number.
Step 12	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# service-policy input POLICE-SETTING`	Attaches a policy map to an interface. • Enter either the **input** or **output** keyword and the policy map name.
Step 13	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode and returns to privileged EXEC mode.

Monitoring and Maintaining Traffic Policing

SUMMARY STEPS

1. **enable**
2. **show policy-map**
3. **show policy-map** *policy-map-name*

4. **show policy-map interface**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show policy-map** **Example:** `Router# show policy-map`	Displays all configured policy maps.
Step 3	**show policy-map** *policy-map-name* **Example:** `Router# show policy-map pmap`	Displays the user-specified policy map.
Step 4	**show policy-map interface** **Example:** `Router# show policy-map interface`	Verifies that the Class-Based Policing feature is configured on your interface. If the feature is configured on your interface. • The command output displays policing statistics.

Verifying Class-Based Traffic Policing

Use the **show policy-map interface** command to verify that the Class-Based Policing feature is configured on your interface. If the feature is configured on your interface, the **show policy-map interface** command output displays policing statistics.

SUMMARY STEPS

1. **enable**
2. **show policy-map interface**
3. **show policy-map interface** *type interface*
4. **show policy-map interface** *type interface* **service instance** *service-instance number*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**show policy-map interface** **Example:** `Router# show policy-map interface`	Verifies that the Class-Based Policing feature is configured on your interface. If the feature is configured on your interface. • The command output displays policing statistics.
Step 3	**show policy-map interface** *type interface* **Example:** `Router# show policy-map interface GigabitEthernet 0/0/1`	Displays traffic statistics for policies applied to a specific interface.
Step 4	**show policy-map interface** *type interface* **service instance** *service-instance number* **Example:** `Router# show policy-map interface GigabitEthernet 0/0/1 service instance 1`	Displays the policy map information for a given service instance under an interface.
Step 5	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Example: Verifying Class-Based Traffic Policing

```
Router# show policy-map interface
  FastEthernet1/1/1
  service-policy output: x
    class-map: a (match-all)
      0 packets, 0 bytes
      5 minute rate 0 bps
      match: ip precedence 0
      police:
        1000000 bps, 10000 limit, 10000 extended limit
        conformed 0 packets, 0 bytes; action: transmit
        exceeded 0 packets, 0 bytes; action: drop
        conformed 0 bps, exceed 0 bps, violate 0 bps
```

Troubleshooting Tips

Check the interface type. Verify that class-based policing is supported on your interface.

Configuration Examples for Class-Based Policing

Example Configuring a Service Policy That Includes Traffic Policing

In the following example, class-based policing is configured with the average rate at 8000 bits per second, the normal burst size at 1000 bytes, and the excess burst size at 1000 bytes for all packets leaving the interface.

```
class-map access-match
 match access-group 1
 exit
policy-map police-setting
 class access-match
  police 8000 1000 1000 conform-action transmit exceed-action set-qos-transmit 1
violate-action drop
  exit
 exit
 service-policy output police-setting
```

The treatment of a series of packets leaving FastEthernet interface 1/1/1 depends on the size of the packet and the number of bytes remaining in the conform and exceed token buckets. The series of packets are policed based on the following rules:

- If the previous arrival of the packet was at T1 and the current arrival of the packet is at T, the bucket is updated with T - T1 worth of bits based on the token arrival rate. The refill tokens are placed in the conform bucket. If the tokens overflow the conform bucket, the overflow tokens are placed in the exceed bucket. The token arrival rate is calculated as follows:

(time between packets < which is equal to T - T1 > * policer rate)/8 bytes

- If the number of bytes in the conform bucket is greater than the length of the packet (for example, B), then the packet conforms and B bytes should be removed from the bucket. If the packet conforms, B bytes are removed from the conform bucket and the conform action is taken. The exceed bucket is unaffected in this scenario.
- If the number of bytes in the conform bucket is less than the length of the packet, but the number of bytes in the exceed bucket is greater than the length of the packet (for example, B), the packet exceeds and B bytes are removed from the bucket.
- If the number bytes in the exceed bucket B is fewer than 0, the packet violates the rate and the violate action is taken. The action is complete for the packet.

In this example, the initial token buckets starts full at 1000 bytes. If a 450-byte packet arrives, the packet conforms because enough bytes are available in the conform token bucket. The conform action (send) is taken by the packet, and 450 bytes are removed from the conform token bucket (leaving 550 bytes).

If the next packet arrives 0.25 seconds later, 250 bytes are added to the conform token bucket ((0.25 * 8000)/8), leaving 800 bytes in the conform token bucket. If the next packet is 900 bytes, the packet does not conform because only 800 bytes are available in the conform token bucket.

The exceed token bucket, which starts full at 1000 bytes (as specified by the excess burst size, is then checked for available bytes. Because enough bytes are available in the exceed token bucket, the exceed action (set the QoS transmit value of 1) is taken, and 900 bytes are taken from the exceed bucket (leaving 100 bytes in the exceed token bucket).

If the next packet arrives 0.40 seconds later, 400 bytes are added to the token buckets ((.40 * 8000)/8). Therefore, the conform token bucket now has 1000 bytes (the maximum number of tokens available in the conform bucket, and 200 bytes overflow the conform token bucket (because only 200 bytes were needed to fill the conform token bucket to capacity). These overflow bytes are placed in the exceed token bucket, giving the exceed token bucket 300 bytes.

If the arriving packet is 1000 bytes, the packet conforms because enough bytes are available in the conform token bucket. The conform action (transmit) is taken by the packet, and 1000 bytes are removed from the conform token bucket (leaving 0 bytes).

If the next packet arrives 0.20 seconds later, 200 bytes are added to the token bucket ((.20 * 8000)/8). Therefore, the conform bucket now has 200 bytes. If the arriving packet is 400 bytes, the packet does not conform because only 200 bytes are available in the conform bucket. Similarly, the packet does not exceed because only 300 bytes are available in the exceed bucket. Therefore, the packet violates and the violate action (drop) is taken.

Verifying Class-Based Traffic Policing

Use the **show policy-map interface** command to verify that the Class-Based Policing feature is configured on your interface. If the feature is configured on your interface, the **show policy-map interface** command output displays policing statistics:

```
Router# show policy-map interface
  FastEthernet1/1/1
  service-policy output: x
    class-map: a (match-all)
      0 packets, 0 bytes
      5 minute rate 0 bps
      match: ip precedence 0
      police:
        1000000 bps, 10000 limit, 10000 extended limit
        conformed 0 packets, 0 bytes; action: transmit
        exceeded 0 packets, 0 bytes; action: drop
        conformed 0 bps, exceed 0 bps, violate 0 bps
```

Use the **show policy-map interface** *type nummber* command to view the traffic statistics for policies applied to that specific interface:

```
Router# show policy-map interface gigabitethernet 0/0/1
 GigabitEthernet0/0/1

  Service-policy input: TUNNEL_MARKING

    Class-map: MATCH_PREC (match-any)
      72417 packets, 25418367 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: ip precedence 0
      QoS Set
        ip precedence tunnel 3
          Marker statistics: Disabled

    Class-map: MATCH_DSCP (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: ip dscp default (0)
      QoS Set
        ip dscp tunnel 3
          Marker statistics: Disabled
```

```
    Class-map: class-default (match-any)
      346462 packets, 28014400 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any

  Service-policy output: POLICE-SETTING

    Class-map: MATCH_PREC (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: ip precedence 0
      police:
          cir 8000 bps, bc 1000 bytes, be 1000 bytes
        conformed 0 packets, 0 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          set-qos-transmit 1
        violated 0 packets, 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps, violate 0000 bps

    Class-map: class-default (match-any)
      31 packets, 2019 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Use the **show policy-map interface service instance** command to view the traffic statistics for policy applied to the specific service instance in that specific interface:

```
Router# show policy-map interface gig0/0/1 service instance 10
 GigabitEthernet0/0/1: EFP 10

       Service-policy input: ac1

 Class-map: ac1 (match-all)
   0 packets, 0 bytes
   5 minute offered rate 0000 bps, drop rate 0000 bps
   Match: access-group 1
   police:
       cir 50000000 bps, bc 1562500 bytes
     conformed 0 packets, 0 bytes; actions:
       transmit
     exceeded 0 packets, 0 bytes; actions:
       drop
     conformed 0000 bps, exceeded 0000 bps

 Class-map: class-default (match-any)
   0 packets, 0 bytes
   5 minute offered rate 0000 bps, drop rate 0000 bps
   Match: any
```

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*

Related Topic	Document Title
Traffic marking	"Marking Network Traffic" module
Traffic policing	"Traffic Policing" module
Traffic policing and shaping concepts and overview information	"Policing and Shaping Overview"
Modular Quality of Service Command-Line Interface (MQC)	"Applying QoS Features Using the MQC" module

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
Class-Based Quality of Service MIB • CISCO-CLASS-BASED-QOS-MIB • CISCO-CLASS-BASED-QOS-CAPABILITY-MIB	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2697	*A Single Rate Three Color Marker*

Technical Assistance

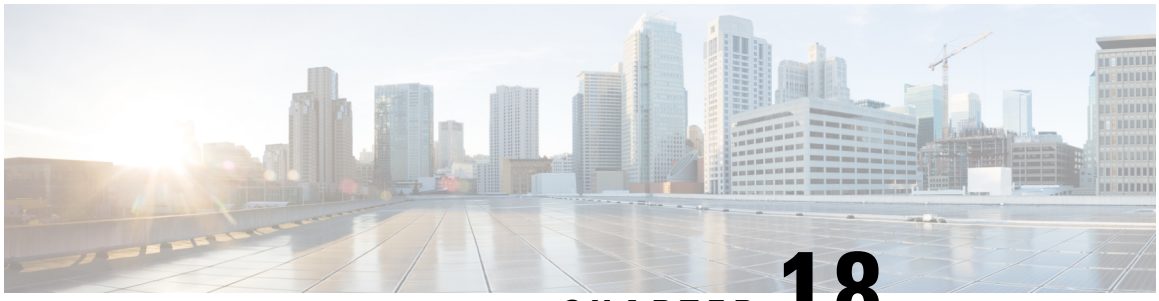
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Class-Based Policing

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 20: Feature Information for Class-Based Policing



CHAPTER 18

QoS Percentage-Based Policing

The QoS Percentage-Based Policing feature allows you to configure traffic policing and traffic shaping on the basis of a percentage of bandwidth available on the interface. This feature also allows you to specify the committed burst (bc) size and the excess burst (be) size (used for configuring traffic policing) in milliseconds (ms). Configuring traffic policing in this manner enables you to use the same policy map for multiple interfaces with differing amounts of bandwidth.

Information About QoS Percentage-Based Policing

Benefits for QoS Percentage-Based Policing

This feature provides the ability to configure traffic policing on the basis of a percentage of bandwidth available on an interface, and it allows you to specify burst sizes in milliseconds. Configuring traffic policing in this manner enables you to use the same policy map for multiple interfaces with differing amounts of bandwidth. That is, you do not have to recalculate the bandwidth for each interface or configure a different policy map for each type of interface.

Configuration of Class and Policy Maps for QoS Percentage-Based Policing

To configure the QoS: Percentage-Based Policing feature, you must define a traffic class, configure a policy map, and then attach that policy map to the appropriate interface.

The MQC is a command-line interface that allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach these traffic policies to interfaces.

In the MQC, the **class-map** command is used to define a traffic class (which is then associated with a traffic policy). The purpose of a traffic class is to classify traffic.

The MQC consists of the following three processes:

- Defining a traffic class with the **class-map** command.

- Creating a traffic policy by associating the traffic class with one or more QoS features (using the **policy-map** command).
- Attaching the traffic policy to the interface with the **service-policy** command.

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 (that is, match-all or match-any). The traffic class is named in the **class-map** command line; for example, if you enter the **class-map cisco** command while configuring the traffic class in the CLI, 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.

Traffic Regulation Mechanisms and Bandwidth Percentages

Cisco IOS XE quality of service (QoS) offers two kinds of traffic regulation mechanisms--traffic policing and traffic shaping. A traffic policer typically drops traffic that violates a specific rate. A traffic shaper typically delays excess traffic using a buffer to hold packets and shapes the flow when the data rate to a queue is higher than expected.

Traffic shaping and traffic policing can work in tandem and can be configured in a class map. Class maps organize data packets into specific categories ("classes") that can, in turn, receive a user-defined QoS treatment when used in policy maps (sometimes referred to as "service policies").

Before this feature, traffic policing and traffic shaping were configured on the basis of a user-specified amount of bandwidth available on the interface. Policy maps were then configured on the basis of that specific amount of bandwidth, meaning that separate policy maps were required for each interface.

This feature provides the ability to configure traffic policing and traffic shaping on the basis of a *percentage* of bandwidth available on the interface. Configuring traffic policing and traffic shaping in this manner enables customers to use the same policy map for multiple interfaces with differing amounts of bandwidth.

Configuring traffic policing and shaping on the basis of a percentage of bandwidth is accomplished by using the **police** (percent) and **shape** (percent) commands.

Burst Size in Milliseconds Option

The purpose of the burst parameters (bc and be) is to drop packets gradually and to avoid tail drop. Setting sufficiently high burst values helps to ensure good throughput.

This feature allows you the option of specifying the committed burst (bc) size and the extended burst (be) as milliseconds (ms) of the class bandwidth when you configure traffic policing. The number of milliseconds is used to calculate the number of bytes that will be used by the QoS: Percentage-Based Policing feature.

Specifying these burst sizes in milliseconds is accomplished by using the **bc** and **be** keywords (and their associated arguments) of the **police** (percent) and **shape** (percent) commands.

How to Configure QoS Percentage-Based Policing

Configuring a Class and Policy Map for Percentage-Based Policing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-name*
4. **class** {*class-name* **class-default**}
5. **police cir percent** *percentage* [*burst-in-ms*] [**bc** *conform-burst-in-msec* **ms**] [**be** *peak-burst-in-msec* **ms**] [**pir percent** *percent*]
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-name* **Example:** `Router(config)# policy-map policy1`	Specifies the name of the policy map to be created. Enters policy-map configuration mode. • Enter the policy map name.
Step 4	**class** {*class-name* **class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the class so that you can configure or modify its policy. Enters policy-map class configuration mode. • Enter the class name or specify the default class (class-default).
Step 5	**police cir percent** *percentage* [*burst-in-ms*] [**bc** *conform-burst-in-msec* **ms**] [**be** *peak-burst-in-msec* **ms**] [**pir percent** *percent*] **Example:** `Router(config-pmap-c)# police cir percent 20 bc 300 ms be 400 ms pir percent 40`	Configures traffic policing on the basis of the specified bandwidth percentage and optional burst sizes. Enters policy-map class police configuration mode. • Enter the bandwidth percentage and optional burst sizes.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config-pmap-c-police)# exit`	Exits policy-map class police configuration mode.

Attaching the Policy Map to an Interface for Percentage-Based Policing

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **pvc** [*name*] *vpi* / *vci* [**ilmi** | **qsaal** | **smds**]
5. **service-policy** {**input**| **output**} *policy-map-name*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**interface** *type number*
	Example:
	`Router(config)#`
	`interface serial4/0/0`
		Configures an interface (or subinterface) type and enters interface configuration mode.
		• Enter the interface type number.
		Note Depending on the needs of your network, you may need to attach the policy map to a subinterface, an ATM PVC, a Frame Relay DLCI, or other type of interface.

Step 4	**pvc** [*name*] *vpi* / *vci* [**ilmi**	**qsaal**	**smds**] **Example:** `Router(config-if)# pvc cisco 0/16 ilmi`	(Optional) Creates or assigns a name to an ATM PVC and specifies the encapsulation type on an ATM PVC. Enters ATM VC configuration mode.

	Command or Action	Purpose
		Note This step is required only if you are attaching the policy map to an ATM PVC. If you are not attaching the policy map to an ATM PVC, skip this step and proceed with Step 5.

Step 5	**service-policy** {**input**| **output**} *policy-map-name*
	Example:
	`Router(config-if)#`
	`service-policy input policy1`
	Example:
		Specifies the name of the policy map to be attached to the input or output direction of the interface.
		Note Policy maps can be configured on ingress or egress routers. They can also be attached in the input or output direction of an interface. The direction (input or output) and the router (ingress or egress) to which the policy map should be attached varies according your network configuration. When using the **service-policy** command to attach the policy map to an interface, be sure to choose the router and the interface direction that are appropriate for your network configuration.
		• Enter the policy map name.

Step 6	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode.

Verifying the Percentage-Based Policing Configuration

SUMMARY STEPS

1. **enable**
2. **show class-map** [*class-map-name*]
3. **show policy-map interface** *interface-name*
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show class-map** [*class-map-name*] **Example:**	Displays all information about a class map, including the match criterion.

	Command or Action	Purpose
	`Router# show class-map class1`	• Enter class map name.

Step 3	**show policy-map interface** *interface-name*
	Example:
	`Router#`
	`show policy-map interface serial4/0/0`
		Displays the packet statistics of all classes that are configured for all service policies either on the specified interface or subinterface or on a specific PVC on the interface.
		• Enter the interface name.

Step 4	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Troubleshooting Tips for Percentage-Based Policing

The commands in the Verifying the Percentage-Based Policing Configuration, on page 191 section allow you to verify that you achieved the intended configuration and that the feature is functioning correctly. If, after using the **show** commands listed above, you find that the configuration is not correct or the feature is not functioning as expected, perform these operations:

If the configuration is not the one you intended, complete the following procedures:

1. Use the **show running-config** command and analyze the output of the command.
2. If the policy map does not appear in the output of the **show running-config** command, enable the **logging console** command.
3. Attach the policy map to the interface again.

If the packets are not being matched correctly (for example, the packet counters are not incrementing correctly), complete the following procedures:

1. Run the **show policy-map**command and analyze the output of the command.
2. Run the **show running-config** command and analyze the output of the command.
3. Use the **show policy-map interface** command to verify that the policy map is attached to the interface and that the committed information rate (CIR) has been calculated on the basis of the percentage of the interface bandwidth.

Configuration Examples for QoS Percentage-Based Policing

Example Specifying Traffic Policing on the Basis of a Bandwidth Percentage

The following example configures traffic policing using a CIR and a peak information rate (PIR) on the basis of a percentage of bandwidth. In this example, a CIR of 20 percent and a PIR of 40 percent have been specified. Additionally, an optional bc value and be value (300 ms and 400 ms, respectively) have been specified.

```
Router> enable
Router# configure terminal
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# police cir percent 20 bc 300 ms be 400 ms pir percent 40

Router(config-pmap-c-police)# end
```

After the policy map and class maps are configured, the policy map is attached to interface as shown in the following example.

```
Router> enable
Router# configure terminal
Router(config-if)#

interface serial4/0/0
Router(config-if)#

service-policy input policy1
Router(config-if)# end
```

Example Verifying the Percentage-Based Policing Configuration

This section contains sample output from the **show policy-map interface** command and the **show policy-map** command. The output from these commands can be used to verify and monitor the feature configuration on your network.

The following is sample output from the **show policy-map** command. This sample output displays the contents of a policy map called "policy1." In policy 1, traffic policing on the basis of a CIR of 20 percent has been configured, and the bc and be have been specified in milliseconds. As part of the traffic policing configuration, optional conform, exceed, and violate actions have been specified.

```
Router# show policy-map policy1
  Policy Map policy1
    Class class1
     police cir percent 20 bc 300 ms pir percent 40 be 400 ms
       conform-action transmit
       exceed-action drop
       violate-action drop
```

The following is sample output from the **show policy-map interface** command. This sample displays the statistics for the serial 2/0 interface on which traffic policing has been enabled. The committed burst (bc) and excess burst (be) are specified in milliseconds (ms).

```
Router# show policy-map interface serial2/0
 Serial2/0/0
  Service-policy output: policy1 (1050)
    Class-map: class1 (match-all) (1051/1)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: ip precedence 0  (1052)
      police:
          cir 20 % bc 300 ms
          cir 409500 bps, bc 15360 bytes
          pir 40 % be 400 ms
          pir 819000 bps, be 40960 bytes
        conformed 0 packets, 0 bytes; actions:
          transmit
```

```
      exceeded 0 packets, 0 bytes; actions:
        drop
      violated 0 packets, 0 bytes; actions:
        drop
      conformed 0 bps, exceed 0 bps, violate 0 bps
  Class-map: class-default (match-any) (1054/0)
    0 packets, 0 bytes
    5 minute offered rate 0 bps, drop rate 0 bps
    Match: any  (1055)
      0 packets, 0 bytes
      5 minute rate 0 bps
```

In this example, the CIR and PIR are displayed in bps, and both the committed burst (bc) and excess burst (be) are displayed in bytes.

The CIR, PIR bc, and be are calculated on the basis of the formulas described below.

Formula for Calculating the CIR

When calculating the CIR, the following formula is used:

CIR percentage specified (as shown in the output of the **show policy-map**command) * bandwidth (BW) of the interface (as shown in the output of the **show interfaces** command) = total bits per second

On serial interface 2/0, the bandwidth (BW) is 2048 kbps. To see the bandwidth of the interface, use the **show interfaces** command. A sample is shown below:

```
Router# show interfaces serial2/0/0
Serial2/0/0 is administratively down, line protocol is down
  Hardware is M4T
  MTU 1500 bytes, BW 2048 Kbit, DLY 20000 usec, rely 255/255, load 1/255
```

The following values are used for calculating the CI:

20 % * 2048 kbps = 409600 bps

Formula for Calculating the PIR

When calculating the PIR, the following formula is used:

PIR percentage specified (as shown in the output of the **show policy-map**command) * bandwidth (BW) of the interface (as shown in the output of the **show interfaces** command) = total bits per second

On serial interface 2/0/0, the bandwidth (BW) is 2048 kbps. To see the bandwidth of the interface, use the **show interfaces** command. A sample is shown below:

```
Router# show interfaces serial2/0
Serial2/0/0 is administratively down, line protocol is down
  Hardware is M4T
  MTU 1500 bytes, BW 2048 Kbit, DLY 20000 usec, rely 255/255, load 1/255
```

The following values are used for calculating the PIR:

40 % * 2048 kbps = 819200 bps

Note Discrepancies between this total and the total shown in the output of the **show policy-map interface** command can be attributed to a rounding calculation or to differences associated with the specific interface configuration.

Formula for Calculating the Committed Burst (bc)

When calculating the bc, the following formula is used:

The bc in milliseconds (as shown in the **show policy-map** command) * the CIR in bits per seconds = total number bytes

The following values are used for calculating the bc:

(300 ms * 409600 bps) / 8 = 15360 bytes

Formula for Calculating the Excess Burst (be)

When calculating the bc and the be, the following formula is used:

The be in milliseconds (as shown in the **show policy-map** command) * the PIR in bits per seconds = total number bytes

The following values are used for calculating the be:

400 ms * 819200 bps = 40960 bytes

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular QoS Command-Line Interface (CLI) (MQC), including information about attaching policy maps	"Applying QoS Features Using the MQC" module
Traffic shaping and traffic policing	"Policing and Shaping Overview" module

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2697	*A Single Rate Three Color Marker*
RFC 2698	*A Two Rate Three Color Marker*

Technical Assistance

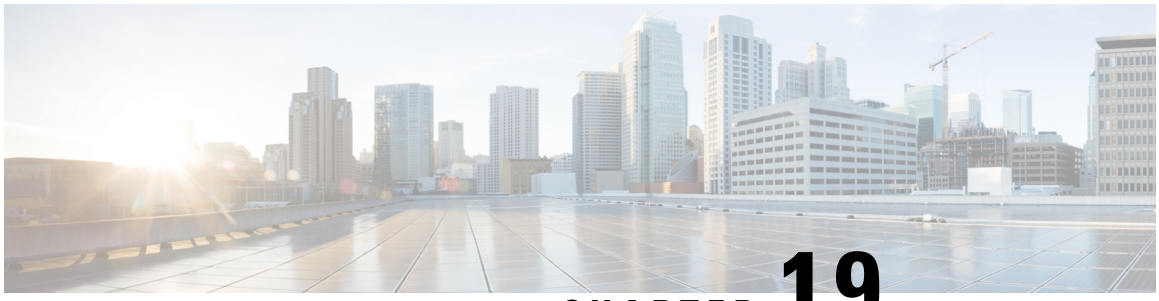
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for QoS Percentage-Based Policing

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 21: Feature Information for QoS: Percentage-Based Policing



CHAPTER 19

Port-Shaper and LLQ in the Presence of EFPs

The Port-Shaper and LLQ in the Presence of EFPs feature allows network designers to configure port and class policies on ports that contain Ethernet Flow Points (EFPs). These policies support Low Latency Queueing (LLQ) and traffic prioritization across the EFPs.

Restrictions for Port-Shaper and LLQ in the Presence of EFPs

- If you configure a class-based policy on the port, then you cannot configure service-policies on Ethernet Flow Points (EFPs).
- Attaching a service policy to the BDI is not supported.
- ACL based shaping policy-map cannot be applied to the EFP and/or egress interface.
- Usage of bandwidth remaining percentage (BRP) in the absence of priority class, allocates the available bandwidth in an iterative way. For example, the bandwidth is allocated for the first BRP class as per the percentage of share that is configured in the respective class-map and the remaining bandwidth is iteratively allocated to all other BRP classes until the bandwidth is exhausted.
-

Information About Port-Shaper and LLQ in the Presence of EFPs

Ethernet Flow Points and LLQ

An Ethernet Flow Point (EFP) is a forwarding decision point in the provider edge (PE) router, which gives network designers flexibility to make many Layer 2 flow decisions within the interface. Many EFPs can be configured on a single physical port. (The number varies from one device to another.) EFPs are the logical demarcation points of an Ethernet virtual connection (EVC) on an interface. An EVC that uses two or more

User-Network Interfaces (UNIs) requires an EFP on the associated ingress and egress interfaces of every device that the EVC passes through.

The Egress HQoS with Port Level Shaping feature allows network designers to configure port and class policies on ports that contain EFPs. These policies support Low Latency Queueing (LLQ) and traffic prioritization across the EFPs.

For information on how to configure LLQ, see the *QoS Congestion Management Configuration Guide*.

How to Configure Port-Shaper and LLQ in the Presence of EFPs

To configure the Port-Shaper and LLQ in the Presence of EFPs feature, you first create either a hierarchical or flat policy map that supports Low Latency Queueing (LLQ), which you then attach to an EFP interface.

Configuring Hierarchical Policy Maps

To configure hierarchical policy maps, you create child policies which you then attach to a parent policy. The parent policy is then attached to an interface.

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **policy-map** *policy-map-name*

Example:

```
Device(config)# policy-map child-llq
```

Creates or modifies the child policy and enters QoS policy-map configuration mode.

- child-llq is the name of the child policy map.

Step 4 **class** *class-map-name*

Example:

```
Device(config-pmap)# class precedenc-1
```

Assigns the traffic class you specify to the policy map and enters QoS policy-map class configuration mode.

- precedenc-1 is the name of a previously configured class map and is the traffic class for which you want to define QoS actions.

Step 5 **set cos** *value*

Example:

```
Device(config-pmap-c)# set cos 5
```

(Optional) Sets the Layer 2 class of service (CoS) value of an outgoing packet.

- The value is a specific IEEE 802.1Q CoS value from 0 to 7.

Step 6 **bandwidth percent** *percent*

Example:

```
Device(config-pmap-c)# bandwidth percent 20
```

(Optional) Specifies a bandwidth percent for class-level queues to be used during congestion to determine the amount of excess bandwidth (unused by priority traffic) to allocate to nonpriority queues.

Step 7 **exit**

Example:

```
Device(config-pmap-c)# exit
```

Exits QoS policy-map class configuration mode.

Step 8 **class** *class-map-name*

Example:

```
Device(config-pmap)# class precedenc-2
```

Assigns the traffic class you specify to the policy map and enters QoS policy-map class configuration mode.

- precedenc-2 is the name of a previously configured class map and is the traffic class for which you want to define QoS actions.

Step 9 **bandwidth percent** *percent*

Example:

```
Device(config-pmap-c)# bandwidth percent 80
```

(Optional) Specifies a bandwidth percent for class-level queues to be used during congestion to determine the amount of excess bandwidth (unused by priority traffic) to allocate to nonpriority queues.

Step 10 **exit**

Example:

```
Device(config-pmap-c)# exit
```

Exits QoS policy-map class configuration mode.

Step 11 **policy-map** *policy-map-name*

Example:

```
Device(config-pmap)# policy-map parent-llq
```

Creates or modifies the parent policy.

- parent-llq is the name of the parent policy map.

Step 12 **class** *class-default*

Example:

```
Device(config-pmap)# class class-default
```

Configures or modifies the parent class-default class and enters QoS policy-map class configuration mode.

- You can configure only the class-default class in a parent policy. Do not configure any other traffic class.

Step 13 **service-policy** *policy-map-name*

Example:

```
Device(config-pmap-c)# service-policy child-llq
```

Applies the child policy to the parent class-default class.

- child-llq is the name of the child policy map configured in step 1.

Configuring an LLQ Policy Map

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **policy-map** *policy-map-name*

Example:

```
Device(config)# policy-map llq-flat
```

Creates a policy and enters QoS policy-map configuration mode.

Step 4 **class** *class-map-name*

Example:

Assigns the traffic class you specify to the policy map and enters policy-map class configuration mode.

Step 5 **priority**

Example:

```
Device(config-pmap-c)# priority
```

Configures LLQ, providing strict priority queueing (PQ) for class-based weighted fair queueing (CBWFQ).

Step 6 **exit**

Example:

```
Device(config-pmap-c)# exit
```

Exits QoS policy-map class configuration mode.

Step 7 **class** *class-map-name*

Example:

Assigns the traffic class you specify to the policy map and enters QoS policy-map class configuration mode.

Step 8 **shape average** *value*

Example:

```
Device(config-pmap-c)# shape average 200000000
```

Configures a shape entity with a Comitted Information Rate of 200 Mb/s.

Step 9 **exit**

Example:

```
Device(config-pmap-c)# exit
```

Exits QoS policy-map class configuration mode.

Step 10 **class** *class-map-name*

Example:

Assigns the traffic class you specify to the policy map and enters QoS policy-map class configuration mode.

Step 11 **bandwidth** *percent*

Example:

```
Device(config-pmap-c)# bandwidth 4000000
```

(Optional) Specifies a bandwidth percent for class-level queues to be used during congestion to determine the amount of excess bandwidth (unused by priority traffic) to allocate to non-priority queues.

Step 12 **exit**

Example:

```
Device(config-pmap-c)# exit
```

Exits QoS policy-map class configuration mode.

Configuring Port Level Shaping on the Main Interface with Ethernet Flow Points

To configure port level shaping on the main interface with EFPS, first you enable the autonegotiation protocol on the interface, then you attach a policy map to the interface and finally you configure the Ethernet service instance.

Step 1 **enable**

Example:

```
Device> enable
```

Enables privileged EXEC mode.

- Enter your password if prompted.

Step 2 **configure terminal**

Example:

```
Device# configure terminal
```

Enters global configuration mode.

Step 3 **interface** *type number*

Example:

```
Device(config)# interface GigabitEthernet 0/0/1
```

Configures an interface type and enters interface configuration mode.

- Enter the interface type number.

Step 4 **no ip address**

Example:

```
Device(config-if)# no ip address
```

Disables IP routing on the interface.

Step 5 **negotiation auto**

Example:

```
Device(config-if)# negotiation auto
```

Enables the autonegotiation protocol to configure the speed, duplex, and automatic flow control of the Gigabit Ethernet interface.

Step 6 **service-policy output** *policy-map-name*

Example:

```
Device(config-if)# service-policy output parent-llq
```

Specifies the name of the policy map to be attached to the input or output direction of the interface.

- You can enter the name of a hierarchical or a flat policy map.

Step 7 **service instance** *id* **ethernet**

Example:

```
Device(config-if)# service instance 1 ethernet
```

Configures an Ethernet service instance on an interface and enters service instance configuration mode.

Step 8 **encapsulation dot1q** *vlan-id*

Example:

```
Device(config-if-srv)# encapsulation dot1q 100
```

Defines the matching criteria to map 802.1Q frames' ingress on an interface to the service instance.

Step 9 **bridge-domain** *bridge-domain-id*

Example:

```
Device(config-if-srv)# bridge-domain 100
```

Binds the bridge domain to the service instance.

Step 10 **exit**

Example:

```
Device(config-if-serv)# exit
```

Exits service instance configuration mode.

Step 11 **service instance** *id* **ethernet**

Example:

```
Device(config-if)# service instance 2 ethernet
```

Configures an Ethernet service instance on an interface and enters service instance configuration mode.

Step 12 **encapsulation dot1q** *vlan-id*

Example:

```
Device(config-if-srv)# encapsulation dot1q 101
```

Defines the matching criteria to map 802.1Q frames' ingress on an interface to the service instance.

Step 13 **bridge-domain** *bridge-domain-id*

Example:

```
Device(config-if-srv)# bridge-domain 101
```

Binds the bridge domain to the service instance.

Step 14 **exit**

Example:

```
Device(config-if-srv)# exit
```

Exits QoS policy-map class configuration mode.

Step 15 **end**

Example:

```
Device(config-if)# end
```

(Optional) Exits interface configuration mode.

Configuration Examples for Port-Shaper and LLQ in the Presence of EFPs

Example: Configuring Hierarchical QoS Port Level Shaping on the Main Interface with EFPs

The following example shows how to configure hierarchical QoS port level shaping on a main physical interface to support traffic prioritization and Low Level Queueing across all EFPs configured on the interface:

```
policy-map parent-llq
 class class-default
   service-policy child-llq

policy-map child-llq
 class precedenc-1
  set cos 5
  bandwidth percent 20
 class precedenc-2
  bandwidth percent 80

interface GigabitEthernet 0/0/1
 no ip address
 negotiation auto
 service-policy output parent-llq
```

```
 service instance 1 ethernet
  encapsulation dot1q 100
  bridge-domain 100
 !
 service instance 2 ethernet
  encapsulation dot1q 101
  bridge-domain 101
```

Note Only match EFP and match qos-group is supported on RSP3 in egress policy map.

Example: Configuring Port Level Shaping on the Main Interface with EFPs

The following example shows how to configure port level shaping on a main physical interface to support traffic prioritization and Low Level Queueing across all Ethernet Flow Points (EFPs) configured on the interface:

```
policy-map llq_flat
 class dscp-af1
  priority
  class dscp-af2
   shape average 200000000
  class dscp-af3
   bandwidth 400000

interface GigabitEthernet 0/0/1
 no ip address
 negotiation auto
 service-policy output llq_flat
 service instance 1 ethernet
  encapsulation dot1q 100
  bridge-domain 100
 !
 service instance 2 ethernet
  encapsulation dot1q 101
  bridge-domain 101
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Commands List, All Releases*
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS QoS Command Reference*
Policing and shaping	"Policing and Shaping Overview" module

Related Topic	Document Title
Class maps	"Applying QoS Features Using the MQC" module
Policy maps	"Applying QoS Features Using the MQC" module
Low Latency Queueing	*QoS Congestion Management Configuration Guide*

Standards and RFCs

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

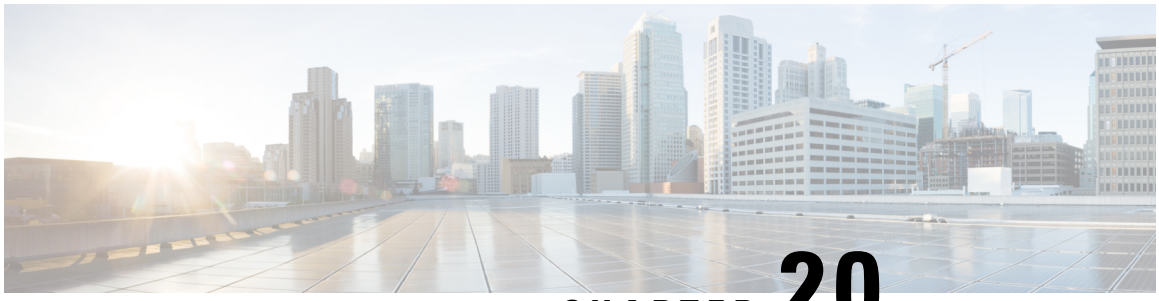
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Port-Shaper and LLQ in the Presence of EFPs

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 22: Feature Information for Port-Shaper and LLQ in the Presence of EFPs



CHAPTER 20

Two-Rate Policer

This module describes the Two-Rate Policer feature and explains how to configure it.

Finding Support Information for Cisco IOS XE Software Images

Use Cisco Feature Navigator to find information about platform support and Cisco IOS XE Software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

Feature Overview

When configured, an ATM switch at the network side of a user-to-network (UNI) interface polices the flow of cells in the forward (into the network) direction of a virtual connection. These traffic policing mechanisms are known as usage parameter control (UPC). With UPC, the switch determines whether received cells comply with the negotiated traffic management values and takes one of the following actions on violating cells:

- Pass the cell without changing the cell loss priority (CLP) bit in the cell header.
- Tag the cell with a CLP bit value of 1.
- Drop (discard) the cell.

The SVC/SoftPVC feature enables you to specify which traffic to police, based on service category, on switched virtual circuits (SVCs) or terminating VCs on the destination end of a soft VC.

Benefits

Bandwidth Management Through Rate Limiting

Traffic policing allows you to control the maximum rate of traffic transmitted or received on an interface. Traffic policing is often configured on interfaces at the edge of a network to limit traffic into or out of the network. In most Traffic Policing configurations, traffic that falls within the rate parameters is transmitted, whereas traffic that exceeds the parameters is dropped or transmitted with a different priority.

Packet Marking

Packet marking allows you to partition your network into multiple priority levels or classes of service (CoS). A packet is marked and these markings can be used to identify and classify traffic for downstream devices. In some cases, such as ATM Cell Loss Priority (CLP) marking or Frame Relay Discard Eligibility (DE) marking, the marking is used to classify traffic.

- Use traffic policing to set the IP precedence or DSCP values for packets entering the network. Networking devices within your network can then use the adjusted IP precedence values to determine how the traffic should be treated. For example, the Weighted Random Early Detection (WRED) feature uses the IP precedence values to determine the probability that a packet will be dropped.
- Use traffic policing to assign packets to a QoS group. The router uses the QoS group to determine how to prioritize packets within the router.

Traffic can be marked without using the Traffic Policing feature. If you want to mark traffic but do not want to use Traffic Policing, see the "Marking Network Traffic" module.

Packet Prioritization for Frame Relay Frames

The Traffic Policing feature allows users to mark the Frame Relay DE bit of the Frame Relay frame. The Frame Relay DE bit is one bit and, therefore, can be set to either 0 or 1. In congested environments, frames with the DE bit set to 1 are discarded before frames with the DE bit set to 0.

Packet Prioritization for ATM Cells

The Traffic Policing feature allows users to mark the ATM CLP bit in ATM cells. The ATM CLP bit is used to prioritize packets in ATM networks. The ATM CLP bit is one bit and, therefore, can be set to either 0 or 1. In congested environments, cells with the ATM CLP bit set to 1 are discarded before cells with the ATM CLP bit set to 0.

Restrictions for Two-Rate Policing

The following restrictions apply to the Two-Rate Policer:

- Two-rate policing can be configured on an interface, a subinterface, a Frame Relay data-link connection identifier (DLCI), and an ATM permanent virtual circuit (PVC).
- Two-rate policing is not supported on EtherChannel or tunnel interfaces.

Prerequisites for Two-Rate Traffic Policing

To configure the Two-Rate Policer, a traffic class and a service policy must be created, and the service policy must be attached to a specified interface.

Configuration Tasks

See the following sections for configuration tasks for the Two-Rate Policer feature.

Configuring the Two-Rate Policer

Command	Purpose
`Router(config-pmap-c)# police cir cir [bc conform-burst ] pir pir [be peak-burst ] [conform-action action [exceed-action action [violate-action action]]]`	Specifies that both the CIR and the PIR are to be used for two-rate traffic policing, and specifies multiple actions applied to packets marked as conforming to, exceeding, or violating a specific rate. Use one line per action that you want to specify. Enters policy-map class police configuration mode. The **bc** and **be** keywords and their associated arguments (*conform-burst* and *peak-burst* , respectively) are optional.

Although not required for configuring the Two-Rate Policer, the command syntax of the **police** command also allows you to specify the action to be taken on a packet when you enable an optional *action* argument. The resulting action corresponding to the keyword choices are listed in Table 1 .

Table 23: police Command Action Keywords

Keyword	Resulting Action
drop	Drops the packet.
set-clp-transmit	Sets the ATM CLP bit from 0 to 1 on the ATM cell and sends the packet with the ATM CLP bit set to 1.
set-dscp-transmit *new-dscp*	Sets the IP DSCP value and sends the packet with the new IP DSCP value setting.
set-frde-transmit	Sets the Frame Relay DE bit from 0 to 1 on the Frame Relay frame and sends the packet with the DE bit set to 1.
set-mpls-exp-transmit	Sets the MPLS experimental bits from 0 to 7 and sends the packet with the new MPLS experimental bit value setting.

Keyword	Resulting Action
set-prec-transmit *new-prec*	Sets the IP precedence and sends the packet with the new IP precedence value setting.
set-qos-transmit *new-qos*	Sets the QoS group value and sends the packet with the new QoS group value setting.
transmit	Sends the packet with no alteration.

Verifying the Two-Rate Policer Configuration

Command	Purpose
Router# **show policy-map interface**	Displays statistics and configurations of all input and output policies attached to an interface.

Troubleshooting Tips

Monitoring and Maintaining the Two-Rate Policer

Command	Purpose
Router# **show policy-map**	Displays all configured policy maps.
Router# **show policy-map** *policy-map-name*	Displays the user-specified policy map.
Router# **show policy-map interface**	Displays statistics and configurations of all input and output policies that are attached to an interface.

Configuration Examples

Example Limiting the Traffic Using a Policer Class

In this example, the Two-Rate Policer is configured on a class to limit traffic to an average committed rate of 500 kbps and a peak rate of 1 Mbps.

```
Router(config)# class-map police
Router(config-cmap)# match access-group 101
Router(config-cmap)# policy-map policy1
```

```
Router(config-pmap)# class police
Router(config-pmap-c)# police cir 500000 bc 10000 pir 1000000 be 10000 conform-action
 transmit exceed-action set-prec-transmit 2 violate-action drop
Router(config)# interface serial3/0/0
Router(config-if)# service-policy output policy1
Router(config-if)# end
Router# show policy-map policy1
 Policy Map policy1
  Class police
   police cir 500000 conform-burst 10000 pir 1000000 peak-burst 10000 conform-action transmit
 exceed-action set-prec-transmit 2 violate-action drop
```

Traffic marked as conforming to the average committed rate (500 kbps) will be sent as is. Traffic marked as exceeding 500 kbps, but not exceeding 1 Mbps, will be marked with IP Precedence 2 and then sent. All traffic exceeding 1 Mbps will be dropped. The burst parameters are set to 10,000 bytes.

In the following example, 1.25 Mbps of traffic is sent ("offered") to a *policer* class.

```
Router# show policy-map interface serial3/0/0
 Serial3/0/0
  Service-policy output: policy1
   Class-map: police (match all)
    148803 packets, 36605538 bytes
    30 second offered rate 1249000 bps, drop rate 249000 bps
    Match: access-group 101
    police:
     cir 500000 bps, conform-burst 10000, pir 1000000, peak-burst 100000
     conformed 59538 packets, 14646348 bytes; action: transmit
     exceeded 59538 packets, 14646348 bytes; action: set-prec-transmit 2
     violated 29731 packets, 7313826 bytes; action: drop
     conformed 499000 bps, exceed 500000 bps violate 249000 bps
   Class-map: class-default (match-any)
    19 packets, 1990 bytes
    30 seconds offered rate 0 bps, drop rate 0 bps
    Match: any
```

The Two-Rate Policer marks 500 kbps of traffic as conforming, 500 kbps of traffic as exceeding, and 250 kbps of traffic as violating the specified rate. Packets marked as conforming will be sent as is, and packets marked as exceeding will be marked with IP Precedence 2 and then sent. Packets marked as violating the specified rate are dropped.

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Token bucket mechanisms	"Policing and Shaping Overview" module
MQC	"Applying QoS Features Using the MQC" module

Related Topic	Document Title
QoS features such traffic marking, and traffic policing	• "Marking Network Traffic" module • "Traffic Policing" module

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
• CISCO-CLASS-BASED-QOS-MIB • CISCO-CLASS-BASED-QOS-CAPABILITY-MIB	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2698	*A Two Rate Three Color Marker*

Technical Assistance

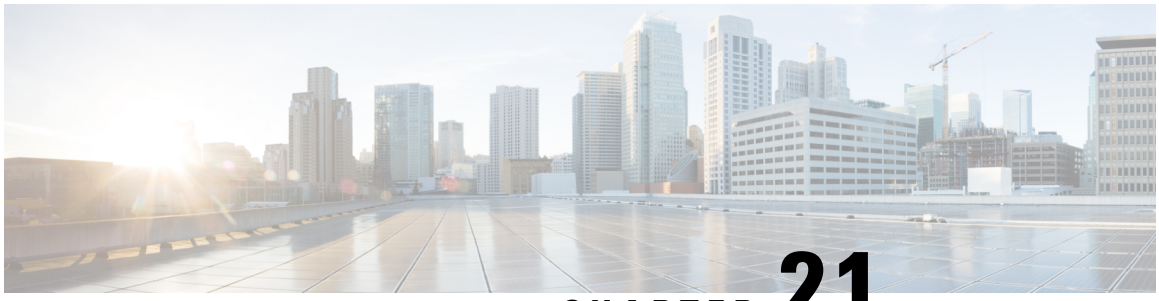
Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Two-Rate Policer

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 24: Feature Information for Two-Rate Policer



CHAPTER 21

Punt Policing and Monitoring

Punt policing protects the Route Processor (RP) from having to process noncritical traffic, which increases the CPU bandwidth available to critical traffic. Traffic is placed into different CPU queues based on various criteria. The Punt Policing and Monitoring feature allows you to police the punt rate on a per-queue basis.

Feature Information for Punt Policing and Monitoring

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 25: Feature Information for Punt Policing and Monitoring

Information About Punt Policing and Monitoring

Overview of Punt Policing and Monitoring

Packets received on an interface are punted to the Router Processor (RP) for various reasons. Some examples of these various reasons include, unicast and multicast control plane traffic that are destined for a routing protocol process running on the RP, and IP packets that generate Internet Control Message Protocol (ICMP) exceptions such as a Time to live (TTL) expiration. The RP has a limited capacity to process the punted packets, and while some of them are critical for the router operation and should not be dropped, some can be dropped without impacting the router operation.

Punt policing frees the RP from having to process noncritical traffic. Traffic is placed in queues based on various criteria, and you can configure the maximum punt rate for each queue which allows you to configure the system so that packets are less likely to be dropped from queues that contain critical traffic.

Note Traffic on certain CPU queues could still be dropped, regardless of the configured punt rate, based on other criteria such as the queue priority, queue size, and traffic punt rate.

Per-Interface Per-Cause Punt Policer

Per-interface per-cause (PIPC) punt policing is an enhancement to the Punt Policing and Monitoring feature that allows you to control and limit traffic per interface. From Cisco IOS XE Release 17.5.1, you can set the PIPC rate for all the control plane-punted traffic. When you set the PIPC rate, any traffic beyond the set limit is dropped, thereby enabling you to control the traffic during conditions such as L2 storming.

The PIPC punt policer configuration is supported for the following interfaces:

- Main interface
- Subinterface
- Port channel
- Port channel subinterface
- Tunnels
- PPPoE interface

Restrictions for Per-Interface Per-Cause Punt Policer

- PIPC punt policing is not supported for L2 Ethernet Flow Points (EFPs).
- This configuration supports only two policies per interface.

How to Configure Punt Policing and Monitoring

Configuring Punt Policing

Note Traffic on a specific CPU queue may be dropped irrespective of the configured maximum punt rate, based on the queue priority, queue size, and the configured traffic punt rate.

Perform this task to specify the maximum punt rate on the specified queue.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos-policer queue** *queue-id cir bc*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** Device# configure terminal	Enters the global configuration mode.
Step 3	**platform qos-policer queue** *queue-id cir bc* **Example:** Device(config)# platform qos-policer queue 20 384000 8000	Enables punt policing on a queue, and specifies the maximum punt rate on a per-queue basis. *cir*— Indicates Committed Information Rate (CIR). The range is 384000-20000000 bps. *bc*— Indicates Committed Burts (BC). The range is 8000-16000000 bps.
Step 4	**end** **Example:** Device(config)# end	(Optional) Returns to privileged EXEC mode.

Configuring Punt Policing on an Interface

Note At an interface level, punt control can be enabled or disabled by the **no punt-control enable** command. You can configure the rate, however, by default, it uses the global configuration if the rate is not configured.

Perform this task to enable or disable punt control on an interface:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform punt-interface rate***rate*
4. **punt-control enable** *rate*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** Device> enable	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** Device# configure terminal	Enters global configuration mode.
Step 3	**platform punt-interface rate***rate* **Example:** Device(config)# platform punt-interface rate 10	Sets the global punt-interface policer rate.
Step 4	**punt-control enable** *rate* **Example:** Device(config)# interface Port-channel 1.2 Device(config-if)# punt-control enable	Punt control is enabled at an interface level.
Step 5	**end** **Example:**	(Optional) Returns to privileged EXEC mode.

Configuring Punt Policing Per Interface Per Cause

SUMMARY STEPS

1. enable
2. configure terminal
3. **punt-control cause <cause> <rate>**
4. end

DETAILED STEPS

	Command or Action	Purpose
Step 1	enable **Example:** `Device> enable`	This enables the privileged EXEC mode. Enter the password, if prompted.
Step 2	configure terminal **Example:** `Device# configure terminal`	This enables the Global Configuration mode.

	Command or Action	Purpose
Step 3	**punt-control cause <cause> <rate>** **Example:** `Device(config-if)# punt-control cause arp 80`	This sets the PIPC rate for the interface that you specify, for example, **punt-control cause arp 80**.
Step 4	end	This exits the current configuration.

Example

```
Device> enable
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# interface GigabitEthernet 1
Device(config-if)# punt-control cause arp 80
```

Configuring the Default PIPC Rate for an Interface

Procedure

	Command or Action	Purpose

Step 1	To set a default global PIPC rate for at an interface level, enter the platform punt-intf per-cause rate <rate> command.
	Example:
	`platform punt-intf per-cause rate 100  // Global PIPC rate`
	`interface Port-channel1.100`
	`punt-control cause bfd-control // PIPC rate 100`
	`punt-control cause arp 200 // PIPC rate 200`
		Sets the default global PIPC rate for the interface that you specify. Here, the default rate for BFD is set to 100, while the default rate for ARP is 200. If there's an inflow beyond the specified rate for this interface, the traffic is dropped.

Verifying Punt Policing

Verifying Queue-Based Punt Policing

Use the **show platform software infrastructure punt statistics** to display punt police statistics:

```
Router# show platform software infrastructure punt statistics
UEA Punt Statistics

Global drops : 0

 Queue Name             | Rx count                   |  Drop count
------------------------+----------------------------+-----------------------
 SW FORWARDING Q        | 0                          | 0
 ROUTING PROTOCOL Q     | 0                          | 0
 ICMP Q                 | 0                          | 0
 HOST Q                 | 57115                      | 0
 ACL LOGGING Q          | 0                          | 0
```

```
STP Q                  | 0                       | 0
L2 PROTOCOL Q          | 6571                    | 0
MCAST CONTROL Q        | 208839                  | 0
BROADCAST Q            | 4                       | 0
REP Q                  | 0                       | 0
CFM Q                  | 0                       | 0
CONTROL Q              | 0                       | 0
IP MPLS TTL Q          | 0                       | 0
DEFAULT MCAST Q        | 0                       | 0
MCAST ROUTE DATA Q     | 0                       | 0
MCAST MISMATCH Q       | 0                       | 0
RPF FAIL Q             | 0                       | 0
ROUTING THROTTLE Q     | 87                      | 0
MCAST Q                | 0                       | 0
MPLS OAM Q             | 0                       | 0
IP MPLS MTU Q          | 0                       | 0
PTP Q                  | 0                       | 0
LINUX ND Q             | 0                       | 0
KEEPALIVE Q            | 0                       | 0
ESMC Q                 | 0                       | 0
FPGA BFD Q             | 0                       | 0
FPGA CCM Q             | 0                       | 0
FPGA CFE Q             | 0                       | 0
L2PT DUP Q             | 0                       | 0
```

Verifying Punt Policing Statistics

Use the **show platform hardware pp active infrastructure pi npd rx policer** command to display the punt policing statistics for all queues.

```
Ring |          Queue Name |           Punt rate |          Burst rate
-----+---------------------+---------------------+--------------------
 0   |     SW FORWARDING Q |                 500 |                1000
 1   |  ROUTING PROTOCOL Q |                 500 |                1000
 2   |              ICMP Q |                 500 |                1000
 3   |              HOST Q |                1000 |                2000
 4   |       ACL LOGGING Q |                 500 |                1000
 5   |               STP Q |                3000 |                6000
 6   |       L2 PROTOCOL Q |                1000 |                2000
 7   |     MCAST CONTROL Q |                1000 |                2000
 8   |         BROADCAST Q |                1000 |                2000
 9   |               REP Q |                3000 |                6000
10   |           BGP LDP Q |                3000 |                6000
11   |           CONTROL Q |                1000 |                2000
12   |       IP MPLS TTL Q |                1000 |                2000
13   |     DEFAULT MCAST Q |                 500 |                1000
14   |  MCAST ROUTE DATA Q |                 500 |                1000
15   |    MCAST HIGH PRI Q |                1000 |                2000
16   |          RPF FAIL Q |                 500 |                1000
17   |  ROUTING THROTTLE Q |                 500 |                1000
18   |             MCAST Q |                 500 |                1000
19   |          MPLS OAM Q |                1000 |                2000
20   |       IP MPLS MTU Q |                 500 |                1000
21   |               PTP Q |                3000 |                6000
22   |          LINUX ND Q |                 500 |                1000
23   |         KEEPALIVE Q |                1000 |                2000
24   |              ESMC Q |                3000 |                6000
25   |          FPGA BFD Q |                4000 |                8000
26   |          FPGA CCM Q |                4000 |                8000
27   |          FPGA CFE Q |                1000 |                2000
28   |          L2PT DUP Q |                4000 |                8000
29   |          TDM CTRL Q |                3000 |                6000
```

```
30   |  ICMP UNREACHABLE Q |                 500 |                1000
31   |              SSFPD Q |                6000 |               12000
```

Use the **show platform software infrastructure punt statistics** command to view the statistics on the RSP3 module.

```
Router#

Global drops : 0

 Queue Name              | Rx count                    |  Drop count
-------------------------+-----------------------------+-----------------------
 SW FORWARDING Q         | 0                           | 0
 ROUTING PROTOCOL Q      | 0                           | 0
 ICMP Q                  | 0                           | 0
 HOST Q                  | 0                           | 0
 ACL LOGGING Q           | 0                           | 0
 STP Q                   | 0                           | 0
 L2 PROTOCOL Q           | 0                           | 0
 MCAST CONTROL Q         | 0                           | 0
 BROADCAST Q             | 0                           | 0
 REP Q                   | 0                           | 0
 BGP LDP Q               | 0                           | 0
 CONTROL Q               | 0                           | 0
 IP MPLS TTL Q           | 0                           | 0
 DEFAULT MCAST Q         | 0                           | 0
 MCAST ROUTE DATA Q      | 0                           | 0
 MCAST MISMATCH Q        | 0                           | 0
 RPF FAIL Q              | 0                           | 0
 ROUTING THROTTLE Q      | 0                           | 0

MCAST Q                 | 0                           | 0
 MPLS OAM Q              | 0                           | 0
 IP MPLS MTU Q           | 0                           | 0
 PTP Q                   | 0                           | 0
 LINUX ND Q              | 0                           | 0
 KEEPALIVE Q             | 0                           | 0
 ESMC Q                  | 0                           | 0
 FPGA BFD Q              | 0                           | 0
 FPGA CCM Q              | 0                           | 0
 FPGA CFE Q              | 0                           | 0
 L2PT DUP Q              | 0                           | 0
 TDM CTRL Q              | 0                           | 0
 ICMP UNREACHABLE Q      | 0                           | 0
 SSFP Q                  | 0                           | 0
 MIRROT Q                | 0                           | 0
```

Use the **show platform hardware pp active feature qos policer cpu all 1** command to clear the statistics of all the CPU queues.

Use the **show platform hardware pp active feature qos policer cpu all 0** command to clear the statistics of a particular CPU queue.

```
########## Stats for CPU queue 0 ##########
Internal Qnum: 1         Queue Name: SW FORWARDING Q
Policer conform: 0 (packets) 0 (bytes)
Policer exceed: 0 (packets) 0 (bytes)
RM Drops: 0 (packets) 0 (bytes)
Policer commit rate is: 1000000, Policer burst commit is 100000

########### Stats for CPU queue 1 ##########
Internal Qnum: 2         Queue Name: ROUTING PROTOCOL Q
Policer conform: 0 (packets) 0 (bytes)
Policer exceed: 0 (packets) 0 (bytes)
RM Drops: 0 (packets) 0 (bytes)
```

```
Policer commit rate is: 1000000, Policer burst commit is 100000
_______
_______
_______
########### Stats for CPU queue 30 ##########
Internal Qnum: 31        Queue Name: ICMP UNREACHABLE Q
Policer conform: 0 (packets) 0 (bytes)
Policer exceed: 0 (packets) 0 (bytes)
RM Drops: 0 (packets) 0 (bytes)
Policer commit rate is: 1000000, Policer burst commit is 100000

########### Stats for CPU queue 31 ##########
Internal Qnum: 32        Queue Name: SSFPD Q
Policer conform: 0 (packets) 0 (bytes)
Policer exceed: 0 (packets) 0 (bytes)
RM Drops: 0 (packets) 0 (bytes)
Policer commit rate is: 1000000, Policer burst commit is 100000
```

Use **show platform hardware pp active feature qos policer cpu 3 0** to display the queue specific statistics.

```
########### Stats for CPU queue 3 ##########
Internal Qnum: 4         Queue Name: HOST Q
Policer conform: 0 (packets) 0 (bytes)
Policer exceed: 0 (packets) 0 (bytes)
RM Drops: 0 (packets) 0 (bytes)
Policer commit rate is: 12000000, Policer burst commit is 3000000

3 —— queueId of CPU   and 0 — show stats
```

Use the **show platform hardware pp active feature qos policer cpu all 0** to display the output after adding the drop cause. Following commands are applicable only for RSP3 module:

```
########### Stats for CPU queue 0 ##########
Internal Qnum: 8000CPU
Port num: 0
Policer conform: 0 (packets) 0 (bytes)
Policer exceed:  0 (packets) 0 (bytes)
Policer commit rate is: 500000 bps, Policer burst commit is 16000 bytes
########### Stats for CPU queue 1 ##########
Internal Qnum: 8008CPU
Port num: 0
Policer conform: 0 (packets) 0 (bytes)
Policer exceed:  0 (packets) 0 (bytes)
Policer commit rate is: 1000000 bps, Policer burst commit is 100000 bytes
########### Stats for CPU queue 2 ##########
Internal Qnum: 8016CPU
Port num: 0
Policer conform: 0 (packets) 0 (bytes)
Policer exceed:  0 (packets) 0 (bytes)
Policer commit rate is: 1000000 bps, Policer burst commit is 100000 bytes
```

Verifying Per-Interface Per-Cause Punt Policer

Step 1 To verify whether the global PIPC rate has been successfully attached, run the **show run | in platform** command:

Example:

```
Device# show run | in platform
platform punt-intf per-cause rate 100
```

Step 2 To verify whether the PIPC configuration is enabled, run the **show run int <interface>** command. In the following sample configuration, the punt-control cause in the output verifies that the PIPC rate of 80 for ARP is successfully applied:

Example:

```
Device# show run int GigabitEthernet 1
Building configuration...
Current configuration : 100 bytes
!
interface GigabitEthernet1
punt-control cause arp 80
!
End
```

Example

What to do next

-

Configuration Examples for Punt Policing and Monitoring

Example: Configuring Punt Policing

The following example shows how to enable punt-policing:

```
Router# enable
Router# configure terminal
Router(config)# platform qos-policer queue 3 384000 8000
```

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Traffic marking	“Marking Network Traffic” module
Traffic policing	“Traffic Policing” module
Traffic policing and shaping concepts and overview information	“Policing and Shaping Overview” module

Related Topic	Document Title
Modular quality of service command-line interface (MQC)	"Applying QoS Features Using the MQC" module

Standards

Standard	Title
None	—

MIBs

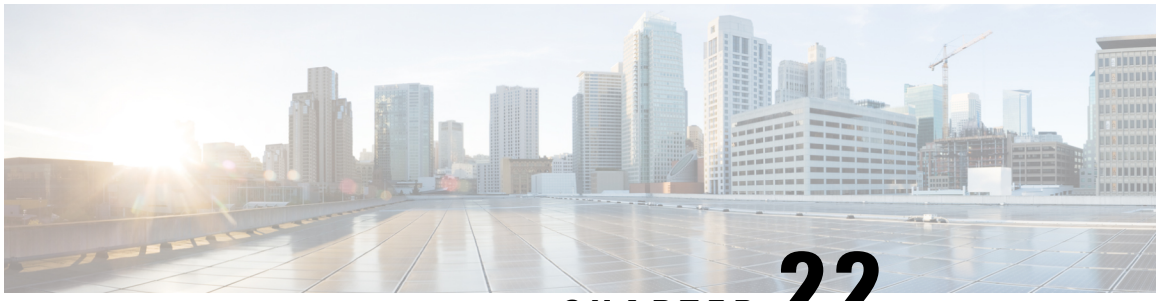
MIB	MIBs Link
None	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
None	

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 22

Adaptive QoS over DMVPN

Adaptive QoS over Dynamic Multipoint VPN (DMVPN) ensures effective bandwidth management using dynamic shapers based on available bandwidth. This feature enables various QoS features to adapt to non service-level agreement (SLA) based environments where bandwidth is variable and fluctuate with time.

Prerequisites for Adaptive QoS over DMVPN

Adaptive QoS over DMVPN can be enabled either on hub or spoke or both. To enable feature at a spoke side, the spoke must support basic egress per-SA QoS policy.

Internet Protocol Security (IPSec) is required and must be configured before Adaptive QoS is enabled on the DMVPN tunnel.

Restrictions for Adaptive QoS over DMVPN

The Adaptive QoS over DMVPN feature configuration is:

- Supported only on DMVPN tunnels
- Allowed only on egress direction
- Allowed only in parent most policy that has class-default only
- Not supported on Point-to-Point tunnels
- Adaptive QOS is not supported on Cisco IWAN 2.1

Information About Adaptive QoS over DMVPN

Overview of Adaptive QoS over DMVPN

Enterprise networks are increasingly using the Internet as form of WAN transport, therefore QoS models needs to be revisited. QoS works effectively when deployed in an service-level agreement (SLA) environment today, like Multiprotocol Label Switching (MPLS) . The available bandwidth on the internet at a given point of time can vary, and can be often much lesser than the actual bandwidth offered by the service provider. In cases of non SLA environments, QoS has limitations - mainly because it cannot predict changing bandwidth on the link.

Cisco Intelligent WAN (IWAN) recommends using Dynamic Multipoint VPN (DMVPN) over Internet to connect branches to the data center or headquarters, and QoS to be deployed in such environments of fluctuating bandwidth. Currently, the shapers that are applied as part of the egress QoS policy are static in value - they are configured based on the service provider bandwidth offering, they do not change with time and hence do not reflect the actual available Internet bandwidth. In many instances where Internet available bandwidth becomes much lesser than the offered bandwidth, the shapers become irrelevant as they do not adapt to the varying bandwidth. Due to the static value of the shapers, application traffic gets dropped indiscriminately at the Internet core, nullifying the very need to have configured a QoS policy to protect critical traffic.

DMVPN provides the ability to do QoS per-tunnel, which means a QoS policy can be applied at the hub towards a specific spoke, to ensure a high bandwidth hub does not overrun a low capacity spoke. However, these QoS policies still work with static shapers per spoke. If the bandwidth towards a particular spoke fluctuates, the shapers towards the spokes do not adapt. Also, it is not possible today to configure a QoS policy for the traffic from the spoke towards the hub, which is very common in many retail-like environments.

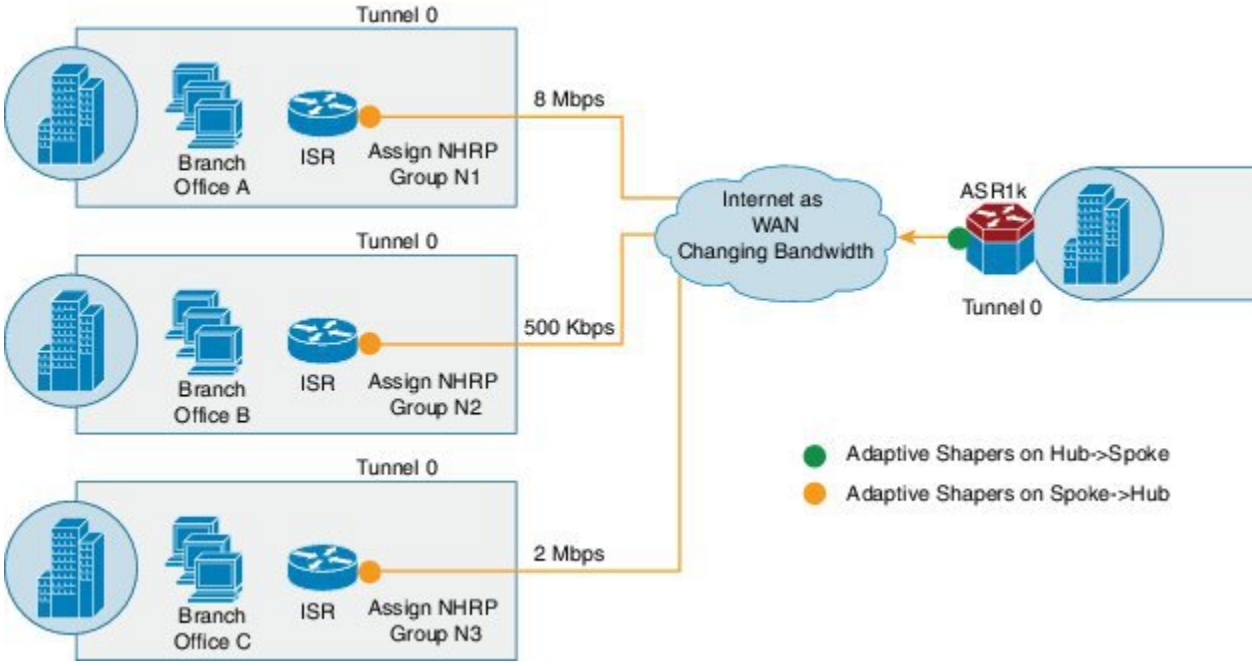
The Adaptive QoS over DMVPN feature provides the following benefits:

- Adjusts the shaper parameters based on the actual available Internet bandwidth in both directions that is periodically computed.
- Allows to configure a QoS policy on the spoke towards the hub.
- Ensures better control of application performance at the enterprise edge even in changing bandwidth scenarios over the Internet.
- Allows aggregate tunnel shape adaptation to provide effective bandwidth between spoke and hub.

Adaptive QoS for Per-Tunnel QoS over DMVPN

Per-tunnel QoS over DMVPN can be configured on the hub towards the spoke today using Next Hop Resolution Protocol (NHRP) groups. The QoS policies contain static shapers. With Adaptive QoS, the framework of per tunnel QoS configuration remains the same, but the shaper can be an adaptive one as shown in the following figure. These shapers would adapt automatically based on the changing Internet bandwidth that is periodically computed using an algorithm.

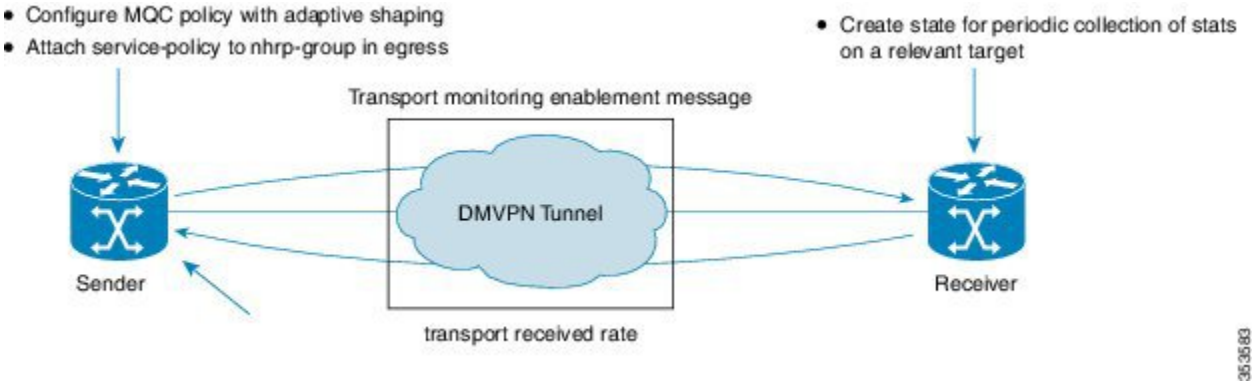
Figure 6: Adaptive QoS for Per-Tunnel QoS over DMVPN



Workflow of Adaptive QoS

The Adaptive QoS over DMVPN feature adapts shaping rate at the Sender based on the available bandwidth between specific Sender and Receiver (two end-points of a DMVPN tunnel).

Figure 7: Workflow of Adaptive QoS



At the Sender:

- Configure MQC Policy with Adaptive shaping
- Attach service-policy to nhrp-group in Egress

At the Receiver:

Create state for periodic collection of stats on a relevant target

How to Configure Adaptive QoS over DMVPN

Note Configure the Per-Tunnel QoS for DMVPN before configuring the Adaptive QoS over DMVPN feature, as Adaptive QoS over DMVPN feature is an enhancement to the Per-Tunnel QoS for DMVPN feature.

Note For details on configuring the Per-Tunnel QoS for DMVPN feature, refer to Per-Tunnel QoS for DMVPN .

Configuring Adaptive QoS for DMVPN

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *parent-policy-name*
4. **class class-default**
5. **shape adaptive** { **upper-bound** *bps* |**percent** *percentage* }[**lower-bound** *bps*| **percent** *percentage*]
6. **end**
7. **configure terminal**
8. **interface tunnel** *tunnel-id*
9. **nhrp map group** *group-name* **service-policy output** *parent-policy-name*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *parent-policy-name* **Example:** `Router(config)# policy-map example`	Creates or modifies a child policy map and enters policy-map configuration mode. • Enter the name of the child policy map.

	Command or Action	Purpose
Step 4	**class class-default** **Example:** `Router(config-pmap)# class class-default`	This step associates the traffic class with the traffic policy. Configures the default class map and enters policy-map class configuration mode.
Step 5	**shape adaptive** { **upper-bound** *bps* |**percent** *percentage* }[**lower-bound** *bps*| **percent** *percentage*] **Example:** `Router(config-pmap-c)# shape adaptive upper-bound 20000`	Creates a specific adaptive shaper that has upper bound on the rate and optionally lower bound on the rate. **Note** When such a template is attached to a target, adaptive shaping is enabled for that instance. Shaping rate adapts to a new rate, that is a function of parameters, including peer's received rate.
Step 6	**end** **Example:** `Router(config-pmap-c)# end`	Returns to privileged EXEC mode.
Step 7	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 8	**interface tunnel** *tunnel-id* **Example:** `Router(config)# interface tunnel 0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and interface number.
Step 9	**nhrp map group** *group-name* **service-policy output** *parent-policy-name* **Example:** `Router(config-if)# nhrp map group 1 service-policy output example`	Adds the NHRP group to the QoS policy map on the hub.
Step 10	**end** **Example:** `Router(config-if)# end`	Returns to privileged EXEC mode.

Verifying the Adaptive QoS over DMVPN

SUMMARY STEPS

1. **enable**
2. **show dmvpn**

3. **show policy-map** [*policy-map-name*]
4. **show policy-map multipoint**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.
Step 2	**show dmvpn** **Example:** `Router# show dmvpn`	Displays detailed DMVPN information for each session, including the Next Hop Server (NHS) and NHS status, crypto session information, and socket details. Also displays the NHRP group received from the spoke and the QoS policy applied to the spoke tunnel.
Step 3	**show policy-map** [*policy-map-name*] **Example:** `Router# show policy-map example`	Displays the configuration of all classes for a specified policy map or of all classes for all existing policy maps.
Step 4	**show policy-map multipoint** **Example:** `Router# show policy-map tunnel 0`	(Optional) Displays the statistics and the configurations of the input and output policies that are attached to an interface.
Step 5	**exit** **Example:** `Router(config-if)# exit`	(Optional) Returns to user EXEC mode.

Troubleshooting the Adaptive QoS over DMVPN

SUMMARY STEPS

1. **enable**
2. **debug qos peer mon detail**
3. **debug qos peer rate detail**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables higher privilege levels, such as privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**debug qos peer mon detail** **Example:** `Router# debug qos peer mon detail`	Displays debug messages for Adaptive QoS over DMVPN.
Step 3	**debug qos peer rate detail** **Example:** `Router# debug qos peer rate detail`	Displays debug messages for Adaptive QoS over DMVPN.

Configuration Examples for Configuring Adaptive QoS over DMVPN

Example Configuring Adaptive QoS over DMVPN

The following example shows how to configure Adaptive QoS over DMVPN:

```
Router(config)# policy-map example
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape adaptive upper-bound 20000
Router(config-pmap-c)# end
Router# configure terminal
Router(config)# interface tunnel 0
Router(config-if)# nhrp map group 1 service-policy output example
Router(config-if)# end
```

Example Verifying Adaptive QoS over DMVPN

The **show policy-map** and **show policy-map interface** commands can be used to confirm that the Adaptive QoS over DMVPN feature is enabled at an interface.

The following is a sample output of the **show dmvpn** command:

```
Router# show dmvpn


Interface: Tunnel1, IPv4 NHRP Details
Type: Hub, NHRP Peers:1,

# Ent  Peer NBMA Addr   Peer Tunnel Add   State UpDn Tm    Attrb
 ----- ------------- ----------------     ----- -------   -----
  1    10.1.1.1          10.10.1.2        UP    00:18:37   D


Interface: Tunnel2, IPv4 NHRP Details
Type: Hub, NHRP Peers:1,
```

```
# Ent Peer NBMA Addr   Peer Tunnel Add   State  UpDn Tm  Attrb
----- ---------------  --------------    ------  -------  -------

  1  10.2.1.1          10.10.2.2        UP     00:22:09  D


Interface: Tunnel3, IPv4 NHRP Details
Type: Hub, NHRP Peers:1,

# Ent Peer NBMA Addr   Peer Tunnel Add   State  UpDn Tm  Attrb
----- --------------   ---------------   ------ -------   ----
   1  10.3.1.1         10.10.3.2         UP     00:22:04   D


Interface: Tunnel4, IPv4 NHRP Details
Type: Hub, NHRP Peers:1,

# Ent Peer NBMA Addr   Peer Tunnel Add   State UpDn Tm  Attrb
----- --------------   ----------------  -----  ------  ----
  1    10.3.1.1        10.10.3.2         UP   00:22:01   D
```

The following is a sample output of the **show policy-map** command:

```
Router# show policy-map


Policy Map test
    Class class-default
      Adaptive Rate Traffic Shaping
      cir upper-bound 2120000 (bps) cir lower-bound 1120000 (bps)
```

The following is a sample output of the **show policy-map multipoint** command:

```
Router# show policy-map multipoint

 Service-policy output: test

  Class-map: class-default (match-any)
   0 packets, 0 bytes
   5 minute offered rate 0000 bps, drop rate 0000 bps
   Match: any
   Queueing
   queue limit 64 packets
   (queue depth/total drops/no-buffer drops)0/0/0
   (pkts output/bytes output) 0/0
   shape (adaptive) cir 2120000,bc 8480, be 8480
   lower bound cir 2120000
   target shape rate 2120000
```

Note One of the important parameters displayed as an output of the **show policy-map multipoint** command is **target shape rate**. The Adaptive QoS over DMVPN feature dynamically changes the value of the **target shape rate** to adapt to the available bandwidth.

Example for Troubleshooting Adaptive QoS over DMVPN

The **debug qos peer mon detail** and **debug qos peer rate detail** commands can be used to display any errors for the Adaptive QoS over DMVPN feature.

The following is a sample output of the **debug qos peer mon detail** command:

```
Router# debug qos peer mon detail

QoS peer remote monitoring debugging is on

Router#

*May 22 21:25:28.006 UTC: [SEND]Processing entry with address :
50.1.1.2,vrfid : 0 sending rate(delta bytes) : 1514
*May 22 21:25:28.006 UTC: [SEND]Processing entry with address :
50.1.1.3,vrfid : 0 sending rate(delta bytes) : 1598
*May 22 21:25:28.201 UTC: [RCV]Received message for interface Tunnel1
address 50.1.1.2 vrf 0
*May 22 21:25:28.201 UTC:
fdiff : 20517, sdiff : 19661, cur_dif : 3318, cum_diff : 20907

*May 22 21:25:28.201 UTC: qos_rate_status_update -- 392
*May 22 21:25:28.201 UTC: Last count : 128650
```

The following is a sample output of the **debug qos peer rate detail** command:

```
Router# debug qos peer rate detail


 *May 22 21:34:32.456 UTC: [RCV]Received message for interface Tunnel1
address 50.1.1.3 vrf 0
 *May 22 21:34:32.456 UTC: Enter qos_process_remote_rate_message:
 *May 22 21:34:32.456 UTC: Message for tun with o_ip : 50.1.1.3 tun t_ip
 : 13.1.1.1
 *May 22 21:34:32.456 UTC: [RCV]<DELTA>Message remote rate value is
116730f_cum_diff: 140155, s_cum_diff: 135612
 HoldTh: 5000, CurTh: 11250
 Gonna Go Up f_cum_diff: 140155, s_cum_diff: 135612
 Yes increasing
 Suggested rate: 120000

 *May 22 21:34:32.456 UTC: rx_bytes = 116730, tx_bytes = 125282, Suggested
 rate = 120000
```

```
*May 22 21:34:32.456 UTC: Exiting : 1
```

Additional References

The following sections provide references related to the Control Plane Logging feature.

Related Documents

Related Topic	Document Title
NHRP MIB	Dynamic Multipoint VPN Configuration Guide
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	Cisco IOS Quality of Service Solutions Command Reference
QoS feature overview	Quality of Service Overview module
Per-Tunnel QoS for DMVPN	Dynamic Multipoint VPN Configuration Guide

Standards

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

MIBs

MIB	MIBs Link
CISCO-CLASS-BASED-QOS-MIB CISCO-NHRP-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
None	—

Technical Assistance

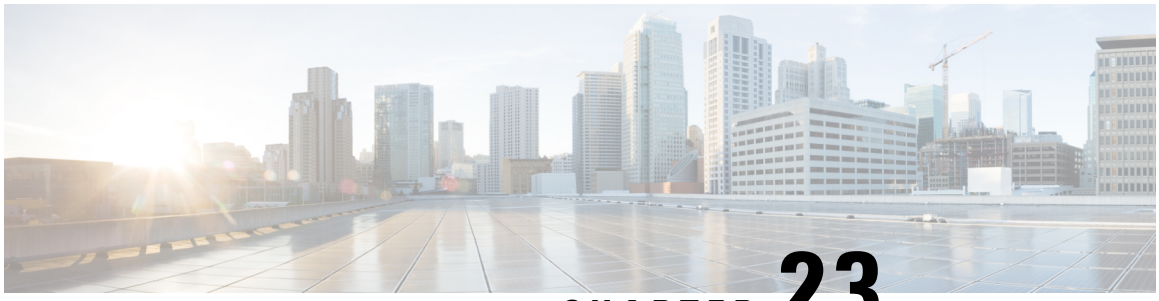
Description	Link
The Cisco Technical Support website contains thousands of pages of searchable technical content, including links to products, technologies, solutions, technical tips, and tools. Registered Cisco.com users can log in from this page to access even more content.	http://www.cisco.com/techsupport

Feature Information for Adaptive QoS over DMVPN

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 26: Feature Information for Adaptive QoS over DMVPN



CHAPTER 23

Regulating Packet Flow Using Traffic Shaping

This module contains an overview of regulating the packet flow on a network. Regulating the packet flow (that is, the flow of traffic) on the network is also known as traffic shaping. Traffic shaping allows you to control the speed of traffic that is leaving an interface. This way, you can match the flow of the traffic to the speed of the interface that is receiving the packet. Cisco provides a traffic-regulating mechanism called Class-Based Traffic Shaping. Before configuring this mechanism, it is important that you understand the overview presented in this module.

Information About Traffic Shaping

Benefits of Shaping Traffic on a Network

- Traffic shaping allows you to control the traffic going out an interface, matching the traffic flow to the speed of the interface.
- It ensures that traffic conforms to the policies contracted for it.
- It helps to ensure that a packet adheres to a stipulated contract, and it determines the appropriate quality of service to apply to the packet.
- It avoids bottlenecks and data-rate mismatches. For instance, central-to-remote site data speed mismatches.
- It prevents packet loss.

Here are some scenarios for which you would use traffic shaping:

- Control access to bandwidth when, for example, policy dictates that the rate of a given interface should not on the average exceed a certain rate even though the access rate exceeds the speed.
- Configure traffic shaping on an interface if you have a network with differing access rates. Suppose that one end of the link in a network runs at 256 kbps and the other end of the link runs at 128 kbps. Sending packets at 256 kbps could cause failure of the applications using the link.

A similar, more complicated case would be a link-layer network giving indications of congestion that has differing access rates on different attached data terminal equipment (DTE); the network may be able to deliver

more transit speed to a given DTE device at one time than another. (This scenario warrants that the token bucket be derived and that then its rate be maintained.)

- Offer a subrate service. In this case, traffic shaping enables you to use the router to partition your T1 or T3 links into smaller channels.

Token Bucket and Traffic Shaping

Traffic shaping uses a token bucket metaphor to shape traffic. A token bucket is a formal definition of a rate of transfer. It has three components: a burst size, a mean rate, and a time interval (Tc). Although the mean rate is generally represented as bits per second, any two values may be derived from the third by the relation shown as follows:

```
mean rate = burst size / time interval
```

Here are some definitions of these terms:

- Mean rate--Also called the committed information rate (CIR), it specifies how much data can be sent or forwarded per unit time on average.
- Burst size--Also called the committed burst (Bc) size, it specifies in bits (or bytes) per burst how much traffic can be sent within a given unit of time to not create scheduling concerns. (For a traffic shaper, it specifies bits per burst.)
- Time interval--Also called the measurement interval, it specifies the time quantum in seconds per burst.

By definition, over any integral multiple of the interval, the bit rate of the interface will not exceed the mean rate. The bit rate, however, may be arbitrarily fast within the interval.

A token bucket is used to manage a device that regulates the data in a flow. For example, the regulator might be a traffic shaper. A token bucket itself has no discard or priority policy. Rather, a token bucket discards tokens and leaves to the flow the problem of managing its transmission queue if the flow overdrives the regulator.

In the token bucket metaphor, tokens are put into the bucket at a certain rate. The bucket itself has a specified capacity. If the bucket fills to capacity, newly arriving tokens are discarded. Each token is permission for the source to send a certain number of bits into the network. To send a packet, the regulator must remove from the bucket a number of tokens equal in representation to the packet size.

If not enough tokens are in the bucket to send a packet, the packet waits until the bucket has enough tokens. If the bucket is already full of tokens, incoming tokens overflow and are not available to future packets. Thus, at any time, the largest burst a source can send into the network is roughly proportional to the size of the bucket.

Note that the token bucket mechanism used for traffic shaping has both a token bucket and a data buffer, or queue; if it did not have a data buffer, it would be a traffic policer. For traffic shaping, packets that arrive that cannot be sent immediately are delayed in the data buffer.

For traffic shaping, a token bucket permits burstiness but bounds it. It guarantees that the burstiness is bounded so that the flow will never send faster than the capacity of the token bucket plus the time interval multiplied by the established rate at which tokens are placed in the bucket. It also guarantees that the long-term transmission rate will not exceed the established rate at which tokens are placed in the bucket.

Traffic Shaping and Rate of Transfer

Traffic shaping limits the rate of transmission of data. You can limit the data transfer to one of the following:

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

As mentioned, the rate of transfer depends on these three components that constitute the token bucket: burst size, mean rate, 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 will 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.

One additional variable applies to traffic shaping: excess burst (Be) size. The Be size corresponds to the number of noncommitted bits--those outside the CIR--that are still accepted by the switch but marked as discard eligible (DE).

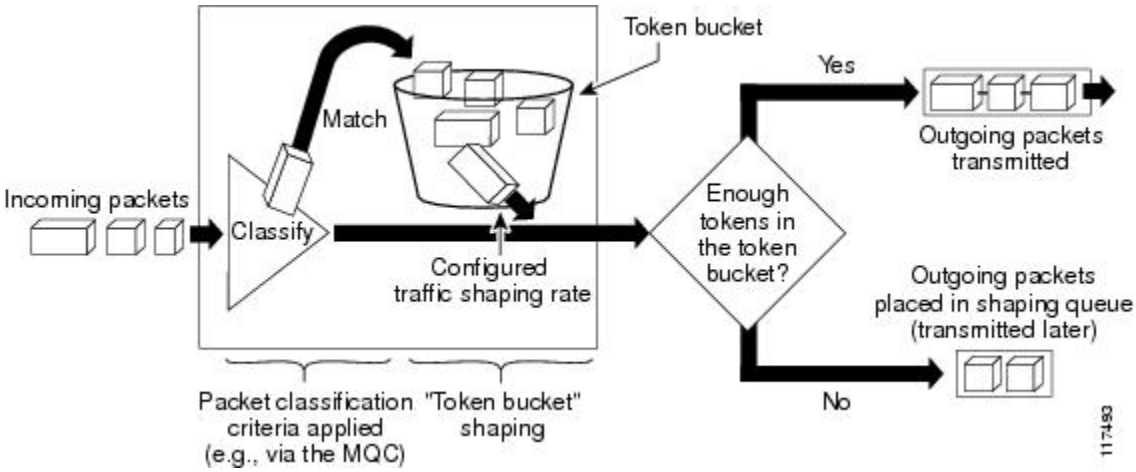
In other words, the Be size allows more than the burst size to be sent during a time interval in certain situations. The switch will allow the packets belonging to the excess burst to go through but it will mark them by setting the DE bit. Whether the packets are sent depends on how the switch is configured.

When the Be size 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 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.

How Traffic Shaping Regulates Traffic

The figure below illustrates how a traffic shaping mechanism regulates traffic.

Figure 8: How a Traffic-Shaping Mechanism Regulates Traffic



In the figure above, incoming packets arrive at an interface. The packets are classified using a "classification engine," such as an access control list (ACL) or the Modular Quality of Service Command-Line Interface (MQC). If the packet matches the specified classification, the traffic shaping mechanism continues. Otherwise, no further action is taken.

Packets matching the specified criteria are placed in the token bucket. The maximum size of the token bucket is the 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 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 the table below.

Table 27: 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 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, exceeding, or violating a specified rate, receive the configured policing action (for example, drop, send, mark then send). • Packets are not placed in queue for transmission later.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Packet classification	"Classifying Network Traffic" module
MQC, policy maps, class maps, and hierarchical policy maps	"Applying QoS Features Using the MQC" module
WFQ, CBWFQ, PQ, CQ, FIFO and other queueing mechanisms	"Congestion Management Overview" module
Class-Based Traffic Shaping	"Regulating Packet Flow on a Per-Class Basis -- Using Class-Based Traffic Shaping" module
GTS	"Regulating Packet Flow on a Per-Interface Basis -- Using Generic Traffic Shaping" module
FRTS	"MQC-Based Frame Relay Traffic Shaping" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

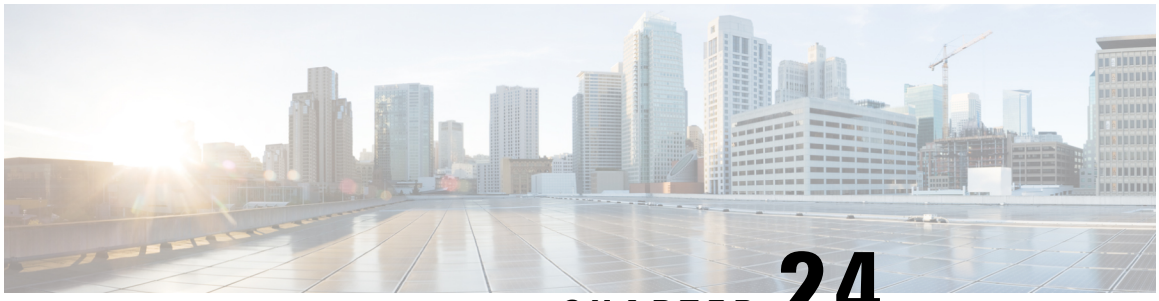
MIBs	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFCs	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 24

Regulating Packet Flow on a Per-Class Basis Using Class-Based Traffic Shaping

Packet flow on a network can be regulated using a traffic shaping mechanism. One such traffic shaping mechanism is a Cisco feature called Class-Based Traffic Shaping. Class-Based Traffic Shaping allows you to regulate the flow of packets (on a per-traffic-class basis) going out an interface, matching the packet flow to the speed of the interface. This module describes the concepts and tasks related to configuring Class-Based Traffic Shaping.

Prerequisites for Configuring Class-Based Traffic Shaping

Be familiar with the concepts in the "Regulating Packet Flow Using Traffic Shaping" module.

Use Feature Navigator to determine if the platform in use supports Class-Based Traffic Shaping. Access Cisco Feature Navigator at http://www.cisco.com/go/fn .

Distributed Cisco Express Forwarding (dCEF) must be enabled if the customer is using a Versatile Interface Processor (VIP) on the router.

A policy map and a class map must be created first using the Modular Quality of Service (QoS) Command-Line Interface (MQC).

Restrictions for Configuring Class-Based Traffic Shaping

Adaptive traffic shaping for Frame Relay networks is supported for Frame Relay networks only.

Class-Based Traffic Shaping applies to outbound traffic only.

Class-Based Traffic Shaping does not support the following commands:

- **traffic-shape adaptive**
- **traffic shape fecn-adaptive**
- **traffic-shape group**
- **traffic-shape rate**

Information About Class-Based Traffic Shaping

Class-Based Traffic Shaping Functionality

Class-Based Traffic Shaping is a traffic shaping mechanism (also known as a "traffic shaper"). A traffic shaper typically delays excess traffic using a buffer, or queueing mechanism, to hold packets and shape the flow when the data rate of the source is higher than expected. It holds and shapes traffic to a particular bit rate by using the token bucket mechanism. For more information about token buckets and traffic shaping, see the "Regulating Packet Flow Using Traffic Shaping" module.

Class-Based Traffic Shaping is the Cisco-recommended traffic shaping mechanism.

Note Class-Based Traffic Shaping should be used instead of what was previously referred to as Distributed Traffic Shaping (DTS). Class-Based Traffic Shaping can and should be used on the Cisco 7500 series router with a VIP2-40, VIP2-50, or greater processor.

Using the Class-Based Traffic Shaping, you can perform the following tasks:

- Configure traffic shaping on a per-traffic-class basis. It allows you to fine-tune traffic shaping for one or more classes and it allows you to configure traffic shaping on a more granular level.
- Specify average rate or peak rate traffic shaping. Specifying peak rate shaping allows you to make better use of available bandwidth by allowing more data than the configured traffic shaping rate to be sent if the bandwidth is available.
- Configure traffic shaping in a hierarchical policy map structure. That is, traffic shaping is configured in a primary-level (parent) policy map and other QoS features (for instance, CBWFQ and traffic policing) can be configured in the secondary-level (child) policy maps. For more information, see the Hierarchical Policy Map Structure of Class-Based Traffic Shaping, on page 245.

Benefits of Class-Based Traffic Shaping

All of the benefits associated with traffic shaping also apply to Class-Based Traffic Shaping, but on a more granular level. For information about the benefits of traffic shaping, see the "Regulating Packet Flow Using Traffic Shaping" module.

Hierarchical Policy Map Structure of Class-Based Traffic Shaping

With the Class-Based Traffic Shaping mechanism, traffic shaping can be configured in a hierarchical policy map structure; that is, traffic shaping is enabled in a primary-level (parent) policy map and other QoS features used with traffic shaping, such as CBWFQ and traffic policing, can be enabled in a secondary-level (child) policy map.

Traffic shaping is enabled by using the **shape** command (and specifying a rate) in a policy map. When traffic shaping is enabled, one the following actions occur:

- Packets exceeding the specified rate are placed in a queue using an appropriate queueing mechanism.
- Packets conforming to the specified rate are transmitted.

When packets are placed in a queue, the default queueing mechanism used is weighted fair queueing (WFQ). However, with Class-Based Traffic Shaping, class-based WFQ (CBWFQ) can be configured as an alternative queueing mechanism.

CBWFQ allows you to fine-tune the way traffic is placed in a queue. For instance, you can specify that all voice traffic be placed in a high-priority queue and all traffic from a specific class be placed in a lower-priority queue.

If you want to use CBWFQ with the Class-Based Traffic Shaping mechanism, the following conditions must be met:

- A secondary-level (child) policy map *must* be created. This secondary-level (child) policy map is then used to configure CBWFQ by enabling the **bandwidth** command.
- Traffic shaping *must* be configured in the primary-level (parent) policy map.

Note CBWFQ is supported in both the primary-level (parent) policy map and the secondary-level (child) policy map. However, to use CBWFQ at the secondary-level (child) policy map, traffic shaping *must* be configured in the primary-level (parent) policy map.

The following sample configuration illustrates how the Class-Based Traffic Shaping mechanism is configured in a hierarchical policy map structure:

```
enable
configure terminal
 policy-map policy_parent        ! This is the primary-level policy map.
  class class-default
   shape average 1000000         ! This enables traffic shaping.
   service-policy policy_child   ! This associates the policy maps.
```

Traffic shaping must be configured in the primary-level (parent) policy map. With this configuration, WFQ is used as the default queueing mechanism for placing all the traffic in a queue.

In the following secondary-level (child) policy map, the alternative queueing mechanism CBWFQ is configured:

```
enable
configure terminal
 policy-map policy_child     ! This is the secondary-level policy map.
```

```
class class-default
 bandwidth percent 50      ! This enables CBWFQ.
```

In the secondary-level (child) policy map, additional QoS features used with traffic shaping (for example, CBWFQ and traffic policing) are typically configured. For Class-Based Traffic Shaping, the only two QoS features supported at the secondary-level (child) policy map are CBWFQ and traffic policing.

How to Configure Class-Based Traffic Shaping

Configuring Class-Based Traffic Shaping in a Primary-Level Policy Map

Traffic shaping is configured in a policy map. Policy maps determine the specific quality of service (QoS) feature that will be applied to traffic on a network. In this module, the QoS feature being applied is traffic shaping.

Traffic shaping is configured in the primary-level (parent) policy map in the hierarchy.

Before you begin

Before configuring traffic shaping, you must use the MQC to create a policy map and a class map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **shape** [**average** | **peak**] *mean-rate* [*burst-size*] [*excess-burst-size*]
6. **service-policy** *policy-map-name*
7. **end**
8. **show policy-map**
9. **show policy-map interface** *type number*
10. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy_parent`	Specifies the name of the policy map created earlier and enters policy-map configuration mode. See Prerequisites for Configuring Class-Based Traffic Shaping, on page 243 for more information. • Enter the policy map name.
Step 4	**class** {*class-name* | **class-default**} **Example:** `Router(config-pmap)# class class-default`	Specifies the name of the class whose policy you want to create and enters policy-map class configuration mode. • Enter the name of the class or enter the **class-default**keyword.
Step 5	**shape** [**average** | **peak**] *mean-rate* [*burst-size*] [*excess-burst-size*] **Example:** `Router(config-pmap-c)# shape average 1000000`	Shapes traffic according to the keyword and rate specified. • Enter the keyword and rate.
Step 6	**service-policy** *policy-map-name* **Example:** `Router(config-pmap-c)# service-policy policy_child`	Uses a service policy as a QoS policy within a policy map (called a hierarchical service policy). • Enter the policy map name.
Step 7	**end** **Example:** `Router(config-pmap-c)# end`	Returns to privileged EXEC mode.
Step 8	**show policy-map** **Example:** `Router# show policy-map`	(Optional) Displays all configured policy maps.

Step 9	**show policy-map interface** *type number*
	Example:
	`Router#`
	`show policy-map interface serial4/0`
		(Optional) Displays the packet statistics of all classes that are configured for all service policies either on the specified interface or subinterface or on a specific PVC on the interface.
		• Enter the interface type and number.

Step 10	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

What to Do Next

To configure a secondary-level (child) policy map in the hierarchical policy map structure (an optional task), proceed with the instructions in Configuring the Secondary-Level Policy Map.

Configuring the Secondary-Level Policy Map

Note CBWFQ is supported in both the primary-level (parent) policy map and the secondary-level (child) policy map. However, to use CBWFQ in the secondary-level (child) policy map, traffic shaping *must* be configured in the primary-level (parent) policy map. For more information about CBWFQ in a secondary-level (child) policy map, see the Hierarchical Policy Map Structure of Class-Based Traffic Shaping, on page 245.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*}
6. **end**
7. **show policy-map**
8. **show policy-map interface** *type number*
9. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy1`	Specifies the name of the policy map created earlier and enters policy-map configuration mode. See Prerequisites for Configuring Class-Based Traffic Shaping, on page 243 for more information. Enter the policy map name.
Step 4	**class** {*class-name* | **class-default**} **Example:**	Specifies the name of the class whose policy you want to create and enters policy-map class configuration mode.

	Command or Action	Purpose
	`Router(config-pmap)# class class-default`	• Enter the name of the class or enter the **class-default** keyword.
Step 5	**bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*} **Example:** `Router(config-pmap-c)# bandwidth percent 50` **Example:**	Specifies or modifies the bandwidth allocated for a class belonging to a policy map. • Enter the amount of bandwidth as a number of kbps, a relative percentage of bandwidth, or an absolute amount of bandwidth. **Note** The **bandwidth** command used here is only an example of a QoS feature than can be configured. The **bandwidth** command configures CBWFQ. You could also use the **police** command to configure traffic policing.
Step 6	**end** **Example:** `Router(config-pmap-c)# end`	Returns to privileged EXEC mode.
Step 7	**show policy-map** **Example:** `Router# show policy-map`	(Optional) Displays all configured policy maps.

Step 8	**show policy-map interface** *type number*
	Example:
	`Router#`
	`show policy-map interface serial4/0`
		(Optional) Displays the packet statistics of all classes that are configured for all service policies either on the specified interface or subinterface or on a specific PVC on the interface.
		• Enter the interface type and number.

Step 9	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Configuration Examples for Class-Based Traffic Shaping

Example Class-Based Traffic Shaping Configuration

The following is an example of Class-Based Traffic Shaping configured in a hierarchical policy map structure. In this example, two policy maps have been created; the primary-level (parent) policy map called "policy_parent," and a secondary-level (child) policy map called "policy_child." Traffic shaping is configured in the policy_parent policy map, and CBWFQ has been configured in the policy_child policy map.

The **service-policy** command associates the two policy maps in the hierarchical policy map structure.

```
enable
configure terminal
 policy-map policy_parent
  class class-default
   shape average 1000000         ! This enables traffic shaping.
   service-policy policy_child   ! This associates the policy maps.
   exit
  exit
 policy-map policy_child
  class class-default
   bandwidth percent 50   ! This enables CBWFQ.
   end
```

Where to Go Next

To configure Generic Traffic Shaping (GTS), see the "Regulating Packet Flow on a Per-Interface Basis Using Generic Traffic Shaping" module.

To configure Frame Relay Traffic Shaping (FRTS), see the "MQC-Based Frame Relay Traffic Shaping" module.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Packet classification	"Classifying Network Traffic" module
MQC, policy maps, class maps, and hierarchical policy maps	"Applying QoS Features Using the MQC" module
CBWFQ and other queueing mechanisms	"Configuring Weighted Fair Queueing" module
Overview information about using traffic shaping to regulate packet flow on a network	"Regulating Packet Flow Using Traffic Shaping" module
GTS	"Regulating Packet Flow on a Per-Interface Basis Using Generic Traffic Shaping" module
FRTS	"MQC-Based Frame Relay Traffic Shaping" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

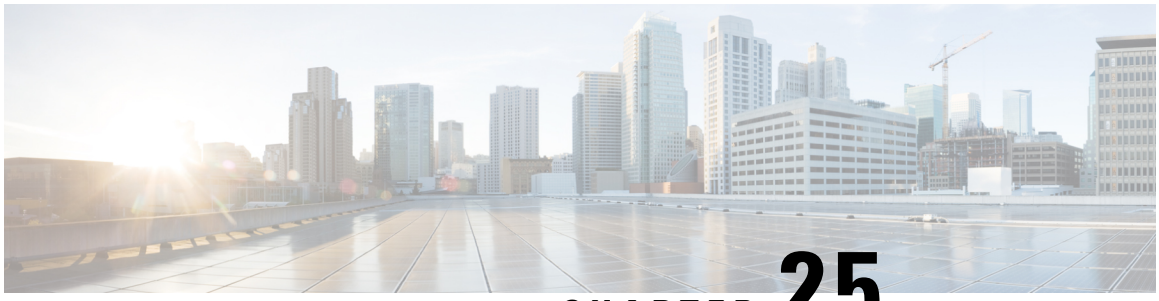
Feature Information for Class-Based Traffic Shaping

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 28: Feature Information for Class-Based Traffic Shaping

Feature Name	Software Releases	Feature Configuration Information
Distributed Traffic Shaping	12.2(8)T	Distributed Traffic Shaping (DTS) is a legacy method for regulating the flow of packets going out an interface. Class-Based Traffic Shaping should be used instead of (DTS).
Generic Traffic Shaping (GTS)	15.0(1)S	The GTS feature was integrated into the Cisco IOS Release 15.0(1)S release.



CHAPTER 25

Service Groups

This feature provides the ability to apply an aggregate QoS service policy across multiple VLAN subinterfaces or service instances that are on the same physical interface. The Service Group feature allows network administrators to create service groups, add members (such as service instances) to those service groups, and apply service policies to the groups. The service policies contain the aggregate features (such as traffic policing and queueing) that can be applied to the groups. These service policies are in compliance with the Service-Level Agreement (SLA) negotiated between the service provider and the subscribers.

Restrictions for Service Groups

- Only EFP service instances, routed sub-interfaces and aggregate port-channel sub-interfaces can be added as members of service groups.

  Each service instance or sub-interface can belong to only one service group at time.

- The service group must exist before any member can join the group.
- All members of a service group must reside on the same physical interface or same aggregate port-channel interface.
- Sub-interfaces or service instances that are members of a service group cannot have a QoS policy applied to the interfaces, even if the service group does not have a QoS policy applied.
- MPOL is not supported on aggregate port-channel when policy is applied on aggregated port-channel main interface, port-channel sub-interface cannot be attached by any policy, or be configured as a member of a service-group.
- Sub-interface belongs to service group and sub-interface applied with service-policy cannot be configured on the same aggregate port-channel simultaneously.
- Each sub-interface belongs to only one service group at a time.

- Interfaces that are a member of a service group cannot have a QoS policy applied.
- A batch configuration including both "define service-group" and "add sub-interface to service-group" may result in membership error, and vice versa in the unconfiguration.

  So it is recommended to define the service-group before adding subinterfaces or service instances to it, and removing them from the service-group before deleting the service-group or deleting the subinterfaces or service instances.

Information About Service Groups

Service Instances and Service Groups

A service instance is a configuration object (container) that holds all management and control plane attributes and parameters that apply to that service instance on a per-port basis. Different service instances that correspond to the same Ethernet Virtual Connection (EVC) must share the same name. Service instances are associated with a global EVC object through their shared name.

The Service Groups feature allows you to create service groups and apply aggregate features to those service groups. Service groups are created with input and output policies. Members join these groups by configuring the group ID in their configuration.

Make note of the following actions when enabling the service group feature:

- A service group must be created before a QoS policy can be configured on the service group.
- A service group sub-interface or service instance must be created before it can be bound to its group interface.

When disabling the service group feature:

- A service group sub-interface or service instance must be unbound from the service group interface before the service member interface is deleted.
- A service group sub-interface or service instance unbound from the service group interface before the service group interface is deleted.
- A QoS policy must be removed from the service group interface before the service group interface is deleted.

How to Configure Service Groups

Creating a Service Group

Before you begin

In this procedure, you need to specify the name of a QoS policy to be attached to the service group. The QoS policy must already exist.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **service-group** *service-group-identifier*
4. **description** *descriptive-text*
5. **service-policy** {**input** | **output**} *policy-map-name*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service-group** *service-group-identifier* **Example:** `Device(config)# service-group 20`	Creates a service group and enters service-group configuration mode. • Enter the service group number. The number of service groups that can be created varies by Device.
Step 4	**description** *descriptive-text* **Example:** `Device(config-service-group)# description subscriber account number 105AB1`	(Optional) Creates a description of the service group. • Enter a description (for example, additional information about the group) of the service group. Descriptions can be a maximum of 240 characters.
Step 5	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Device(config-service-group)# service-policy input policy1`	(Optional) Attaches a policy map to the service group, in either the ingress (input) or egress (output) direction. • Enter either the **input** or **output** keyword and the name of the previously created policy map.
Step 6	**end** **Example:** `Device(config-service-group)# end`	(Optional) Returns to privileged EXEC mode.

Adding or Deleting Service Group Members

Note The following restrictions apply to service group members:

- A member can join only one service group at a time.
- All members of a service group must reside on the same physical interface.
- Service instances cannot join the same group from multiple interfaces. Group members must come from the same interface, as shown in the sample configuration below:

```
interface GigabitEthernet 2/0/0
service instance 1 ethernet
group 32
service instance 2 ethernet
group 32
interface GigabitEthernet 2/0/0.2
encapsulation dot1q 2
group 37
interface GigabitEthernet 2/0/1
service instance 1 ethernet
group 32  |<--Disallowed because this group has members in g2/0/0 already |
>
```

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service instance** *service-instance-number* **ethernet**
5. **group** *service-group-identifier*
6. **no group** *service-group-identifier*
7. **exit**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Configures an interface and enters interface configuration mode.
Step 4	**service instance** *service-instance-number* **ethernet** **Example:** `Device(config-if)# service instance 200 ethernet`	Specifies the service instance to be added or deleted from a service group and enters service configuration mode.
Step 5	**group** *service-group-identifier* **Example:** `Device(config-if-srv)# group 20`	Number of the service group specified by the member will be added.
Step 6	**no group** *service-group-identifier* **Example:** `Device(config-if-srv)# no group 20`	(Optional) Number of the service group specified by the member will be added.
Step 7	**exit** **Example:** `Device(config-if-srv)# exit`	(Optional) Returns to interface configuration mode.
Step 8	**end** **Example:** `Device(config-if-srv)# end`	(Optional) Returns to privileged EXEC mode.

Deleting a Service Group

Before you begin

- A service member interface must be unbound from the service group interface before the service group interface is deleted.
- A QoS policy must be removed from the service group interface before the service group interface is deleted.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no service-group** *service-group-identifier*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no service-group** *service-group-identifier* **Example:** `Device(config)# no service-group 20`	Deletes a service group and deletes all members from the service group. • Enter the service group number to be deleted. **Note** When you delete a service group, all members of the service group are automatically removed from the service group.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Exits global configuration mode.

Verifying the Service Group Configuration

SUMMARY STEPS

1. **enable**
2. **show running-config service-group**
3. **show service-group** {*service-group-identifier* | **all**}
4. **show service-group interface** *type number*
5. **show service-group stats**
6. **show service-group state**
7. **show service-group traffic-stats**
8. **show policy-map interface** *type number* **service group** {*service-group-identifier*
9. **show policy-map target service-group** {*service-group-identifier*}
10. **show ethernet service instance** [**detail**]
11. **clear service-group traffic-stats**
12. **debug service-group** {**all** | **error** | **feature** | **group** | **interface** | **ipc** | **member** | **qos** | **stats**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show running-config service-group** **Example:** `Device# show running-config service-group`	(Optional) Displays the running service-group configuration.
Step 3	**show service-group** {*service-group-identifier* | **all**} **Example:** `Device# show service-group all`	(Optional) Displays service-group configuration information for one or all service groups.
Step 4	**show service-group interface** *type number* **Example:** `Device# show service-group interface gigabitethernet 3/1`	(Optional) Displays service-group membership information by interface. • Enter the interface type and number.
Step 5	**show service-group stats** **Example:** `Device# show service-group stats`	(Optional) Displays service-group statistical information.
Step 6	**show service-group state** **Example:** `Device# show service-group state`	(Optional) Displays state information about service groups.
Step 7	**show service-group traffic-stats** **Example:** `Device# show service-group traffic-stats`	(Optional) Displays traffic statistics for all the members of a service group. • The information displayed is the combined total of the traffic statistics for all members.
Step 8	**show policy-map interface** *type number* **service group** {*service-group-identifier* **Example:** `Device# show policy-map interface gigabitEthernet 9/5 service group`	(Optional) Displays policy-map information for service groups. • Enter the interface type and number.

	Command or Action	Purpose
Step 9	**show policy-map target service-group** {*service-group-identifier*} **Example:** `Device# show policy-map target service-group 1`	(Optional) Displays policy-map information for service groups that have members attached to the specified interface. • Enter the service group identifier.
Step 10	**show ethernet service instance** [**detail**] **Example:** `Device# show ethernet service instance detail`	(Optional) Displays information about the service instances. **Note** To display the service group number, use the **detail** keyword.
Step 11	**clear service-group traffic-stats** **Example:** `Device# clear service-group traffic-stats`	(Optional) Clears the traffic statistics for the service group. **Note** Clearing the traffic statistics for the service group does not clear the traffic statistics for the group members. To clear the traffic statistics for group members, use the **clear ethernet service instance** command. For more information about the **clear ethernet service instance** command, see the Cisco IOS Carrier Ethernet Command Reference .
Step 12	**debug service-group** {**all** | **error** | **feature** | **group** | **interface** | **ipc** | **member** | **qos** | **stats**} **Example:** `Device# debug service-group qos`	(Optional) Debugs service-group events and errors.

Adding or Deleting a Subinterface from a Service Group

Note If a subinterface is already a member of a group, you cannot add it to another group. To move a subinterface, first delete it from the current group, then add it to the new group.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **group** *service-group-identifier*
5. **no group** *service-group-identifier*
6. **exit**
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**interface** *type number*
	Example:
	`Device(config)# interface GigabitEthernet 1/1/0.30`
	`Device(config)# interface range GigabitEthernet 1/1/0.30 - GigabitEthernet 1/1/0.36`
		Configures a subinterface and enters subinterface configuration mode.
		• Enter the interface type and the subinterface number.
		or
		Configures a range of subinterfaces and enters subinterface configuration mode.
		• Enter the interface types and the subinterface numbers.

Step 4	**group** *service-group-identifier* **Example:** `Device(config-subif)# group 20`	Number of the service group to which the subinterfaces will be added. • Enter the service group number.
Step 5	**no group** *service-group-identifier* **Example:** `Device(config-subif)# no group 30`	(Optional) Number of the service group from which the subinterfaces will be deleted. • Enter the service group number.
Step 6	**exit** **Example:** `Device(config-subif)# exit`	(Optional) Returns to interface configuration mode.
Step 7	**end** **Example:** `Device(config-subif)# end`	(Optional) Returns to privileged EXEC mode.

Verifying the Subinterface Configuration

SUMMARY STEPS

1. **enable**
2. **show running-config service-group**
3. **show service-group** {*service-group-identifier* | **all**}
4. **show service-group interface** *type number*
5. **show policy-map target service-group** *service-group-identifier*
6. **show service-group stats**
7. **show service-group state**
8. **show service-group traffic-stats**
9. **clear service-group traffic-stats**
10. **debug service-group** {**all** | **error** | **feature** | **group** | **interface** | **ipc** | **member** | **qos** | **stats**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show running-config service-group** **Example:** `Device# show running-config service-group`	(Optional) Displays the running service-group configuration.
Step 3	**show service-group** {*service-group-identifier*	**all**} **Example:** `Device# show service-group all`	(Optional) Displays service-group configuration information for one or all service groups.
Step 4	**show service-group interface** *type number* **Example:** `Device# show service-group interface gigabitethernet 3/1`	(Optional) Displays service-group membership information by interface. • Enter the interface type and number.
Step 5	**show policy-map target service-group** *service-group-identifier* **Example:** `Device# show policy-map target service-group 1`	(Optional) Displays the policy-map information for all service groups or the specified service group. • Enter the target and service group.
Step 6	**show service-group stats** **Example:**	(Optional) Displays service-group statistical information.

	Command or Action	Purpose
	`Device# show service-group stats`	
Step 7	**show service-group state** **Example:** `Device# show service-group state`	(Optional) Displays state information about service groups.
Step 8	**show service-group traffic-stats** **Example:** `Device# show service-group traffic-stats`	(Optional) Displays the traffic statistics for all the members of a service group. • The information displayed is the combined total of the traffic statistics for all members.
Step 9	**clear service-group traffic-stats** **Example:** `Device# clear service-group traffic-stats`	(Optional) Clears the traffic statistics for the service group. **Note** Clearing the traffic statistics for the service group does not clear the traffic statistics for the group members. To clear the traffic statistics for group members, use the **clear ethernet service instance** command. For more information about the **clear ethernet service instance** command, see the *Cisco IOS Carrier Ethernet Command Reference*.
Step 10	**debug service-group** {**all** | **error** | **feature** | **group** | **interface** | **ipc** | **member** | **qos** | **stats**} **Example:** `Device# debug service-group qos`	(Optional) Debugs service-group events and errors.

Configuration Examples for Service Groups

Example Creating a Service Group

In the following example, service group 20 has been created:

```
Device> enable
Device# configure terminal
Device(config)# service-group 20
Device(config-service-group)# description account number 105AB1
Device(config-service-group)# service-policy input policy1
Device(config-service-group)# end
```

Example Adding Service Instance Members to a Service Group

In the following example, service instance 200 will be added to service group 20:

```
Device> enable

Device# configure terminal

Device(config)# interface GigabitEthernet 1/0

Device(config-if)# service instance 200 ethernet

Device(config-if-srv)# group 20

Device(config-if-srv)# end
```

Example Adding Subinterfaces to a Service Group

In the following example, subinterface g3/7.12 will be added to service group 10:

```
Device> enable

Device# configure terminal

Device(config)# interface GigabitEthernet 3/7.12

Device(config-subif)# group 10

Device(config-subif)# end
```

Example Deleting Service Instance Members from a Service Group

In the following example, service instance 300 will be deleted from service group 30 on a port channel:

```
Device> enable
Device# configure terminal
Device(config)# interface GigabitEthernet 1/0.56 or
Device(config-if)# service instance 300 ethernet
Device(config-if-srv)# no group 30
Device(config-if-srv)# end
```

Example Deleting Subinterfaces from a Service Group

In the following example, subinterface g3/7.12 will be deleted from service group 10:

```
Device> enable
Device# configure terminal
Device(config)# interface g3/7.12
Device(config-subif)# no group 10
Device(config-subif)# end
```

Example Deleting a Service Group

In the following example, service group 20 will be deleted:

```
Device> enable

Device# configure terminal

Device(config)# no service-group 20

Device(config)# end
```

Example Verifying the Service Group Configuration

This section contains sample output from the **show policy-map target service-group**command. The **show policy-map target service-group** command displays policy-map information for service groups.

Note This command is one of several that you can use to verify the service-group configuration. For additional commands that can be used, see *Verifying the Service Group Configuration*.

In the following example, service group 1 is specified. Service group 1 contains two policy maps (service policies), policy1 and policy2. Traffic policing is enabled in the EVC policy map. Traffic queuing is enabled in the ISG policy map.

```
Device# show policy-map target service-group 1

 GigabitEthernet9/5: Service Group 1

  Service-policy input: policy1

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
      police:
          cir 200000 bps, bc 6250 bytes
        conformed 0 packets, 0 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps

  Service-policy output: policy2

  Counters last updated 00:00:34 ago
    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
      Queueing
      queue limit 131072 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
```

```
(pkts output/bytes output) 0/0
bandwidth remaining ratio 2
```

How to Configure Service-group Support on Aggregate Port-channel

Adding Service Instance Members to a Service Group

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **service-group** *service-group-identifier*
4. **service-policy** {**input** | **output**} *policy-map-name*
5. **platform qos port-channel-aggregate** *port-channel-number*
6. **interface port-channel** *port-channel-number*
7. **interface** *interface*
8. **channel-group** *number*
9. **interface port-channel** *port-channel-number.subinterface-number*
10. **encapsulation dot1Q** *vlan-id* **second-dot1q** *vlan-id*
11. **group** *service-group-identifier*
12. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**service-group** *service-group-identifier* **Example:** `Device(config)# service-group 10`	Adds a service group and all members to the service group.

Step 4	**service-policy** {**input** | **output**} *policy-map-name*
	Example:
	`Device(config-service-group)# service-policy input policy1`
	`Device(config-service-group)# service-policy output policy2`
		(Optional) Attaches a policy map to the service group, in either the ingress (input) or egress (output) direction.
		• Enter either the **input** or **output** keyword and the name of the previously created policy map.

	Command or Action	Purpose
Step 5	**platform qos port-channel-aggregate** *port-channel-number* **Example:** `Device(config)# platform qos port-channel-aggregate 1`	Enables aggregate mode for a port-channel interface. **Note** It must be configured before a port-channel is created. Enable aggregate mode before a port-channel interface is attached by policy, or subinterfaces of the port-channel to be added to a service-group.
Step 6	**interface port-channel** *port-channel-number* **Example:** `Device(config)# interface port-channel 1`	Enters interface configuration mode to configure a specific port channel.
Step 7	**interface** *interface* **Example:** `Device(config)# interface g0/0/0`	Configures physical interface as a member link of the port-channel.
Step 8	**channel-group** *number* **Example:** `Device(config)# channel-group 1`	Adds the physical interface to the port-channel 1 as a member link.
Step 9	**interface port-channel** *port-channel-number.subinterface-number* **Example:** `Device(config)# interface port-channel 1.10`	Enters interface configuration mode to configure a specific port channel subinterface.
Step 10	**encapsulation dot1Q** *vlan-id* **second-dot1q** *vlan-id* **Example:** `Device(config-subif)# encapsulation dot1Q 10 second-dot1q 11`	Defines the matching criteria to map Q-in-Q ingress frames on the port-channel subinterface. **Note** Configuring second-dot1q is optional.
Step 11	**group** *service-group-identifier* **Example:** `Device(config-subif)# group 10`	Adds the port-channel sub interface to the specified service-group.
Step 12	**end** **Example:** `Device(config-subif)# end`	Returns to privileged EXEC mode.

Deleting Service Instance Members from a Service Group

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **interface port-channel** *port-channel-number.subinterface-number*
4. **no group** *service-group-identifier*
5. **no service-group** *service-group-identifier*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-number.subinterface-number* **Example:** `Device(config)# interface port-channel 1.10`	Enters interface configuration mode to configure a specific port channel subinterface.
Step 4	**no group** *service-group-identifier* **Example:** `Device(config-subif)# no group 10`	Removes the port-channel sub-interface from the service group specified by the number.
Step 5	**no service-group** *service-group-identifier* **Example:** `Device(config-subif)# no service-group 10`	Deletes a service group. **Note** All members should be removed from the service group first.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuration Examples for Service-group on Aggregate Port-channel

Example: Adding Service Instance Members to a Service Group

```
Device> enable
Device# configure terminal
Device(config)# service-group 10
Device(config-service-group)# service-policy input policy1
```

```
Device(config-service-group)# service-policy output policy2
Device(config)# platform qos port-channel-aggregate 1
Device(config)# interface port-channel 1
Device(config)# interface g0/0/0
Device(config-if)# channel-group 1
Device(config)# interface port-channel 1.10
Device(config-subif)# encapsulation dot1Q 10 second-dot1q 11
Device(config-subif)# group 10
Device(config-subif)# end
```

Example: Deleting Service Instance Members to a Service Group

```
Device> enable
Device# configure terminal
Device(config)# interface port-channel 1.10
Device(config-subif)# no group 10
Device(config-subif)# no service-group 10
Device(config)# end
```

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples.	*Cisco IOS Quality of Service Solutions Command Reference*
Debug commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples.	*Cisco IOS Debug Command Reference*
MQC, policy maps	"Applying QoS Features Using the MQC" module
Service instance configuration information and concepts	*Cisco IOS Carrier Ethernet Configuration Guide*
Service instance commands	*Cisco IOS Carrier Ethernet Command Reference*

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	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 Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

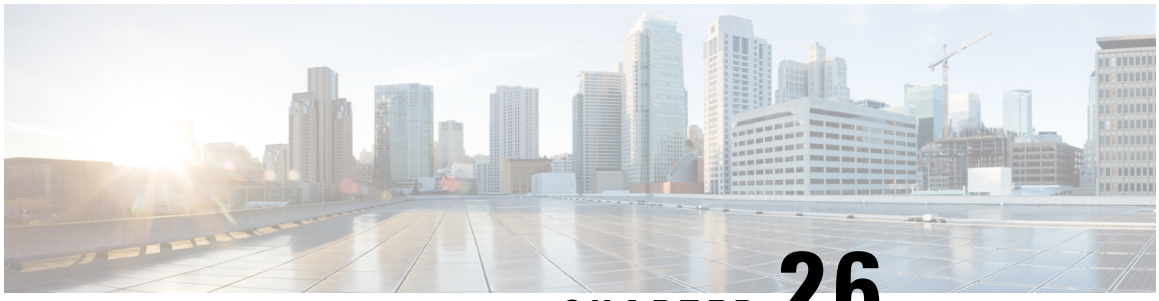
Feature Information for Service Groups

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 29: Feature Information for Service Groups

Service Groups	12.2(33)SRE	The Service Groups feature allows network administrators to create service groups, add members (such as service instances) to those service groups, and apply service policies (also known as policy maps) to those newly created groups. In Release 12.2(33)SRE, this feature was introduced on the Cisco 7600 series router. The following commands were introduced or modified: **clear service-group traffic-stats**, **debug service-group**, **description**, **group, service-group**, **service instance ethernet, service-policy, show policy-map interface service group**, **show running-config service-group**, **show service-group**, **show service-group interface**, **show service-group state**, **show service-group stats**, **show service-group traffic-stats**.



CHAPTER 26

Header Compression

Header compression is a mechanism that compresses the IP header in a packet before the packet is transmitted. Cisco provides two types of header compression: RTP header compression (used for RTP packets) and TCP header compression (used for TCP packets).

This module contains a high-level overview of header compression. Before configuring header compression, you need to understand the information contained in this module.

Information About Header Compression

Header Compression Defined

Header compression is a mechanism that compresses the IP header in a data packet before the packet is transmitted. Header compression reduces network overhead and speeds up the transmission of Real-Time Transport Protocol (RTP) and Transmission Control Protocol (TCP) packets. Header compression also reduces the amount of bandwidth consumed when the RTP or TCP packets are transmitted.

Types of Header Compression

Cisco provides the following two types of header compression:

- RTP header compression (used for RTP packets)
- TCP header compression (used for TCP packets)

Both RTP header compression and TCP header compression treat packets in a similar fashion, as described in the sections that follow.

Note RTP and TCP header compression are typically configured on a *per-interface* (or *subinterface*) basis. However, you can choose to configure either RTP header compression or TCP header compression on a *per-class* basis using the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC). More information about class-based RTP and TCP header compression is provided later in this module.

RTP Functionality and Header Compression

RTP provides end-to-end network transport functions for applications that support audio, video, or simulation data over unicast or multicast services.

RTP provides support for real-time conferencing of groups of any size within the Internet. This support includes source identification support for gateways such as audio and video bridges, and support for multicast-to-unicast translators. RTP provides QoS feedback from receivers to the multicast group and support for the synchronization of different media streams.

RTP includes a data portion and a header portion. The data portion of RTP is a thin protocol that provides support for the real-time properties of applications, such as continuous media, including timing reconstruction, loss detection, and content identification. The header portion of RTP is considerably larger than the data portion. The header portion consists of the IP segment, the User Datagram Protocol (UDP) segment, and the RTP segment. Given the size of the IP/UDP/RTP segment combinations, it is inefficient to send the IP/UDP/RTP header without compressing it.

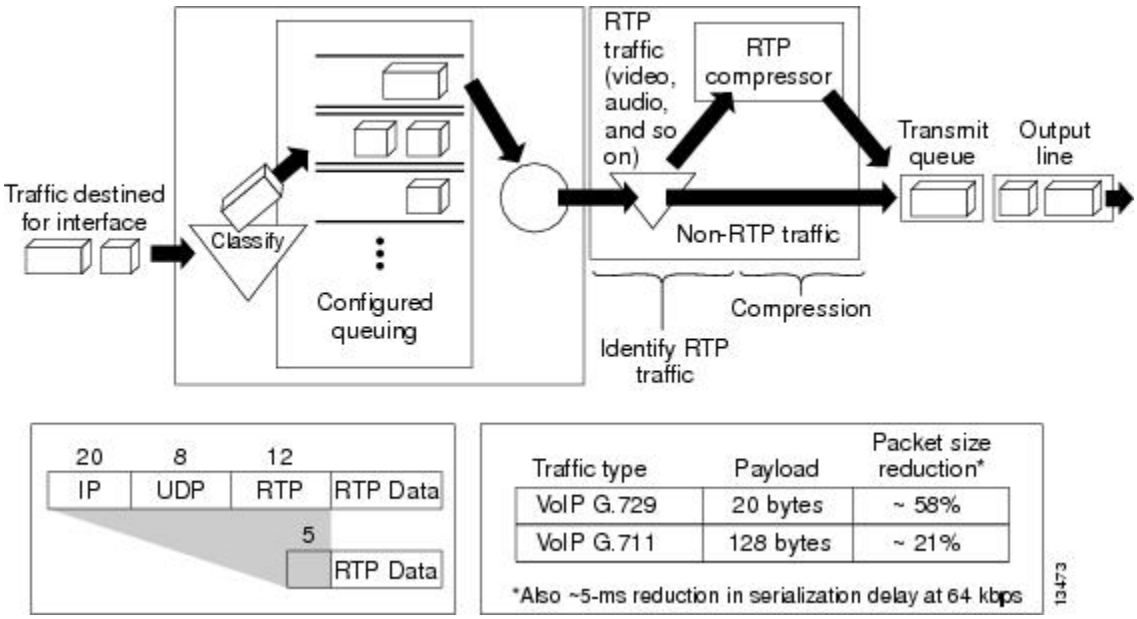
To avoid the unnecessary consumption of available bandwidth, RTP header compression is used on a link-by-link basis.

How RTP Header Compression Works

RTP header compression compresses the RTP header (that is, the combined IP, UDP, and RTP segments) in an RTP packet. the figure below illustrates this process and shows how RTP header compression treats incoming packets.

In this example, packets arrive at an interface and the packets are classified. After the packets are classified, they are queued for transmission according to the configured queuing mechanism.

Figure 9: RTP Header Compression



For most audio applications, the RTP packet typically has a 20- to 128-byte payload.

RTP header compression identifies the RTP traffic and then compresses the IP header portion of the RTP packet. The IP header portion consists of an IP segment, a UDP segment, and an RTP segment. In the figure above, the minimal 20 bytes of the IP segment, combined with the 8 bytes of the UDP segment, and the 12 bytes of the RTP segment, create a 40-byte IP/UDP/RTP header. In the figure above, the RTP header portion is compressed from 40 bytes to approximately 5 bytes.

Note RTP header compression is supported on serial interfaces using Frame Relay, HDLC, or PPP encapsulation. It is also supported over ISDN interfaces.

Why Use RTP Header Compression

RTP header compression accrues major gains in terms of packet compression because although several fields in the header change in every packet, the difference from packet to packet is often constant, and therefore the second-order difference is zero. The decompressor can reconstruct the original header without any loss of information.

RTP header compression also reduces overhead for multimedia RTP traffic. The reduction in overhead for multimedia RTP traffic results in a corresponding reduction in delay; RTP header compression is especially beneficial when the RTP payload size is small, for example, for compressed audio payloads of 20 to 50 bytes.

Use RTP header compression on any WAN interface where you are concerned about bandwidth and where there is a high portion of RTP traffic. RTP header compression can be used for media-on-demand and interactive services such as Internet telephony. RTP header compression provides support for real-time conferencing of groups of any size within the Internet. This support includes source identification support for gateways such as audio and video bridges, and support for multicast-to-unicast translators. RTP header compression can benefit both telephony voice and multicast backbone (MBONE) applications running over slow links.

Note Using RTP header compression on any high-speed interfaces--that is, anything over T1 speed--is not recommended. Any bandwidth savings achieved with RTP header compression may be offset by an increase in CPU utilization on the router.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands	*Cisco IOS QoS Command Reference*
MQC	"Applying QoS Features Using the MQC"
RTP header compression	"Configuring RTP Header Compression"

Standards and RFCs

Standard/RFC	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--
• RFC 1144 • RFC 2507 • RFC 2508 • RFC 3544 • RFC 3550	• Compressing TCP/IP Headers for Low-Speed Serial Links • IP Header Compression • Compressing IP/UDP/RTP Headers for Low-Speed Serial Links • IP Header Compression over PPP • A Transport Protocol for Real-Time Applications

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Glossary

compression --The running of a data set through an algorithm that reduces the space required to store the data set or the bandwidth required to transmit the data set.

decompression --The act of reconstructing a compressed header.

HDLC --High-Level Data Link Control. A bit-oriented synchronous data link layer protocol developed by International Organization for Standardization (ISO). Derived from Synchronous Data Link Control (SDLC), HDLC specifies a data encapsulation method on synchronous serial links using frame characters and checksums.

header --A chain of subheaders.

incorrect decompression --The circumstance in which a compressed and then decompressed header is different from the uncompressed header. This variance is usually due to a mismatched context between the compressor and decompressor or bit errors during transmission of the compressed header.

ISDN --Integrated Services Digital Network. A communication protocol offered by telephone companies that permits telephone networks to carry data, voice, and other source traffic.

MQC --Modular Quality of Service Command-Line Interface. The MQC allows you to create traffic classes and policy maps and then attach the policy maps to interfaces. The policy maps apply QoS features to your network.

PPP --Point-to-Point Protocol. A protocol that provides router-to-router and host-to-network connections over synchronous and asynchronous circuits.

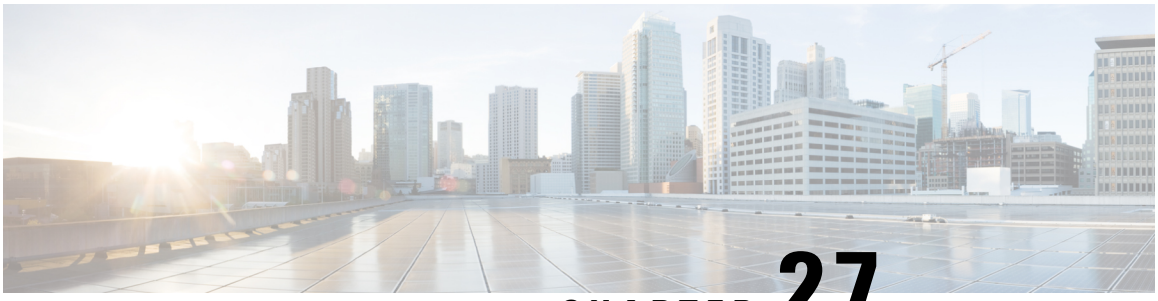
regular header --A normal, uncompressed header. A regular header does not carry a context identifier (CID) or generation association.

RTP --Real-Time Transport Protocol. A protocol that is designed to provide end-to-end network transport functions for applications that transmit real-time data, such as audio, video, or simulation data, over unicast

or multicast network services. RTP provides such services as payload type identification, sequence numbering, timestamping, and delivery monitoring to real-time applications.

subheader --An IPv6 base header, an IPv6 extension header, an IPv4 header, a UDP header, an RTP header, or a TCP header.

UDP --User Datagram Protocol. A connectionless transport layer protocol in the TCP/IP protocol stack. UDP is a simple protocol that exchanges datagrams without acknowledgments or guaranteed delivery, requiring that error processing and retransmission be handled by other protocols. UDP is defined in RFC 768.



CHAPTER 27

Configuring RTP Header Compression

Header compression is a mechanism that compresses the header in a packet before the packet is transmitted. RTP header compression reduces network overhead and speeds up the transmission of Real-Time Transport Protocol (RTP) packets.

Prerequisites for Configuring RTP Header Compression

- Before configuring RTP header compression, read the information in the "Header Compression" module.
- You must configure RTP header compression on both ends of the network.

Information About Configuring RTP Header Compression

Configurable RTP Header-Compression Settings

With RTP header compression, you can configure the maximum size of the compressed header, the maximum time between transmitting full-header packets, and the maximum number of compressed packets between full headers. These settings are configured using the following three commands:

- **ip header-compression max-header**
- **ip header-compression max-time**
- **ip header-compression max-period**

The **ipheader-compressionmax-header** command allows you to define the maximum size of the header of a packet to be compressed. Any packet with an header that exceeds the maximum size is sent uncompressed.

The **ipheader-compressionmax-time**command allows you to specify the maximum time between transmitting full-header packets, and the **ipheader-compressionmax-period** command allows you to specify the maximum number of compressed packets between full headers. With the **ipheader-compressionmax-time** and **ipheader-compressionmax-period** commands, the full-header packet is transmitted at the specified time period or when the maximum number of packets is reached, respectively. The counters for both the time period and the number of packets sent are reset after the full-header packet is sent.

For more information about these commands, see the Cisco IOS Quality of Service Solutions Command Reference.

RTP Header-Compression Keywords

When you configure RTP header compression, you can specify the circumstances under which the RTP packets are compressed and the format that is used when the packets are compressed. These circumstances and formats are defined by the following keywords:

- **passive**
- **iphc-format**
- **ietf-format**
- cisco

These keywords (described below) are available with many of the quality of service (QoS) commands used to configure RTP header compression, such as the **iprtpheader-compression** command. For more information about the **iprtpheader-compression** command, these keywords, and the other QoS commands, see the Cisco IOS Quality of Service Solutions Command Reference.

The passive Keyword

By default, the **iprtpheader-compression** command compresses outgoing RTP traffic. If you specify the **passive** keyword, outgoing RTP traffic is compressed only if *incoming* RTP traffic on the *same* interface is compressed. If you do not specify the **passive** keyword, *all* outgoing RTP traffic is compressed.

The **passive** keyword is ignored on PPP interfaces.

The iphc-format Keyword

The **iphc-format**keyword indicates that the IP Header Compression (IPHC) format of header compression will be used. For PPP and HDLC interfaces, when the **iphc-format** keyword is specified, TCP header compression is also enabled. Since both RTP and TCP header compression are enabled, both UDP and TCP packets are compressed.

The **iphc-format** keyword includes checking whether the destination port number is even and is in the ranges of 16,385 to 32,767 (for Cisco audio) or 49,152 to 65,535 (for Cisco video). Valid RTP packets that meet the criteria (that is, the port number is even and is within the specified range) are compressed using the compressed RTP packet format. Otherwise, packets are compressed using the less-efficient compressed non-TCP packet format.

The**iphc-format**keyword is not available for interfaces that use Frame Relay encapsulation.

Note The header compression format (in this case, IPHC) must be the same at *both* ends of the network. That is, if you specify the **iphc-format** keyword on the local router, you must also specify the **iphc-format** keyword on the remote router.

The ietf-format Keyword

The **ietf-format** keyword indicates that the Internet Engineering Task Force (IETF) format of header compression will be used. For HDLC interfaces, the **ietf-format** keyword compresses both TCP and UDP packets. UPD and TCP packets are compressed separately. For PPP interfaces, when the **ietf-format**keyword is specified, TCP header compression is also enabled. Since both RTP header compression and TCP header compression are enabled, both UDP packets and TCP packets are compressed.

With the **ietf-format** keyword, any even destination port number higher than 1024 can be used. Valid RTP packets that meet the criteria (that is, the port number is even and is higher than 1024) are compressed using the compressed RTP packet format. Otherwise, packets are compressed using the less-efficient compressed non-TCP packet format.

The **ietf-format** keyword is not available for interfaces that use Frame Relay encapsulation.

Note The header compression format (in this case, IETF) must be the same at *both* ends of the network. That is, if you specify the **ietf-format**keyword on the local router, you must also specify the **ietf-format** keyword on the remote router.

The cisco Keyword

The **cisco** keyword indicates that the Cisco-propriety or "original" format of header compression will be used.

RTP header-compression using the cisco format supports even-numbered UDP destination ports in the Cisco audio range of 16384 to 32767 or in the video range of 49152 to 65535.

The **cisco** keyword is only available on interfaces that use Frame Relay or HDLC encapsulation.

How to Configure RTP Header Compression

Enabling RTP Header Compression on an Interface

To enable RTP header compression on an interface, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **encapsulation** *encapsulation-type*
5. **ip address** *ip-address mask* [**secondary**]

6. **ip rtp header-compression** [**passive** | **iphc-format** | **ietf-format**| **cisco**] [**periodic-refresh**]
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [*name-tag*] **Example:** `Router(config)# interface serial0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and the interface number.
Step 4	**encapsulation** *encapsulation-type* **Example:** `Router(config-if)# encapsulation ppp`	Sets the encapsulation method used by the interface. • Enter the encapsulation method.
Step 5	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 209.165.200.225 255.255.255.224`	Sets a primary or secondary IP address for an interface. • Enter the IP address and mask for the associated IP subnet.
Step 6	**ip rtp header-compression** [**passive**	**iphc-format**	**ietf-format**| **cisco**] [**periodic-refresh**] **Example:** `Router(config-if)# ip rtp header-compression`	Enables RTP header compression.
Step 7	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode.

Specifying the Header-Compression Settings

With RTP header compression, you can configure the maximum size of the compressed header, the time period for an automatic resend of full-header packets, and the number of packets transmitted before a new full-header packet is sent.

To specify these header-compression settings, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **ip header-compression max-header** *max-header-size*
5.
6. **ip header-compression max-time** *length-of-time*
7.
8. **ip header-compression max-period** *number-of-packets*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [*name-tag*] **Example:** `Router(config)# interface serial0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and the interface number.
Step 4	**ip header-compression max-header** *max-header-size* **Example:** `Router(config-if)# ip header-compression max-header 100`	Specifies the maximum size of the compressed IP header. • Enter the maximum size of the compressed IP header, in bytes.
Step 5		
Step 6	**ip header-compression max-time** *length-of-time* **Example:** `Router(config-if)# ip header-compression max-time 30`	Specifies the maximum amount of time to wait before the compressed IP header is refreshed. • Enter the amount of time, in seconds.
Step 7		

	Command or Action	Purpose
Step 8	**ip header-compression max-period** *number-of-packets* **Example:** `Router(config-if)# ip header-compression max-period 160`	Specifies the maximum number of compressed packets between full headers. • Enter the maximum number of compressed packets between full headers.
Step 9	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode.

Changing the Number of Header-Compression Connections

For PPP and HDLC interfaces, the default is 16 compression connections.

To change the default number of header-compression connections, perform the following steps.

Implications of Changing the Number of Header-Compression Connections

Each header-compression connection sets up a compression cache entry, so you are in effect specifying the maximum number of cache entries and the size of the cache. Too few cache entries for the specified interface can lead to degraded performance, and too many cache entries can lead to wasted memory. Choose the number of header-compression connections according to the network requirements.

Note **Header-Compression Connections on HDLC Interfaces**

For HDLC interfaces, the number of header-compression connections on *both sides* of the network must match. That is, the number configured for use on the local router must match the number configured for use on the remote router.

Header-Compression Connections on PPP Interfaces

For PPP interfaces, if the header-compression connection numbers on both sides of the network do not match, the number used is "autonegotiated." That is, any mismatch in the number of header-compression connections between the local router and the remote router will be automatically negotiated to the lower of the two numbers. For example, if the local router is configured to use 128 header-compression connections, and the remote router is configured to use 64 header-compression connections, the negotiated number will be 64.

Note This autonegotiation function applies to PPP interfaces *only* . For HDLC interfaces, no au

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **interface** *type number* [*name-tag*]
4. **ip rtp compression-connections** *number*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [*name-tag*] **Example:** `Router(config)# interface serial0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and the interface number.
Step 4	**ip rtp compression-connections** *number* **Example:** `Router(config-if)# ip rtp compression-connections 150`	Specifies the total number of RTP header-compression connections that can exist on an interface. • Enter the number of compression connections. **Note** This command can be used for PPP interfaces and HDLC interfaces.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode.

Displaying Header-Compression Statistics

You can display header-compression statistics, such as the number of packets sent, received, and compressed, by using the **showiprtpheader-compression** command.

To display header-compression statistics, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **show ip rtp header-compression** [*interface-typeinterface-number*]
3. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip rtp header-compression** [*interface-typeinterface-number*] **Example:** `Router# show ip rtp header-compression` **Example:**	Displays RTP header-compression statistics for one or all interfaces.
Step 3	**end** **Example:** `Router# end`	(Optional) Exits privileged EXEC mode.

Configuration Examples for RTP Header Compression

Example Enabling RTP Header Compression on an Interface

In the following example, RTP header compression is enabled on serial interface 0.

```
Router> enable
Router# configure terminal
Router(config)# interface serial0
Router(config-if)# encapsulation ppp
Router(config-if)# ip address 209.165.200.225 255.255.255.224
Router(config-if)# ip rtp header-compression
Router(config-if)# end
```

Example Specifying the Header-Compression Settings

In the following example, the maximum size of the compressed IP header (100 bytes) has been specified by using the **ipheader-compressionmax-header** command.

```
Router> enable

Router# configure terminal

Router(config)# interface serial0

Router(config-if)# ip header-compression max-header 100

Router(config-if)# end
```

Example Changing the Number of Header-Compression Connections

In the following example, the number of header-compression connections has been changed to 150 by using the **ip rtp compression-connections**command.

```
Router> enable

Router# configure terminal

Router(config)# interface serial0

Router(config-if)# ip rtp compression-connections 150

Router(config-if)# end
```

Example Displaying Header-Compression Statistics

You can use the **showiprtpheader-compression**command to display header-compression statistics such as the number of packets received, sent, and compressed. The following is sample output from the **showiprtpheader-compression** command.

```
Router# show ip rtp header-compression
 serial0
RTP/UDP/IP header compression statistics:
  Interface Serial0 (compression on, IETF)
    Rcvd:    1473 total, 1452 compressed, 0 errors, 0 status msgs
             0 dropped, 0 buffer copies, 0 buffer failures
    Sent:    1234 total, 1216 compressed, 0 status msgs, 379 not predicted
             41995 bytes saved, 24755 bytes sent
             2.69 efficiency improvement factor
    Connect: 16 rx slots, 16 tx slots,
             6 misses, 0 collisions, 0 negative cache hits, 13 free contexts
             99% hit ratio, five minute miss rate 0 misses/sec, 0 max
```

Additional References

The following sections provide references related to configuring RTP header compression.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Header compression overview	"Header Compression" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2507	*IP Header Compression*
RFC 2508	*Compressing IP/UDP/RTP Headers for Low-Speed Serial Links*
RFC 3544	*IP Header Compression over PPP*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Configuring RTP Header Compression

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 30: Feature Information for Configuring RTP Header Compression

Feature Name	Releases	Feature Information
Express RTP and TCP Header Compression	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers.
RTP Header Compression	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers.

Glossary

compression --The running of a data set through an algorithm that reduces the space required to store the data set or the bandwidth required to transmit the data set.

context --The state that the compressor uses to compress a header and that the decompressor uses to decompress a header. The context is the uncompressed version of the last header sent and includes other information used to compress and decompress the packet.

context-state packet --A special packet sent from the decompressor to the compressor to communicate a list of (TCP or NON_TCP/RTP) context identifiers (CIDs) for which synchronization has been lost. This packet is sent only over a single link, so it requires no IP header.

DLCI --data-link connection identifier. A value that specifies a permanent virtual circuit (PVC) or switched virtual circuit (SVC) in a Frame Relay network. In the basic Frame Relay specification, DLCIs are locally significant (connected devices might use different values to specify the same connection). In the Local Management Interface (LMI) extended specification, DLCIs are globally significant (DLCIs specify individual end devices).

encapsulation --A method of wrapping data in a particular protocol header. For example, Ethernet data is wrapped in a specific Ethernet header before network transit. Also, when dissimilar networks are bridged, the entire frame from one network is simply placed in the header used by the data link layer protocol of the other network.

full header (header refresh) --An uncompressed header that updates or refreshes the context for a packet stream. It carries a CID that will be used to identify the context. Full headers for non-TCP packet streams also carry the generation of the context that they update or refresh.

HDLC --High-Level Data Link Control. A bit-oriented synchronous data link layer protocol developed by the International Organization for Standardization (ISO). Derived from Synchronous Data Link Control (SDLC), HDLC specifies a data encapsulation method on synchronous serial links using frame characters and checksums.

header --A chain of subheaders.

IETF --Internet Engineering Task Force. A task force that consists of over 80 working groups responsible for developing Internet standards.

IPHC --IP Header Compression. A protocol capable of compressing both TCP and UDP headers.

ISDN --Integrated Services Digital Network. A communication protocol offered by telephone companies that permits telephone networks to carry data, voice, and other source traffic.

lossy serial links --Links in a network that are prone to lose packets.

packet stream --The sequence of packets whose headers are similar and share context. For example, headers in an RTP packet stream have the same source and final destination address and the same port numbers in the RTP header.

PPP --Point-to-Point Protocol. A protocol that provides router-to-router and host-to-network connections over synchronous and asynchronous circuits.

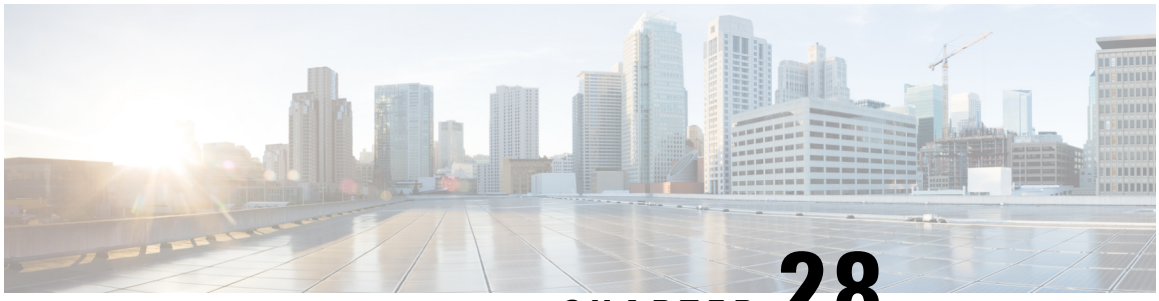
regular header --A normal, uncompressed header. A regular header does not carry a context identifier (CID) or generation association.

RTP --Real-Time Transport Protocol. A protocol that is designed to provide end-to-end network transport functions for applications that transmit real-time data, such as audio, video, or simulation data, over unicast or multicast network services. RTP provides such services as payload type identification, sequence numbering, timestamping, and delivery monitoring to real-time applications.

subheader --An IPv6 base header, an IPv6 extension header, an IPv4 header, a UDP header, an RTP header, or a TCP header.

PART II

Modular QoS



CHAPTER 28

Applying QoS Features Using the MQC

Restrictions for Applying QoS Features Using the MQC

The MQC-based QoS does not support classification of legacy Layer 2 protocol packets such as Internetwork Packet Exchange (IPX), DECnet, or AppleTalk. When these types of packets are being forwarded through a generic Layer 2 tunneling mechanism, the packets can be handled by MQC but without protocol classification. As a result, legacy protocol traffic in a Layer 2 tunnel is matched only by a "match any" class or class-default.

The number of QoS policy maps and class maps supported varies by platform and release.

Note The policy map limitations do not refer to the number of applied policy map instances, only to the definition of the policy maps.

About

The MQC Structure

The MQC (Modular Quality of Service (QoS) Command-Line Interface (CLI)) enables you to set packet classification and marking based on a QoS group value. MQC CLI allows you to create traffic classes and policies, enable a QoS feature (such as packet classification), and attach these policies to interfaces.

The MQC structure necessitates developing the following entities: traffic class, policy map, and service policy.

Elements of a Traffic Class

A traffic class contains three major elements: a traffic class name, a series of **match** commands, and, if more than one **match** command is used in the traffic class, instructions on how to evaluate these **match** commands.

The **match** commands are used for classifying packets. Packets are checked to determine whether they meet the criteria specified in the **match**commands; if a packet meets the specified criteria, that packet is considered a member of the class. Packets that fail to meet the matching criteria are classified as members of the default traffic class.

Available match Commands

The table below lists *some* of the available **match** commands that can be used with the MQC. The available **match** commands vary by Cisco IOS XE release. For more information about the commands and command syntax, see the *Cisco IOS Quality of Service Solutions* Command Reference.

Table 31: match Commands That Can Be Used with the MQC

Command	Purpose
match access-group	Configures the match criteria for a class map on the basis of the specified access control list (ACL).
match any	Configures the match criteria for a class map to be successful match criteria for all packets.
match cos	Matches a packet based on a Layer 2 class of service (CoS) marking.
match destination-address mac	Uses the destination MAC address as a match criterion.
match discard-class	Matches packets of a certain discard class.
match [**ip**] **dscp**	Identifies a specific IP differentiated service code point (DSCP) value as a match criterion. Up to eight DSCP values can be included in one match statement.
match fr-dlci	Specifies the Frame Relay data-link connection identifier (DLCI) number as a match criterion in a class map.
match input-interface	Configures a class map to use the specified input interface as a match criterion.
match ip rtp	Configures a class map to use the Real-Time Transport Protocol (RTP) port as the match criterion.
match mpls experimental	Configures a class map to use the specified value of the Multiprotocol Label Switching (MPLS) experimental (EXP) field as a match criterion.
match mpls experimental topmost	Matches the MPLS EXP value in the topmost label.

Command	Purpose
match not	Specifies the single match criterion value to use as an unsuccessful match criterion. **Note** The **match not** command, rather than identifying the specific match parameter to use as a match criterion, is used to specify a match criterion that prevents a packet from being classified as a member of the class. For instance, if the **match not qos-group 6** command is issued while you configure the traffic class, QoS group 6 becomes the only QoS group value that is not considered a successful match criterion. All other QoS group values would be successful match criteria.
match packet length	Specifies the Layer 3 packet length in the IP header as a match criterion in a class map.
match port-type	Matches traffic on the basis of the port type for a class map.
match [**ip**] **precedence**	Identifies IP precedence values as match criteria.
match protocol	Configures the match criteria for a class map on the basis of the specified protocol. **Note** A separate **match protocol** (NBAR) command is used to configure network-based application recognition (NBAR) to match traffic by a protocol type known to NBAR.
match protocol fasttrack	Configures NBAR to match FastTrack peer-to-peer traffic.
match protocol gnutella	Configures NBAR to match Gnutella peer-to-peer traffic.
match protocol http	Configures NBAR to match Hypertext Transfer Protocol (HTTP) traffic by URL, host, Multipurpose Internet Mail Extension (MIME) type, or fields in HTTP packet headers.
match protocol rtp	Configures NBAR to match RTP traffic.
match qos-group	Identifies a specific QoS group value as a match criterion.
match source-address mac	Uses the source MAC address as a match criterion.

Multiple match Commands in One Traffic Class

If the traffic class contains more than one **match** command, you need to specify how to evaluate the **match** commands. You specify this by using either the **match-any** or **match-all** keyword of the **class-map** command. Note the following points about the **match-any** and **match-all** keywords:

- If you specify the **match-any** keyword, the traffic being evaluated by the traffic class must match *one* of the specified criteria.
- If you specify the **match-all** keyword, the traffic being evaluated by the traffic class must match *all* of the specified criteria.

- If you do not specify either keyword, the traffic being evaluated by the traffic class must match *all* of the specified criteria (that is, the behavior of the **match-all** keyword is used).

Elements of a Traffic Policy

A traffic policy contains three elements: a traffic policy name, a traffic class (specified with the **class** command), and the command used to enable the QoS feature.

The traffic policy (policy map) applies the enabled QoS feature to the traffic class once you attach the policy map to the interface (by using the **service-policy** command).

Note A packet can match only *one* traffic class within a traffic policy. If a packet matches more than one traffic class in the traffic policy, the *first* traffic class defined in the policy will be used.

Commands Used to Enable QoS Features

The commands used to enable QoS features vary by Cisco IOS XE release. The table below lists *some* of the available commands and the QoS features that they enable. For complete command syntax, see the *Cisco IOS QoS Command Reference*.

For more information about a specific QoS feature that you want to enable, see the appropriate module of the Cisco IOS XE Quality of Service Solutions Configuration Guide.

Table 32: Commands Used to Enable QoS Features

Command	Purpose
bandwidth	Configures a minimum bandwidth guarantee for a class.
bandwidth remaining	Configures an excess weight for a class.
fair-queue	Enables the flow-based queueing feature within a traffic class.
fair-queue pre-classify	Configures and checks whether the **qos pre-classify** command can be used for fair queue. When the **qos pre-classify** command is enabled on the tunnel inteface, and then the **fair-queue pre-classify** command is enabled for the policy-map, the policy-map is attached to either the tunnel interface or the physical interface. The inner IP header of the tunnel will be used for the hash algorithm of the fair queue.
drop	Discards the packets in the specified traffic class.
police	Configures traffic policing.
police (percent)	Configures traffic policing on the basis of a percentage of bandwidth available on an interface.
police (two rates)	Configures traffic policing using two rates, the committed information rate (CIR) and the peak information rate (PIR).

Command	Purpose
priority	Gives priority to a class of traffic belonging to a policy map.
queue-limit	Specifies or modifies the maximum number of packets the queue can hold for a class configured in a policy map.
random-detect	Enables Weighted Random Early Detection (WRED).
random-detect discard-class	Configures the WRED parameters for a discard-class value for a class in a policy map.
random-detect discard-class-based	Configures WRED on the basis of the discard class value of a packet.
random-detect exponential-weighting-constant	Configures the exponential weight factor for the average queue size calculation for the queue reserved for a class.
random-detect precedence	Configure the WRED parameters for a particular IP Precedence for a class policy in a policy map.
service-policy	Specifies the name of a traffic policy used as a matching criterion (for nesting traffic policies [hierarchical traffic policies] within one another).
set atm-clp	Sets the cell loss priority (CLP) bit when a policy map is configured.
set cos	Sets the Layer 2 class of service (CoS) value of an outgoing packet.
set discard-class	Marks a packet with a discard-class value.
set [**ip**] **dscp**	Marks a packet by setting the differentiated services code point (DSCP) value in the type of service (ToS) byte.
set fr-de	Changes the discard eligible (DE) bit setting in the address field of a Frame Relay frame to 1 for all traffic leaving an interface.
set mpls experimental	Designates the value to which the MPLS bits are set if the packets match the specified policy map.
set precedence	Sets the precedence value in the packet header.
set qos-group	Sets a QoS group identifier (ID) that can be used later to classify packets.
shape	Shapes traffic to the indicated bit rate according to the algorithm specified.

Nested Traffic Classes

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

In a scenario where packets satisfy more than one match criterion, the MQC enables you to associate multiple traffic classes with a single traffic policy (also termed nested traffic classes) using the **match class-map** command. (We term these *nested class maps* or *MQC Hierarchical class maps*.) This command provides the only method of combining match-any and match-all characteristics within a single traffic class. By doing so, you can create a traffic class using one match criterion evaluation instruction (either match-any or match-all)

and then use that traffic class as a match criterion in a traffic class that uses a different match criterion type. For example, a traffic class created with the match-any instruction must use a class configured with the match-all instruction as a match criterion, or vice versa.

Consider this likely scenario: Suppose A, B, C, and D were all separate match criterion, and you wanted traffic matching A, B, or C and D (i.e., A or B or [C and D]) to be classified as belonging to a traffic class. Without the nested traffic class, traffic would either have to match all four of the match criterion (A and B and C and D) or match any of the match criterion (A or B or C or D) to be considered part of the traffic class. You would not be able to combine "and" (match-all) and "or" (match-any) statements within the traffic class; you would be unable to configure the desired configuration.

The solution: Create one traffic class using match-all for C and D (which we will call criterion E), and then create a new match-any traffic class using A, B, and E. The new traffic class would have the correct evaluation sequence (A or B or E, which is equivalent to A or B or [C and D]).

match-all and match-any Keywords of the class-map Command

One of the commands used when you create a traffic class is the **class-map**command. The command syntax for the **class-map** command includes two keywords: **match-all** and **match-any**. The **match-all** and **match-any** keywords need to be specified only if more than one match criterion is configured in the traffic class. Note the following points about these keywords:

- The **match-all** keyword is used when *all* of the match criteria in the traffic class must be met in order for a packet to be placed in the specified traffic class.
- The **match-any** keyword is used when only *one* of the match criterion in the traffic class must be met in order for a packet to be placed in the specified traffic class.
- If neither the **match-all** keyword nor **match-any** keyword is specified, the traffic class will behave in a manner consistent with the **match-all** keyword.

input and output Keywords of the service-policy Command

As a general rule, the QoS features configured in the traffic policy can be applied to packets entering the interface or to packets leaving the interface. Therefore, when you use the **service-policy** command, you need to specify the direction of the traffic policy by using the **input** or **output** keyword.

For instance, the **service-policy output policy-map1** command would apply the QoS features in the traffic policy to the interface in the output direction. All packets leaving the interface (output) are evaluated according to the criteria specified in the traffic policy named policy-map1.

Note For Cisco releases, queueing mechanisms are not supported in the input direction. Nonqueueing mechanisms (such as traffic policing and traffic marking) are supported in the input direction. Also, classifying traffic on the basis of the source MAC address (using the **match source-address mac** command) is supported in the input direction only.

Benefits of Applying QoS Features Using the MQC

The MQC structure allows you to create the traffic policy (policy map) once and then apply it to as many traffic classes as needed. You can also attach the traffic policies to as many interfaces as needed.

How to Apply QoS Features Using the MQC

Creating a Traffic Class

To create a traffic class, use the **class-map** command to specify the traffic class name. Then use one or more **match** commands to specify the appropriate match criteria. Packets matching the criteria that you specify are placed in the traffic class. For more information about the **match-all** and **match-any** keywords of the class-map comand, see the "match-all and match-any Keywords of the class-map Command" section.

Note The **match cos** command is shown in Step 4. The **match cos** command is simply an example of one of the **match** commands that you can use. For information about the other available **match** commands, see the "match-all and match-any Keywords of the class-map Command" section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** [**match-all** | **match-any**] *class-map-name*
4. **match cos** *cos-number*
5. Enter additional match commands, if applicable; otherwise, continue with step 6.
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** [**match-all**	**match-any**] *class-map-name* **Example:** `Router(config)# class-map match-any class1`	Creates a class to be used with a class map and enters class-map configuration mode. • The class map is used for matching packets to the specified class.

	Command or Action	Purpose
		• Enter the class name. **Note** The **match-all** keyword specifies that all match criteria must be met. The **match-any** keyword specifies that one of the match criterion must be met. Use these keywords only if you will be specifying more than one **match** command.
Step 4	**match cos** *cos-number* **Example:** `Router(config-cmap)# match cos 2`	Matches a packet on the basis of a Layer 2 class of service (CoS) number. • Enter the CoS number. **Note** The **match cos** command is an example of the **match** commands you can use. For information about the other **match** commands that are available, see the "match-all and match-any Keywords of the class-map Command" section.
Step 5	Enter additional match commands, if applicable; otherwise, continue with step 6.	--
Step 6	**end** **Example:** `Router(config-cmap)# end`	(Optional) Exits QoS class-map configuration mode and returns to privileged EXEC mode.

Creating a Traffic Policy

Note The **bandwidth** command is shown in Step 5. The **bandwidth** command is an example of the commands that you can use in a policy map to enable a QoS feature (in this case, Class-based Weighted Fair Queuing (CBWFQ). For information about other available commands, see the "Elements of a Traffic Policy" section.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **bandwidth** {*bandwidth-kbps* | **percent** *percent*}
6. Enter the commands for any additional QoS feature that you want to enable, if applicable; otherwise, continue with Step 7.
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy1`	Creates or specifies the name of the traffic policy and enters QoS policy-map configuration mode. • Enter the policy map name.
Step 4	**class** {*class-name* | **class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the name of a traffic class and enters QoS policy-map class configuration mode. **Note** This step associates the traffic class with the traffic policy.
Step 5	**bandwidth** {*bandwidth-kbps* | **percent** *percent*} **Example:** `Router(config-pmap-c)# bandwidth 3000`	(Optional) Specifies a minimum bandwidth guarantee to a traffic class in periods of congestion. • A minimum bandwidth guarantee can be specified in kb/s or by a percentage of the overall available bandwidth. **Note** The **bandwidth** command enables CBWFQ. The **bandwidth** command is an example of the commands that you can use in a policy map to enable a QoS feature. For information about the other commands available, see the "Elements of a Traffic Policy" section.
Step 6	Enter the commands for any additional QoS feature that you want to enable, if applicable; otherwise, continue with Step 7.	--
Step 7	**end** **Example:** `Router(config-pmap-c)# end`	(Optional) Exits QoS policy-map class configuration mode and returns to privileged EXEC mode.

Attaching a Traffic Policy to an Interface Using the MQC

Note Cisco IOS XE Release 2.3.0 and later releases do not support the attachment of policies for ATM interfaces that have unspecified bit rate (UBR) configured as the default mode on their VC or virtual path (VP). An attempt to use this configuration results in an error message: CBWFQ: Not supported on ATM interfaces with UBR configuration. You can also specify UBR with a rate in the UBR configuration, if you do not want to use the default UBR value.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **service-policy** {**input** | **output**} *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# interface serial 0/0/1`	Configures an interface type and enters interface configuration mode. • Enter the interface type and interface number.
Step 4	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# service-policy input policy1`	Attaches a policy map to an interface. • Enter either the **input** or **output** keyword and the policy map name.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode and returns to privileged EXEC mode.

Verifying the Traffic Class and Traffic Policy Information

The show commands described in this section are optional and can be entered in any order.

SUMMARY STEPS

1. **enable**
2. **show class-map**
3. **show policy-map** *policy-map-name* **class** *class-name*
4. **show policy-map**
5. **show policy-map interface** *type number*
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show class-map** **Example:** `Router# show class-map`	(Optional) Displays all class maps and their matching criteria.

Step 3	**show policy-map** *policy-map-name* **class** *class-name*
	Example:
	`Router#`
	`show policy-map policy1 class class1`
		(Optional) Displays the configuration for the specified class of the specified policy map.
		• Enter the policy map name and the class name.

Step 4	**show policy-map** **Example:** `Router# show policy-map`	(Optional) Displays the configuration of all classes for all existing policy maps.
Step 5	**show policy-map interface** *type number* **Example:** `Router# show policy-map interface serial 0/0/1`	(Optional) Displays the statistics and the configurations of the input and output policies that are attached to an interface. • Enter the interface type and number.
Step 6	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Configuration Examples for Applying QoS Features Using the MQC

Creating a Traffic Class

In the following example, we create traffic classes and define their match criteria. For the first traffic class (class1), we use access control list (ACL) 101 as match criteria; for the second traffic class (class2), ACL 102. We check the packets 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
  end
```

Creating a Traffic Class

In the following example, we create traffic classes and define their match criteria. For the first traffic class (class1), we use access control list (ACL) 101 as match criteria; for the second traffic class (class2), ACL 102. We check the packets 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
  end
```

Example: Attaching a Traffic Policy to an Interface

The following example shows how to attach an existing traffic policy to an interface. After you define a traffic policy with the **policy-map** command, you can attach it to one or more 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 in the input direction and only one traffic policy attached in the output direction.

```
Router(config)# interface fastethernet 1/1/1
Router(config-if)# service-policy output policy1
Router(config-if)# exit
Router(config)# interface fastethernet 1/0/0
Router(config-if)# service-policy output policy1
Router(config-if)# end
```

Using the match not Command

Use the **match not** command to specify a QoS policy value that is not used as a match criterion. All other values of that QoS policy become successful match criteria. For instance, if you issue the **match not qos-group**

4 command in QoS class-map configuration mode, the specified class will accept all QoS group values except 4 as successful match criteria.

In the following traffic class, all protocols except IP are considered successful match criteria:

```
class-map noip
  match not protocol ip
  end
```

Configuring a Default Traffic Class

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

If you do not configure a default class, packets are still treated as members of that class. The default class has no QoS features enabled so packets belonging to this class have no QoS functionality. Such packets are placed into a first-in, first-out (FIFO) queue managed by tail drop, which is a means of avoiding congestion that treats all traffic equally and does not differentiate between classes of service. Queues fill during periods of congestion. When the output queue is full and tail drop is active, packets are dropped until the congestion is eliminated and the queue is no longer full.

The following example configures a policy map (policy1) for the default class (always called class-default) with these characteristics: 10 queues for traffic that does not meet the match criteria of other classes whose policy is defined by class policy1, and a maximum of 20 packets per queue before tail drop is enacted to handle additional queued packets.

In the following example, we configure a policy map (policy1) for the default class (always termed class-default) with these characteristics: 10 queues for traffic that does not meet the match criterion of other classes whose policy is defined by the traffic policy policy1.

```
policy-map policy1
  class class-default
    shape average 100m
```

How "fair-queue" Supports "pre-classify" Command

Prior to the Cisco IOS 16.4 release, when you configure fair-queue on the tunnel interface, the outer IP header of the tunnel was used for the hash algorithm of fair queue. Therefore, the packets of all flows on the tunnel were put into the same flow queue. This is the behavior seen even when the **qos pre-classify** command is configured on the tunnel interface

From the Cisco IOS 16.4 release onwards, **fair-queue** supports the **pre-classify** command. This command is added so that **qos pre-classify** can be used with the **fair-queue** command.

The following example configures **fair-queue pre-classify** command for policy-map under class configuration:

```
interface tunnel 0
    qos pre-classify
policy-map po1
    class c1
        shape average percentage 10
        fair-queue pre-classify
```

When **qos pre-classify** is enabled on the tunnel interface, and the **fair-queue pre-classify** is enabled for policy-map, then the policy-map is attached to either the tunnel interface or the physical interface. The inner IP header of the tunnel is used for the hash algorithm of the fair queue.

To disable this feature, use the **fair-queue** command without the **pre-classify** keyword.

The default behavior of fair queue remains unchanged.

How Commands "class-map match-any" and "class-map match-all" Differ

This example shows how packets are evaluated when multiple match criteria exist. It illustrates the difference between the **class-map match-any** and **class-map match-all** commands. Packets must meet either all of the match criteria (**match-all**) or one of the match criteria (**match-any**) to be considered a member of the traffic class.

The following examples show a traffic class configured with the **class-map match-all** command:

```
class-map match-all cisco1
  match qos-group 4
  match access-group 101
```

If a packet arrives on a router with traffic class cisco1 configured on the interface, we assess whether it matches the IP protocol, QoS group 4, and access group 101. If all of these match criteria are met, the packet is classified as a member of the traffic class cisco1 (a logical AND operator; Protocol IP AND QoS group 4 AND access group 101).

```
class-map match-all vlan
  match vlan 1
  match vlan inner 1
```

The following example illustrates use of the **class-map match-any** command. Only one match criterion must be met for us to classify the packet as a member of the traffic class (i.e., a logical OR operator; protocol IP OR QoS group 4 OR access group 101):

```
class-map match-any cisco2
  match protocol ip
  match qos-group 4
  match access-group 101
```

In the traffic class cisco2, the match criterion are evaluated consecutively until a successful match is located. The packet is first evaluated to determine whether the IP protocol can be used as a match criterion. If so, the packet is matched to traffic class cisco2. If not, then QoS group 4 is evaluated as a match criterion and so on. If the packet matches none of the specified criteria, the packet is classified as a member of the default traffic class (*class default-class*).

Establishing Traffic Class as a Match Criterion (Nested Traffic Classes)

There are two reasons to use the **match class-map** command. One reason is maintenance; if a large traffic class currently exists, using the traffic class match criterion is easier than retyping the same traffic class configuration. The second and more common reason is to mix match-all and match-any characteristics in one traffic policy. This enables you to create a traffic class using one match criterion evaluation instruction (either match-any or match-all) and then use that traffic class as a match criterion in a traffic class that uses a different match criterion type.

Consider this likely scenario: Suppose A, B, C, and D were all separate match criterion, and you wanted traffic matching A, B, or C and D (i.e., A or B or [C and D]) to be classified as belonging to a traffic class. Without the nested traffic class, traffic would either have to match all four of the match criterion (A and B and C and D) or match any of the match criterion (A or B or C or D) to be considered part of the traffic class. You would not be able to combine "and" (match-all) and "or" (match-any) statements within the traffic class; you would be unable to configure the desired configuration.

The solution: Create one traffic class using match-all for C and D (which we will call criterion E), and then create a new match-any traffic class using A, B, and E. The new traffic class would have the correct evaluation sequence (A or B or E, which is equivalent to A or B or [C and D]).

Example: Nested Traffic Class for Maintenance

In the following example, the traffic class class1 has the same characteristics as the traffic class class2, with the exception that the former has added a destination address as a match criterion. Rather than configuring traffic class class1 line by line, you can enter the **match class-map class2** command. This command allows you to include all of the characteristics in the traffic class called class2 in the traffic class class1, and you can add the new destination address match criterion without reconfiguring the entire traffic class.

```
Router(config)# class-map match-any class2
Router(config-cmap)# match protocol ip
Router(config-cmap)# match qos-group 3
Router(config-cmap)# match access-group 2
Router(config-cmap)# exit
Router(config)# class-map match-all class1
Router(config-cmap)# match class-map class2
Router(config-cmap)# match destination-address mac 00.00.00.00.00.00
Router(config-cmap)# exit
```

Example: Nested Traffic Class to Combine match-any and match-all Characteristics in One Traffic Class

The only method of including both match-any and match-all characteristics in a single traffic class is to use the **match class-map** command. To combine match-any and match-all characteristics into a single class, use the match-any instruction to create a traffic class that uses a class configured with the match-all instruction as a match criterion (through the **match class-map** command).

The following example shows how to combine the characteristics of two traffic classes, one with match-any and one with match-all characteristics, into one traffic class with the **match class-map** command. The result requires a packet to match one of the following three match criteria to be considered a member of traffic class class4: IP protocol *and* QoS group 4, destination MAC address 00.00.00.00.00.00, or access group 2.

In this example, only the traffic class called class4 is used with the traffic policy called policy1.

```
Router(config)# class-map match-all class3
Router(config-cmap)# match protocol ip
Router(config-cmap)# match qos-group 4
Router(config-cmap)# exit
Router(config)# class-map match-any class4
Router(config-cmap)# match class-map class3
Router(config-cmap)# match destination-address mac 00.00.00.00.00.00
Router(config-cmap)# match access-group 2
Router(config-cmap)# exit
Router(config)# policy-map policy1
Router(config-pmap)# class class4
Router(config-pmap-c)# police 8100 1500 2504 conform-action transmit exceed-action
```

```
set-qos-transmit 4
Router(config-pmap-c)# end
```

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

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

A hierarchical traffic policy contains a child policy 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 preexisting traffic policy is the parent policy. In the example in this section, the traffic policy called child is the child policy and traffic policy called parent is the parent policy.

Hierarchical traffic policies can be attached to subinterfaces and ATM PVCs. When hierarchical traffic policies are used, a single traffic policy (with a child and a parent policy) can be used to shape and prioritize permanent virtual connection (PVC) traffic. 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.

```
Router(config)# policy-map child
Router(config-pmap)# class voice
Router(config-pmap-c)# priority ?
384-100000000 Kilo Bits per second
level Multi-Level Priority Queue
percent % of total bandwidth
Router(config-pmap-c)# priority 50
Router(config)# policy-map parent
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# service-policy child
```

The value used with the **shape** command is provisioned from the committed information rate (CIR) value from the service provider.

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Packet classification	“Classifying Network Traffic” module
Frame Relay Fragmentation (FRF) PVCs	“FRF .20 Support” module
Selective Packet Discard	“IPv6 Selective Packet Discard” module

Related Topic	Document Title
Scaling and performance information	"Broadband Scalability and Performance" module of the Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide .

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Applying QoS Features Using the MQC

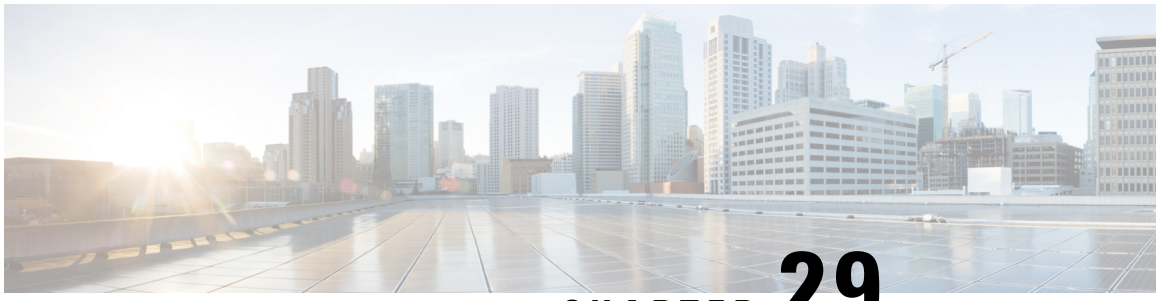
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 33: Feature Information for Applying QoS Features Using the MQC

Feature Name	Releases	Feature Information
Class-Based Weighted Fair Queueing (CBWFQ)	Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.5S	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers. In Cisco IOS XE Release 3.5S, support was added for the Cisco ASR 903 Router.

Feature Name	Releases	Feature Information
Modular QoS CLI (MQC)	Cisco IOS XE Release 2.1 Cisco IOS XE Release 3.5S	This module describes how to apply and configure quality of service (QoS) features using the modular QoS CLI (MQC). The MQC allows you to define a traffic class, create a traffic policy (policy map), and attach the traffic policy to an interface. The traffic policy contains the QoS feature that will be applied to the traffic class. This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers. This feature was enhanced to provide infrastructure support for additional features included with Cisco IOS XE Release 2.3. In Cisco IOS XE Release 3.5S, support was added for the Cisco ASR 903 router.
MQC Hierarchical Class Map	Cisco IOS XE Release 3.2	MQC allows multiple traffic classes (nested traffic classes, which are also called nested class maps or MQC hierarchical class maps) to be configured as a single traffic class. This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers.
Priority Queueing	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers.
Weighted Random Early Detection	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers.



CHAPTER 29

3-Level User-Defined Queuing Policy Support

3-level user-defined queuing policy support feature allows 3 level policy with topmost layer user defined classes to support and enhance the flexibility of the traffic class in the hierarchy.

Restrictions for 3-Level User-Defined Queuing Policy Support

- User-defined class in top layer of a 3-level hierarchical queuing policy is not supported on port-channel main interface.

  User-defined class at the topmost layer is not supported on any logical target. Logical targets include service-group, tunnel, session, dealer interface, etc.

Information About 3-Level User-Defined Queuing Policy Support

Three-Parameter Scheduler in Hierarchical QoS

Classic IOS uses max value (shape) and min value (bandwidth) to define each scheduler node behavior when traffic congestion happens, or 2 parameter scheduler.

ASR 1000 utilize a different 3-parameter scheduler: max value (shape), min value (bandwidth) and excess value (bandwidth remaining) which is to adjust sharing of excess bandwidth. In a 2-parameter scheduler, the excess bandwidth are shared by the classes proportionally (same as the bandwidth ratio for each class); But in a 3-parameter scheduler, the excess bandwidth are shared equally by default after satisfying minimum bandwidth requirements, but it can be tuned when using ‘bandwidth remaining’ command. ISR 4000 platforms share the same design.

In Classic IOS, it is permitted to configure bandwidth at the leaf and intermediate nodes of a hierarchy. In IOS XE, bandwidth (bandwidth rate , or bandwidth percent) is only allowed at the leaf node of a hierarchy. In other words, bandwidth (bandwidth rate , or bandwidth percent) class cannot be attached with a child policy-map containing queuing features. This is a restriction in software and may be lifted in the future.

For current deployments where a Classic IOS QoS policy-map is being moved to a IOS XE platform, the best option is to convert the intermediate node bandwidth commands to bandwidth remaining commands. bandwidth remaining percent or bandwidth remaining ratio commands could be used to achieve very similar behavior.

Guidelines for Hierarchical Policies

In general, three levels of hierarchy are supported on ASR 1000. Hierarchical policy can be applied on most of the physical and logical targets that supports QoS.

If you mix queuing and non-queuing policies together in a hierarchy, the non-queuing policy-maps must be at the leaf level of the policy-map (for example, child policy beneath grandparent and parent queuing policies).

If the policy-map is applied to a virtual interface (such as a tunnel or session), there may be additional restrictions limiting the hierarchy to two levels of queuing, depending on the configuration.

- Queuing features: shape, bandwidth, bandwidth remaining, random-detect, fair-queue, queue limit, and priority.
- Non-queuing features: police, mark, and account.

Hierarchy Feature Combination	Ingress Policy Support	Egress Policy Support
One-level Non-queuing Policy	Yes	Yes
Two-level Non-queuing Policy (including color-aware police)	Yes	Yes
Three-level Non-queuing Policy (including hierarchical color-aware police)	Yes	Yes
One-level Queuing Policy	-	Yes
Two-level Queuing Policy	-	Yes
Three-level Queuing Policy	-	Yes
Two-level Mixed Policy, Queuing feature at parent level	-	Yes
Three-level Mixed Policy, Queuing feature at grandparent level, or at grandparent + parent level	-	Yes

User-defined Traffic Class in Top-level Policy of HQoS

Any traffic class configured explicitly by 'class-map' is called 'user-defined class'. Class-default classes need not be configured, and can be used in any policy to match all the traffic that does not belong to user-defined classes.

In a three-level queuing policy-map, only class-default class can be configured in the highest level before Release Polaris 16.3. From Polaris 16.3, user-defined class in top layer of a 3-level hierarchical policy is supported.

How to Configure 3-Level User-Defined Queuing Policy Support

Configuring 3-level Hierarchical QoS Policy

```
enable
configure terminal
class-map vlan10
 match vlan10
class-map vlan20
 match vlan 20
class-map ef
 match dscp ef
policy-map child
 class ef
  priority
  police 1000000
 class class-default
 police 3000000
policy-map parent
 class vlan10
  shape average 4000000
  service-policy child
 class vlan20
  shape average 8000000
  service-policy child
policy-map grand-parent
 class class-default
  shape average 10000000
  service-policy parent
end
```

Configuring User-Defined Traffic Class in Top Level Policy

```
ip access-list extended PEER
permit ip host 200.0.0.2 any

class-map match-all ef
 match dscp ef
class-map match-all vlan100
 match vlan 100
class-map match-all vlan101
 match vlan 101
class-map match-all PEER
 match access-group name PEER

policy-map child
 class ef
  bandwidth remaining percent 15
 class class-default
  fair-queue
  queue-limit 512 packets
  bandwidth remaining percent 85

policy-map parent
 class PEER
  shape average 8000000
  bandwidth remaining percent 10
```

```
  service-policy child
class class-default
 shape average 8000000

policy-map grandparent
 class vlan100
  shape average 8000000
  bandwidth remaining ratio 1000
  service-policy parent
 class vlan101
  shape average 8000000
  bandwidth remaining ratio 1000
  service-policy parent
 class class-default
  bandwidth remaining ratio 1
  shape average 10000000
end
```

Additional References for 3-Level User-Defined Queuing Policy Support

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

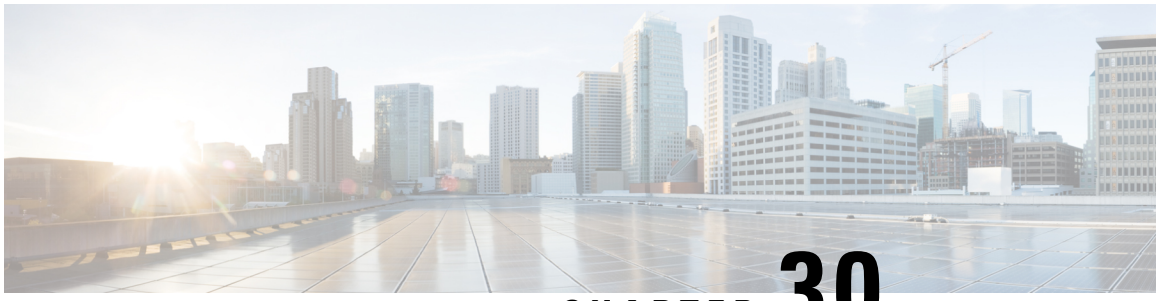
Feature Information for 3-Level User-Defined Queuing Policy Support

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 34: Feature Information for 3-Level User-Defined Queuing Policy Support

Feature Name	Releases	Feature Information
3-Level User-Defined Queuing Policy Support	Cisco IOS XE Denali 16.3.1.	This feature is introduced on Cisco ASR 1000, ISR4000, CSR1000v platforms. User-defined class can be configured in top layer of a 3-level hierarchical policy.



CHAPTER 30

Complex Hierarchical Scheduling: Fragmented Policies (i.e, Policies Aggregation)

The QoS: Policies Aggregation feature supports Modular QoS CLI (MQC) configuration of default traffic classes in policy maps on different subinterfaces to be queued as a single, user-defined traffic class at the main-interface policy map. It is most useful in quality of service (QoS) configurations where you have several subinterface policy maps on the same physical interface and you want identical treatment of the default traffic classes on those subinterfaces.

Beginning in Cisco IOS XE Release 2.6, the QoS: Policies Aggregation feature is enhanced to support queueing aggregation at the primary interface for other traffic classes, including Differentiated Services Code Point (DSCP) traffic classes such as the expedited forwarding (EF), Assured Forwarding 1 (AF1), and AF4 traffic classes. With this enhancement, any traffic classes from VLAN subinterfaces can share a common queue for that traffic class at the main-interface policy map. Other enhancements include the ability to configure and show drop statistics that occur at the aggregate level for these classes.

Prerequisites for QoS: Policies Aggregation

- This feature is configured using the MQC.
- All traffic over the main interface should come through one or more subinterfaces.

Restrictions for QoS: Policies Aggregation

- Applies only when multiple subinterfaces with policy maps are attached to the same physical interface. This feature cannot be used to collectively classify default traffic classes or other traffic classes of policy maps on different physical interfaces.

- Certain traffic class configuration prior to Cisco IOS XE Release 2.6 at the subinterface policy map and main-interface policy map will have different behavior and queueing results. See the "Understanding the QoS Policies Aggregation MQC" section on page 3 and the "Differences Between the Original Feature and the MQC Support for Multiple Queue Aggregation" section on page 4.
- The **service-fragment** keyword is only supported on the Gigabit Ethernet interfaces and not on Fast Ethernet interfaces.

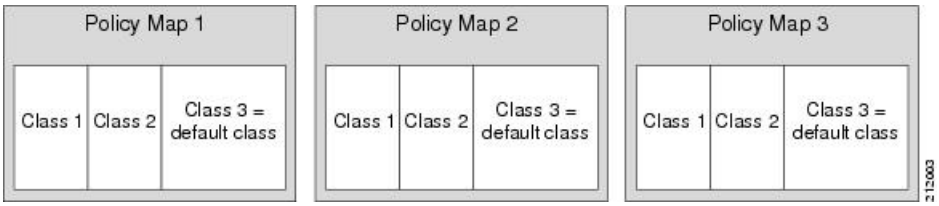
About QoS: Policies Aggregation

Fragments in Class Definition Statements

QoS: Policies Aggregation introduces the idea of fragments in class definition statements. A default traffic class definition statement can be marked as a fragment within a policy map. Other policy maps on the same interface can also define their default traffic class statements as fragments, if desired. A separate policy map can then be created with a service-fragment class definition statement that will be used to apply QoS to all of the fragments as a single group.

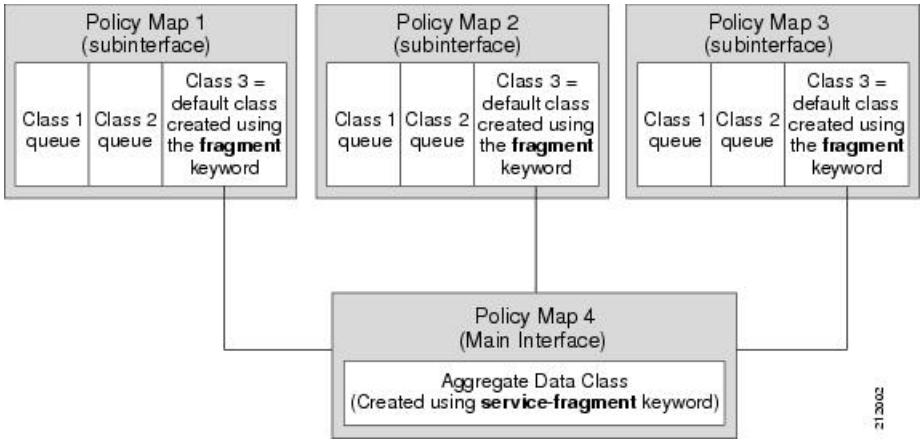
The figure below provides an example of one physical interface with three attached policy maps that is not using fragments. Note that each policy map has a default traffic class that can only classify traffic for the default traffic within its own policy map.

Figure 10: Three Policy Maps Configured Without Fragments



The figure below shows the same configuration configured with fragments and adds a fourth policy map with a class definition statement that classifies the fragments collectively. The default traffic classes are now classified as one service-fragment group rather than three separate default traffic classes within the individual policy maps.

Figure 11: Three Policy Maps Configured Using Fragments



Fragments for Gigabit Etherchannel Bundles

When fragments are configured for Gigabit Etherchannel bundles, the policy-maps that have a default traffic class configured using the **fragment** keyword are attached to the member subinterface links, and the policy-maps that have a traffic class configured with the **service-fragment** keyword to collectively classify the fragments is attached to the physical interface.

All port-channel subinterfaces configured with fragments that are currently active on a given port-channel member link will use the aggregate service fragment class on that member link. If a member link goes down, the port-channel subinterfaces that must switch to the secondary member link will then use the aggregate service fragment on the new interface.

Fragment Traffic Class in a Policy Map

Only the default class statement in a policy map can be configured as a fragment.

Fragments work only when multiple policy maps are attached to the same physical interface. This process cannot be used to classify default traffic classes as fragments on policy maps on different physical interfaces.

Only queuing features are allowed in classes where the **fragment** keyword is entered, and at least one queuing feature must be entered in classes where the **fragment** keyword is used.

A policy map with a class using the **fragment** keyword can only be applied to traffic leaving the interface (policy maps attached to interfaces using the **service-policy output** command).

The **fragment** keyword cannot be entered in a child policy map.

Understanding Service Fragment Traffic Classes

A service fragment can be used to collectively classify fragments only from the same physical interface. Fragments from different interfaces cannot be classified using the same service fragment.

Only queueing features are allowed in classes where the **service-fragment** keyword is entered, and at least one queueing feature must be entered in classes when the **service-fragment** keyword is used.

A policy map with a class using the **service-fragment** keyword can be applied only to traffic leaving the interface (policy maps attached to interfaces using the **service-policy output** command).

A class configured using the **service-fragment** keyword cannot be removed when it is being used to collectively apply QoS to fragments that are still configured on the interface. If you wish to remove a class configured using the **service-fragment** keyword, remove the fragment traffic classes before removing the service fragment.

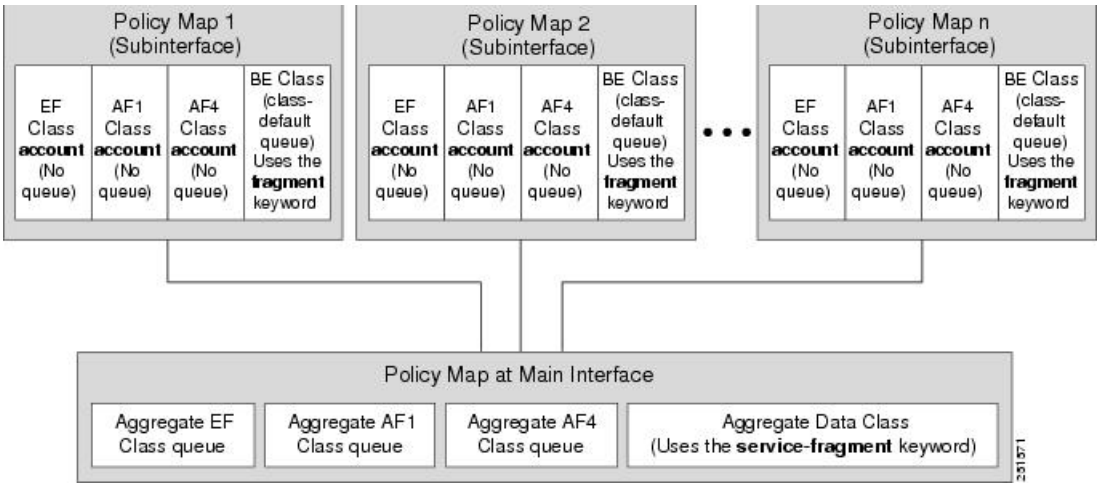
The **service-fragment** keyword cannot be entered in a child policy map.

QoS: Policies Aggregation MQC

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature extends the previous support of aggregation of class-default traffic using the **fragment** and **service-fragment** configurations, to other user-defined traffic classes in a subinterface policy-map, such as DSCP-based traffic classes, that are aggregated at the main-interface policy-map as shown in the figure below.

When no queueing is configured on a traffic class in the subinterface policy-map, the **account** command can be used to track queueing drops that occur at the aggregate level for these classes, and can be displayed using the **show policy-map interface** command.

Figure 12: Policy-Map Overview for the MQC Support for Multiple Queue Aggregation at Main Interface Feature



Differences Between the Original Feature and the MQC Support for Multiple Queue Aggregation

Although some of the configuration between the original QoS policies aggregation feature and enhancements in the MQC Support for Multiple Queue Aggregation at Main Interface feature appears similar, there are some important differences in the queueing behavior and the internal data handling.

For example, both configurations share and require the use of the **fragment** keyword for the **class class-default** command in the subscriber policy map, as well as configuration of the **service-fragment** keyword for a user-defined class in the main-interface policy map to achieve common policy treatment for aggregate traffic. However, the use of this configuration results in different behavior between the original and enhanced QoS policies aggregation implementation:

- In the original implementation (prior to Cisco IOS XE Release 2.6) using the fragment and service-fragment architecture, all default class traffic and any traffic for classes without defined queueing features at the subinterface goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy map. Subinterface traffic aggregation (for example, from

multiple subscribers on the same physical interface) ultimately occurs only for a single class, which is the default class.

Here are the feature characteristics:

- All subinterface traffic classes have queues. However, when a traffic class in the subinterface policy-map is not configured with any queueing feature (commands such as **priority**, **shape**, **bandwidth**, **queue-limit**, **fair-queue**, **random-detect**, and so on, are not configured), the traffic is assigned to the class-default queue.
- Default class traffic from multiple subinterfaces can be aggregated into a common policy map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main-interface class.
- No classification occurs or is supported at the main-interface policy map for any subinterface traffic classes that do not use the **fragment** and **service-fragment** configuration.
- Queueing occurs at the subinterface for other traffic classes defined with queueing features in the subinterface policy map.

- In the enhanced implementation (beginning with Cisco IOS XE Release 2.6) of the MQC Support for Multiple Queue Aggregation at Main Interface feature also using the fragment and service-fragment architecture, all default class traffic also goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy map. However, other classes, such as DSCP-based subscriber traffic classes, are also supported for an aggregate policy. These traffic classes do not support any queues or queueing features other than **account** at the subscriber policy map. The use of the fragment and service-fragment architecture enables these other subscriber traffic classes (from multiple subscribers on the same physical interface) to achieve common policy treatment for aggregate traffic that is defined for those same classes at the main policy map.

  Here are the feature characteristics:

  - Subinterface traffic classes without configured queueing features do not have queues at the subscriber level.
  - Default class traffic from multiple subinterfaces can be aggregated into a common policy map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main-interface class. This configuration additionally enables support for other subinterface traffic classes (such as DSCP-based classes) to be aggregated into a common policy-map at the main interface.
  - Other class traffic from multiple subinterfaces can be aggregated into a common policy map at the main interface, according to the following configuration requirements:
  - You enable this behavior by using the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main-interface class (this also enables aggregation of the default class).
  - You do not configure any queueing features at the subinterface policy-map for the other traffic classes.
  - Queueing occurs at the main-interface policy map for other subinterface traffic classes as an aggregate.
  - Optional tracking of statistics is supported using the **account** command for other traffic classes in the subinterface policy map.

Changes in Queue Limit and WRED Thresholds

In Cisco IOS XE Release 2.6 the Cisco ASR 1000 Series Routers support the addition of bytes as a unit of configuration for both queue limits and WRED thresholds. Therefore, as of this release, packet-based and byte-based limits are configurable, with some restrictions.

Configuration Examples for QoS: Policies Aggregation

Examples 1: Configuring QoS: Policies Aggregation for an Interface

Configuring a Fragment Traffic Class in a Policy-Map

Before you begin

This procedure shows only how to configure the default traffic class as a fragment within a policy-map. It does not include steps on configuring other classes within the policy-map, or other policy-maps on the device.

Example

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a fragment named BestEffort is created in policy-map subscriber1 and policy-map subscriber 2. In this example, queuing features for other traffic classes are supported at the subinterface policy-map.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example also shows how to configure a fragment named BestEffort for the default class in a policy-map on a subinterface using the QoS Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface implementation. In this example, notice that queuing features are not supported for the other classes in the policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

After configuring default class statements as fragments in multiple subinterface policy-maps, a separate policy-map with a class statement using the service-fragment keyword must be configured to apply QoS to the class stratements configured as fragments.

What to Do Next

After configuring default class statements as fragments in multiple subinterface policy maps, a separate policy map with a class statement using the **service-fragment** keyword must be configured to apply QoS to the class statements configured as fragments.

This task is documented in the "Configuring a Service Fragment Traffic Class" section on page 8.

Configuring a Service Fragment Traffic Class

Before you begin

This task describes how to configure a service fragment traffic class statement within a policy-map. A service fragment traffic class is used to apply QoS to a collection of default class statements that have been configured previously in other policy-maps as fragments.

This procedure assumes that fragment default traffic classes were already created. The procedure for creating fragment default traffic classes is documented in the “Configuring a Fragment Traffic Class in a Policy-Map” section.

Like any policy-map, the configuration does not manage network traffic until it has been attached to an interface. This procedure does not cover the process of attaching a policy-map to an interface.

Note A service fragment can be used to collectively classify fragments only from the same physical interface. Fragments from different interfaces cannot be classified using the same service fragment.

Only queueing features are allowed in classes where the **service-fragment** keyword is entered, and at least one queueing feature must be entered in classes when the **service-fragment** keyword is used.

A policy-map with a class using the **service-fragment** keyword can be applied only to traffic leaving the interface (policy-maps attached to interfaces using the **service-policy output** command).

A class configured using the **service-fragment** keyword cannot be removed when it is being used to collectively apply QoS to fragments that are still configured on the interface. If you wish to remove a class configured using the **service-fragment** keyword, remove the fragment traffic classes before removing the service fragment.

The **service-fragment** keyword cannot be entered in a child policy-map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name* **service-fragment** *fragment-class-name*
5. **shape average percent** *percent*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map BestEffortFragments`	Specifies the name of the traffic policy to configure and enters policy-map configuration mode.
Step 4	**class** *class-name* **service-fragment** *fragment-class-name* **Example:** `Device(config-pmap)# class data service-fragment BestEffort`	Specifies a class of traffic that is the composite of all fragments matching the *fragment-class-name*. The *fragment-class-name* when defining the fragments in other policy-maps must match the *fragment-class-name* in this command line to properly configure the service fragment class.

	Command or Action	Purpose
Step 5	**shape average percent** *percent* **Example:** `Device(config-pmap-c)# shape average percent 50`	Enters a QoS configuration command. Only queueing features are supported in default traffic classes configured as fragments. The queueing features that are supported are **bandwidth**, **shape**, and **random-detect exponential-weighting-constant**. Multiple QoS queueing commands can be entered.
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	Exits policy-map class configuration mode and returns to privileged EXEC mode.

Examples

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a policy-map is created to apply QoS to all fragments named BestEffort.

```
policy-map main-interface
 class data service-fragment BestEffort
  shape average 400000000
```

In the following example, two fragments are created and then classified collectively using a service fragment.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example shows the creation of two fragments called BestEffort in the subinterface policy-maps, followed by a sample configuration for the **service-fragment** called BestEffort to aggregate the queues at the main interface policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber2
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map main-interface
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
```

Troubleshooting Tips

Ensure that all class statements that are supposed to be part of the same service fragment share the same *fragment-class-name*.

What to Do Next

The policy map (traffic policy) must be attached to an interface. This task is documented in the "Attaching a Traffic Policy to an Interface Using the MQC" section in chapter "Applying QoS Features Using the MQC."

Configuring QoS: Policies Aggregation on Gigabit Etherchannels

To properly configure QoS: Policies Aggregation on a Gigabit Etherchannel bundle, the following actions must be completed:

- Service-fragment traffic classes must be configured and attached to the main physical interfaces.
- Fragment traffic classes must be configured and attached to the member link subinterfaces.

Configuring Service Fragments on a Physical Interface Supporting a Gigabit Etherchannel Bundle

Before you begin

This procedure assumes that a service fragment traffic class has already been created. A service fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy-Map" section. The procedure for creating a service fragment traffic classes is documented in the "Configuring a Service Fragment Traffic Class" section.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions document only the procedure for attaching a policy-map that already has a fragment traffic class to a member link subinterface.

Note For proper behavior, when a port-channel member link goes down, all member links should have the same policy-map applied.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *service-fragment-class-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service-policy configuration.
Step 4	**service-policy output** *service-fragment-class-name* **Example:** `Device(config-if)# service-policy output aggregate-member-link`	Attaches a service policy that contains a service fragment default traffic class to the physical Gigabit Ethernet interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Examples

In the following example, the policy-map aggregate-member-link is attached to the physical interface.

```
interface GigabitEthernet1/1/1
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 service-policy output aggregate-member-link
```

What to do next

Ensure that the fragment class name is consistent across service-fragment and fragment class definitions. Continue to the “Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces” section.

Troubleshooting Tips

Ensure that the *fragment-class-name* is consistent across service-fragment and fragment-class definitions.

What to Do Next

Attach the fragment service policy on the Gigabit Etherchannel member link subinterfaces. This task is documented in the "Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces" section on page 14.

Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces

Before you begin

This task assumes that a service-fragment traffic class has already been created. A service-fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy Map" section on page 6. The procedure

for creating a service-fragment traffic classes is documented in the "Configuring a Service Fragment Traffic Class" section on page 8.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions only document the procedure for attaching a policy map that already has a fragment traffic class to a member link subinterface.

Note Fragments cannot be used for traffic on two or more physical interfaces. The GEC must all be on the same physical interface for this configuration to work properly.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-interface-number.port-channel-subinterface-number*
4. **service-policy output** *fragment-class-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-interface-number.port-channel-subinterface-number* **Example:** `Router(config)# interface port-channel 1.100`	Enters subinterface configuration mode to configure a Etherchannel member link subinterface.
Step 4	**service-policy output** *fragment-class-name* **Example:** `Router(config-subif)# service-policy output subscriber`	Attaches a service policy that contains a fragment default traffic class to the Etherchannel member link subinterface.

Example

Note This example shows a sample configuration that is supported for the original QoS: Policies Aggregation feature in releases prior to Cisco IOS XE Release 2.6. By following the newer policy-map configuration guidelines for the updates in Cisco IOS XE Release 2.6, it can be adapted to the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature.

In the following example, the service policy named subscriber has a fragment default traffic class and is attached to the member link subinterface of a Gigabit Etherchannel bundle.

Note This example only shows how to attach a fragment default traffic class to the member link subinterface of a Gigabit Etherchannel bundle. This configuration is incomplete and would not classify default traffic appropriately until the physical interface was configured to support a service-fragment traffic class.

```
policy-map subscriber
 class voice
  priority level 1
 class video
  priority level 2
 class class-default fragment BE
  shape average 100000000
  bandwidth remaining ratios 80
policy-map aggregate-member-link
 class BestEffort service-fragment BE
  shape average 100000000
!
interface Port-channel1
 ip address 172.16.2.3 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 192.168.2.100 255.255.255.0
 service-policy output subscriber
!
```

Troubleshooting Tips

This configuration will not work until a service-fragment default traffic class is created to classify the default traffic classes marked as fragments. This service-fragment traffic class must be configured for this configuration to have any affect on network traffic.

How to Configure QoS: Policies Aggregation MQC

Some backward-compatibility exists between support of policies aggregation feature configuration in Cisco IOS XE Release 2.6 and prior Cisco IOS XE software releases. However, we recommend that you follow these upgrade guidelines for any physical interface where you want to move to the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature configuration.

For best results, you should upgrade any service policies configuration that you implemented prior to Cisco IOS XE Release 2.6, to the latest supported configuration.

The original and enhanced QoS: Policies Aggregation feature configuration can only reside on the same Cisco ASR 1000 Series Router if the mixed configuration does not reside on the same physical interface. In other words, you can support the original configuration for one physical interface, and the enhanced configuration on a different physical interface.

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature requires the same configuration of a fragment traffic class as the original feature, using the **class class-default fragment** command to enable and then define all subinterface policies aggregation, both for the default traffic class and the other traffic classes.

In the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature, the queueing features for the aggregate class queues (with traffic from the corresponding classes identified at the subinterfaces), are configured at the main-interface policy map.

Upgrading Your Service Policies for QoS: Policies Aggregation MQC

Before You Begin

Upgrading your service policies to support the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature assumes the following network conditions:

- The corresponding class-map statements appropriate for your network traffic are already configured.
- QoS service policies aggregation has been previously configured and applied for the main-interface policy map for a given physical interface and its corresponding subinterfaces, or subscriber interfaces, prior to Cisco IOS XE Release 2.6 for the default traffic class.
- A port on the same physical interface where you have previously configured the service policies aggregation feature prior to Cisco IOS XE Release 2.6 needs to support the configuration for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface.

Upgrade Tasks

SUMMARY STEPS

1. Configure the service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature.
2. Remove any service policies configured prior to Cisco IOS XE Release 2.6 for any prior configured policies aggregation features using the **no service-policy** and **no policy-map** commands as follows:
3. Apply the new service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature at the appropriate interfaces using the **service-policy output** command as follows:

DETAILED STEPS

Step 1 Configure the service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature.

See the tasks described in the "Configuring QoS Policies Aggregation MQC Traffic Classes" section on page 18.

Step 2 Remove any service policies configured prior to Cisco IOS XE Release 2.6 for any prior configured policies aggregation features using the **no service-policy** and **no policy-map** commands as follows:

a) At each of the subinterfaces, configure the **no service-policy** command. Be sure to remove the policies at the subinterfaces first.

b) At the physical interface, configure the **no service-policy**command.

Step 3 Apply the new service policies for the QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature at the appropriate interfaces using the **service-policy output** command as follows:

a) At the physical interface, configure the **service-policy output** command.

b) At each of the subinterfaces, configure the **service-policy output** command.

Configuring QoS: Policies Aggregation MQC Traffic Classes

Configuring Traffic Classes on the Subscriber Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name*
5. **account** [**drop**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map subscriber1`	Specifies the name of the traffic policy to configure and enters policy map configuration mode.

	Command or Action	Purpose
Step 4	**class** *class-name* **Example:** `Router(config-pmap)# class EF`	Specifies the name of the traffic class to be aggregated at the main-interface policy map, and enters policy-map class configuration mode. **Note** Do not configure any queueing features for this class. Queueing is configured and aggregated at the main-interface policy map for all subinterfaces associated with this class and physical interface.
Step 5	**account** [**drop**] **Example:** `Router(config-pmap-c)# account`	(Optional) Enables collection of statistics for packets matching the traffic class where this command is configured, where the **drop** keyword collects all packet drop statistics. Collection of drop statistics is the default.

Example

The following example configures the EF traffic class for policies aggregation at the subscriber subinterface with collection of drop statistics:

```
policy-map subscriber1
 class EF
 account
```

What to Do Next

Perform this task for all traffic classes that you want to aggregate, then perform the task in the "Configuring the Fragment Traffic Class on a Subinterface" section on page 19.

Configuring the Fragment Traffic Class on a Subinterface

What to Do Next

If you are upgrading your subinterface policy-map configuration from an earlier implementation of the QoS: Policies Aggregation feature, then remove the current service-policy from the subinterface using the **no service-policy** command.

Apply the new policy map to outbound traffic on the subinterface using the **service-policy output** command.

Configuring Traffic Classes at the Main Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name*
5. **priority level** *level*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map main-interface`	Specifies the name of the traffic policy to configure and enters policy map configuration mode.
Step 4	**class** *class-name* **Example:** `Router(config-pmap)# class EF`	Specifies the name of the traffic class to be aggregated at the main-interface policy map, and enters policy-map class configuration mode.
Step 5	**priority level** *level* **Example:** `Router(config-pmap-c)# priority level 1`	Enters a QoS configuration command. The queueing features that are currently supported are **bandwidth**, **priority**, **shape**, and **random-detect exponential-weighting-constant**. Multiple QoS queueing commands can be entered.

Example

The following example configures three traffic classes at the main-interface policy map, along with the aggregate service-fragment data class:

```
policy-map main-interface
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
```

What to Do Next

Perform this task to define queueing features for all traffic classes that you want to aggregate, then perform the task in the "Configuring the Service Fragment Traffic Class at the Main Interface" section on page 21.

Configuring the Service Fragment Traffic Class at the Main Interface

What to Do Next

After configuring multiple default class statements as fragments in a policy-map, a separate policy-map with a class statement using the **service-fragment** keyword must be configured to apply QoS to the class statements configured as fragments.

This process is documented in the "Configuring a Service Fragment Traffic Class" section.

Configuring QoS: Policies Aggregation MQC Support

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature also supports configuration of the enhanced service policies on Gigabit Etherchannels according to the subscriber and main-interface configuration guidelines described for this enhancement.

For more information, see the following sections:

Verifying the Traffic Policy Class Policy Information and Drop Statistics

To display information about policy-map configuration and subscriber drop statistics enabled using the account command, use the **show policy-map interface** command:

```
Router# show policy-map interface port-channel 1.1
Port-channel1.1
   Service-policy input: input_policy
     Class-map: class-default (match-any)
       0 packets, 0 bytes
       5 minute offered rate 0000 bps, drop rate 0000 bps
       Match: any
       QoS Set
       dscp default
       No packet marking statistics available
   Service-policy output: Port-channel_1_subscriber
     Class-map: EF (match-any)
       105233 packets, 6734912 bytes
       5 minute offered rate 134000 bps, drop rate 0000 bps
       Match: dscp ef (46)
       Match: access-group name VLAN_REMARK_EF
       Match: qos-group 3
       Account QoS statistics
         Queueing
           Packets dropped 0 packets/0 bytes
       QoS Set
       cos 5
       No packet marking statistics available
       dscp ef
       No packet marking statistics available
     Class-map: AF4 (match-all)
       105234 packets, 6734976 bytes
       5 minute offered rate 134000 bps, drop rate 0000 bps
       Match: dscp cs4 (32)
       Account QoS statistics
         Queueing
           Packets dropped 0 packets/0 bytes
       QoS Set
       cos 4
       No packet marking statistics available
```

```
Class-map: AF1 (match-any)
  315690 packets, 20204160 bytes
  5 minute offered rate 402000 bps, drop rate 0000 bps
  Match: dscp cs1 (8)
  Match: dscp af11 (10)
  Match: dscp af12 (12)
  Account QoS statistics
    Queueing
      Packets dropped 0 packets/0 bytes
  QoS Set
  cos 1
  No packet marking statistics available
Class-map: class-default (match-any) fragment Port-channel_BE
  315677 packets, 20203328 bytes
  5 minute offered rate 402000 bps, drop rate 0000 bps
  Match: any
  Queueing
    queue limit 31250 bytes
    (queue depth/total drops/no-buffer drops) 0/0/0
    (pkts output/bytes output) 315679/20203482
    bandwidth remaining ratio 1
```

Configuration Examples for QoS: Policies Aggregation

Example: QoS: Policies Aggregation

Note This example shows a sample configuration that is supported in the original QoS: Policies Aggregation feature prior to Cisco IOS XE Release 2.6.

In the following example, QoS: Policies Aggregation is used to define a fragment class of traffic to classify default traffic using the default traffic class named BestEffort. All default traffic from the policy maps named subscriber1 and subscriber2 is part of the fragment default traffic class named BestEffort. This default traffic is then shaped collectively by creating a class called data that uses the **service-fragment** keyword and the **shape** command.

Note the following about this example:

- The *class-name* for each fragment default traffic class is "BestEffort."
- The *class-name* of "BestEffort" is also used to define the class where the **service-fragment** keyword is entered. This class applies a shaping policy to all traffic forwarded using the fragment default traffic classes named "BestEffort."

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

```
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map input_policy
 class class-default
  set dscp default
policy-map main-interface
 class data service-fragment BestEffort
  shape average 400000000
interface portchannel1.1001
 encapsulation dot1q 1001
 service-policy output subscriber1
 service-policy input input_policy
interface portchannel1.1002
 encapsulation dot1q 1002
 service-policy output subscriber2
 service-policy input input_policy
interface gigabitethernet 0/1
 description member-link1
 port channel 1
 service-policy output main-interface
interface gigabitethernet 0/2
 description member-link2
 port channel 1
service-policy output main-interface
```

Example: Gigabit Etherchannel QoS Policies Aggregation

Note This example shows a sample configuration that is supported in the original QoS: Policies Aggregation feature prior to Cisco IOS XE Release 2.6.

In the following example, policy map subscriber is configured with a fragment class named BE. The fragment is then configured as part of a policy map named aggregate-member-link. Policy map subscriber is then attached to the bundle subinterfaces while policy map aggregate-member-link is attached to the physical interface.

```
port-channel load-balancing vlan-manual
class-map match-all BestEffort
!
class-map match-all video
!
class-map match-all voice
!
policy-map subscriber
 class voice
  priority level 1
 class video
  priority level 2
 class class-default fragment BE
  shape average 100000000
```

```
  bandwidth remaining ratios 80
policy-map aggregate-member-link
 class BestEffort service-fragment BE
  shape average 100000000
!
interface Port-channel1
 ip address 10.1.1.3 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 10.1.2.1 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.200
 encapsulation dot1Q 200
 ip address 10.1.2.2 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 10.1.2.3 255.255.255.0
 service-policy output subscriber
!
interface GigabitEthernet1/1/1
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
```

Example: QoS: Policies Aggregation MQC Support at Main Interface

Note This example shows a sample configuration that is supported beginning in Cisco IOS XE Release 2.6.

At the main-interface policy map called Port-channel_1_main_policy, the queueing features for the DSCP-based subscriber traffic classes are configured. You can also see the use of byte-based queue limits and random-detect thresholds implemented at the main-interface queues.

The service fragment called Port-channel_BE is also configured to aggregate the traffic from the subscriber class-default fragment class.

```
policy-map Port-channel_1_main_policy
 class EF
  priority level 1
  queue-limit 547500 bytes
 class AF4
  priority level 2
  queue-limit 4037500 bytes
 class AF1
  bandwidth remaining ratio 90
  queue-limit 750000 bytes
  random-detect dscp-based
  random-detect dscp 8 750000 bytes 750000 bytes
  random-detect dscp 10 750000 bytes 750000 bytes
  random-detect dscp 12 600000 bytes 675000 bytes
```

```
 class data service-fragment Port-channel_BE
  shape average 250000000
  bandwidth remaining ratio 1
!
```

In this example, the policy map Port-channel_1_subscriber is configured with a fragment class named Port-channel_BE. (For simplicity, only a single subinterface policy is shown.) This enable queueing and policies aggregation for the subscriber traffic classes at the main-interface policy map.

The Port-channel_1_subscriber policy map identifies the DSCP-based traffic classes of EF, AF4, and AF1 and enables collection of drop statistics for those classes.

```
policy-map Port-channel_1_subscriber
 class EF
  account
  set cos 5
  set dscp ef
 class AF4
  account
  set cos 4
 class AF1
  account
  set cos 1
 class class-default fragment Port-channel_BE
  bandwidth remaining ratio 1
  queue-limit 31250 bytes
!
port-channel load-balancing vlan-manual
!
interface Port-channel1
 no ip address
 no negotiation auto
!
```

The service policies are applied first to the physical interface, and then to the subinterfaces as shown:

```
interface GigabitEthernet1/2/0
 no ip address
 negotiation auto
 no cdp enable
 service-policy output Port-channel_1_main_policy
 channel-group 1
!
interface GigabitEthernet2/2/0
 no ip address
 negotiation auto
 service-policy output Port-channel_1_main_policy
 channel-group 1
!
interface Port-channel1.1
 encapsulation dot1Q 2 primary GigabitEthernet1/2/0 secondary GigabitEthernet2/2/0
 ip address 10.0.0.2 255.255.255.0
 service-policy output Port-channel_1_subscriber
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular Quality of Service Command-Line Interface	"Applying QoS Features Using the MQC" module
Distribution of Remaining Bandwidth Using Ratio	"Distribution of Remaining Bandwidth Using Ratio" module
Class-Based Shaping	"Regulating Packet Flow--Using Class-Based Traffic Shaping" module

Standards

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

MIBs

MIB	MIBs Link
CISCO-CLASS-BASED-QOS-MIB	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

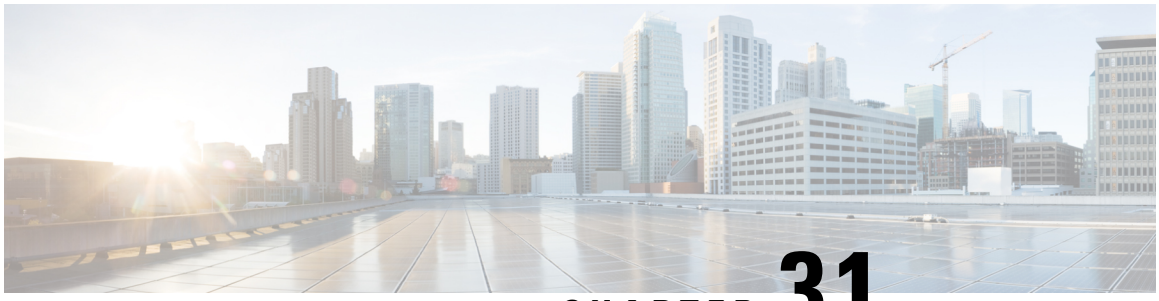
Feature Information for QoS: Policies Aggregation

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 35: Feature Information for QoS: Policies Aggregation

Feature Name	Releases	Feature Information
QoS: Policies Aggregation	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers. The following command was modified: **class (policy-map)**.
QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface	Cisco IOS XE Release 2.6	This feature was enhanced to support queueing aggregation at the primary interface for other traffic classes, including DSCP-based classes such as EF, AF1, and AF4 traffic classes. With this enhancement, other traffic classes from different subinterfaces share a common queue for that traffic class. Other enhancements include the ability to configure and show per-subscriber drop statistics on the aggregate queues and byte-based queue limits and WRED thresholds. In Cisco IOS XE Release 2.6, support for the CISCO-CLASS-BASED-QOS-MIB was added. The following commands are new or modified: **account**, **show policy-map interface**.



CHAPTER 31

Configuring IP to ATM Class of Service

This module describes the tasks for configuring the IP to ATM Class of Service (CoS), a feature suite that maps QoS characteristics between IP and ATM.

Use Cisco Feature Navigator to find information about platform support and software image support. Cisco Feature Navigator enables you to determine which Cisco IOS and Catalyst OS software images support a specific software release, feature set, or platform. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn . An account on Cisco.com is not required.

IP to ATM CoS on a Single ATM VC Configuration Task List

To configure IP to ATM CoS for a single ATM virtual circuit (VC), perform the tasks described in the following sections. The tasks in the first two sections are required; the tasks in the remaining sections are optional.

The IP to ATM CoS feature requires ATM permanent virtual circuit (PVC) management.

Defining the WRED Parameter Group

Command	Purpose
`Router(config)# random-detect-group group-name`	Defines the WRED or VIP-distributed WRED (DWRED) parameter group.

Configuring the WRED Parameter Group

SUMMARY STEPS

1. Device(config)# **random-detect-group** *group-name*
2. Device(config)# **exponential-weighting-constant** *exponent*
3. Device(config)# **precedence** *precedence min-threshold max-threshold mark-probability-denominator*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Device(config)# **random-detect-group** *group-name*	Specifies the WRED or DWRED parameter group.
Step 2	Device(config)# **exponential-weighting-constant** *exponent*	Configures the exponential weight factor for the average queue size calculation for the specified WRED or DWRED parameter group. or
Step 3	Device(config)# **precedence** *precedence min-threshold max-threshold mark-probability-denominator*	Configures the specified WRED or DWRED parameter group for a particular IP precedence.

Displaying the WRED Parameters

Command	Purpose
`Router# show queueing random-detect [interface atm_subinterface [vc [[vpi/] vci]]]`	Displays the parameters of every VC with WRED or DWRED enabled on the specified ATM subinterface.

Displaying the Queueing Statistics

Command	Purpose
`Router# show queueing interface interface-number [vc [[vpi/] vci]]`	Displays the queueing statistics of a specific VC on an interface.

IP to ATM CoS on an ATM Bundle Configuration Task List

To configure IP to ATM CoS on an ATM bundle, perform the tasks in the following sections.

The IP to ATM CoS feature requires ATM PVC management.

Creating a VC Bundle

Command	Purpose
`Router(config-subif)# bundle bundle-name`	Creates the specified bundle and enters bundle configuration mode.

Applying Bundle-Level Parameters

Configuring Bundle-Level Parameters

Command	Purpose
`Device(config-atm-bundle)# protocol protocol {protocol-address | inarp} [[no] broadcast]`	Configures a static map or enables Inverse Address Resolution Protocol (Inverse ARP) or Inverse ARP broadcasts for the bundle. **Note** Bundle-level parameters can be applied either by assigning VC classes or by directly applying them to the bundle. Parameters applied through a VC class assigned to the bundle are superseded by those applied at the bundle level. Bundle-level parameters are superseded by parameters applied to an individual VC.
`Device(config-atm-bundle)# encapsulation aal-encap`	Configures the ATM adaptation layer (AAL) and encapsulation type for the bundle.
`Device(config-atm-bundle)# inarp minutes`	Configures the Inverse ARP time period for all VC bundle members.
`Device(config-atm-bundle)# broadcast`	Enables broadcast forwarding for all VC bundle members.
`Device(config-atm-bundle)# oam retry up-count down-count retry frequency`	Configures the VC bundle parameters related to operation, administration, and maintenance (OAM) management.
`Device(config-atm-bundle)# oam-bundle [manage] [frequency]`	Enables end-to-end F5 OAM loopback cell generation and OAM management for all VCs in the bundle.

Configuring VC Class Parameters to Apply to a Bundle

Command	Purpose
`Router(config-vc-class)# oam-bundle [manage] [frequency]`	Enables end-to-end F5 OAM loopback cell generation and OAM management for all VCs in the bundle. **Note** Use of a VC class allows you to configure a bundle applying multiple attributes to it at once because you apply the class itself to the bundle. Use of a class allows you to generalize a parameter across all VCs, after which (for some parameters) you can modify that parameter for individual VCs. (See the section "Applying Parameters to Individual VCs" for more information.)

Attaching a Class to a Bundle

Command	Purpose
`(config-atm-bundle)#` **`class-bundle`** *`vc-class-name`*	Configures a bundle with the bundle-level commands contained in the specified VC class. **Note** Parameters set through bundle-level commands contained in the VC class are applied to the bundle and all of its VC members. Bundle-level parameters applied through commands configured directly on the bundle supersede those applied through a VC class. Some bundle-level parameters applied through a VC class or directly to the bundle can be superseded by commands that you directly apply to individual VCs in bundle-vc configuration mode.

Committing a VC to a Bundle

Command	Purpose
`Device(config-atm-bundle)#` **`pvc-bundle`** *`pvc-name`* [*`vpi/`*] [*`vci`*]	Adds the specified VC to the bundle and enters bundle-vc configuration mode in order to configure the specified VC bundle member.

Applying Parameters to Individual VCs

Configuring a VC Bundle Member Directly

Command	Purpose
`Device(config-if-atm-member)#` **`ubr`** *`output-pcr`* [*`input-pcr`*]	Configures the VC for unspecified bit rate (UBR) QoS and specifies the output peak cell rate (PCR) for it.
`Device(config-if-atm-member)#` **`ubr+`** *`output-pcr output-mcr`* [*`input-pcr`*] [*`input-mcr`*]	Configures the VC for UBR QoS and specifies the output PCR and output minimum guaranteed cell rate for it.
`Device(config-if-atm-member)#` **`vbr-nrt`** *`output-pcr output-scr output-mbs`* [*`input-pcr`*] [*`input-scr`*] [*`input-mbs`*]	Configures the VC for variable bit rate nonreal-time (VBR-nrt) QoS and specifies the output PCR, output sustainable cell rate, and output maximum burst cell size for it.
`Device(config-if-atm-member)#` **`precedence`** [**`other`** | *`range`*]	Configures the precedence levels for the VC.
`Device(config-if-atm-member)#` **`bump`** {**`implicit`** | **`explicit`** *`precedence-level`* | **`traffic`**}	Configures the bumping rules for the VC.

Command	Purpose
`Device(config-if-atm-member)# protect {group | vc}`	Configures the VC to belong to the protected group of the bundle or to be an individually protected VC bundle member.

Configuring VC Class Parameters to Apply to a VC Bundle Member

Command	Purpose
`Device(config-vc-class)# bump {implicit | explicit` *precedence-level* `| traffic}`	Specifies the bumping rules for the VC member to which the class is applied. These rules determine to which VC in the bundle traffic is directed when the carrier VC bundle member goes down. **Note** You can also add the following commands to a VC class to be used to configure a VC bundle member: **ubr**, **ubr+**, and **vbr-nrt**. When a VC is a member of a VC bundle, the following commands cannot be used in vc-class mode to configure the VC: **encapsulation**, **protocol**, **inarp**, and **broadcast**. These commands are useful only at the bundle level, not the bundle member level. Configuration for an individual VC overrides the collective configuration applied to all VC bundle members through application of a VC class to the bundle.
`Device(config-vc-class)# precedence` *precedence min-threshold max-threshold mark-probability-denominator*	Defines precedence levels for the VC member to which the class is applied.
`Device(config-vc-class)# protect {group | vc}`	Configures the VC as a member of the protected group of the bundle or as an individually protected VC.

Applying a VC Class to a Discrete VC Bundle Member

Command	Purpose
`Device(config-if-atm-member)# class-vc` *vc-class-name*	Assigns a VC class to a VC bundle member.

Configuring a VC Not to Accept Bumped Traffic

Command	Purpose
`Device(config-if-atm-member)# no bump traffic`	Configures the VC not to accept any bumped traffic that would otherwise be redirected to it.

Monitoring and Maintaining VC Bundles and Their VC Members

Command	Purpose
`Device# show atm bundle bundle-name`	Displays the bundle attributes assigned to each bundle VC member and the current working status of the VC members.
`Device# show atm bundle bundle-name statistics [detail]`	Displays statistics or detailed statistics on the specified bundle.
`Device# show atm map`	Displays a list of all configured ATM static maps to remote hosts on an ATM network and on ATM bundle maps.
`Device# debug atm bundle errors`	Displays information on bundle errors.
`Device# debug atm bundle events`	Displays a record of bundle events.

Per-VC WFQ and CBWFQ Configuration Task List

To configure IP to ATM CoS for per-VC WFQ and CBWFQ, perform the tasks described in the following sections.

The IP to ATM CoS feature requires ATM PVC management.

Configuring Class-Based Weighted Fair Queueing

Before configuring CBWFQ for a VC, you must perform the following tasks using standard CBWFQ commands:

- Create one or more classes to be used to classify traffic sent across the VC
- Define a policy-map containing the classes to be used as the service policy

Note You can configure class policies for as many classes as are defined on the router, up to the maximum of 64. However, the total amount of bandwidth allocated for all classes included in a policy-map to be attached to a VC must not exceed 75 percent of the available bandwidth of the VC. The remaining 25 percent of available bandwidth is used for encapsulation, such as the ATM cell overhead (also referred to as ATM cell tax), routing and best-effort traffic, and other functions that assume overhead. For more information on bandwidth allocation, see the "Congestion Management Overview" module.

Because CBWFQ gives you minimum bandwidth guarantee, you can only apply CBWFQ to VCs having these classes of service: available bit rate (ABR) and variable bit rate (VBR). You cannot apply per-VC WFQ and CBWFQ to UBR and unspecified bit rate plus (UBR+) VCs because both of these service classes are best-effort classes that do not guarantee minimum bandwidth. When CBWFQ is enabled for a VC, all classes configured as part of the service policy are installed in the fair queueing system.

In addition to configuring CBWFQ at the VC level, the IP to ATM CoS feature allows you to configure flow-based WFQ at the VC level. Because flow-based WFQ gives you best-effort class of service--that is, it does not guarantee minimum bandwidth--you can configure per-VC WFQ for all types of CoS VCs: ABR, VBR, UBR, and UBR+.

Per-VC WFQ uses the class-default class. Therefore, to configure per-VC WFQ, you must first create a policy-map and configure the class-default class. (You need not create the class-default class, which is predefined, but you must configure it.) For per-VC WFQ, the class-default class must be configured with the **fair-queue** policy-map class configuration command.

In addition to configuring the **fair-queue** policy-map class configuration command, you can configure the default class with either the **queue-limit** command or the **random-detect**command, but not both. Moreover, if you want the default class to use flow-based WFQ, you cannot configure the default class with the **bandwidth** policy-map class configuration command--to do so would disqualify the default class as flow-based WFQ, and therefore limit application of the service policy containing the class to ABR and VBR VCs.

Attaching a Service Policy and Enabling CBWFQ for a VC

Attaching a Policy-Map to a Standalone VC and Enabling CBWFQ

Command	Purpose
`Router(config-if-atm-vc)#` **`service-policy output`** *`policy-map`*	Enables CBWFQ and attaches the specified service policy-map to the VC being created or modified.

Attaching a Policy-Map to an Individual VC and Enabling CBWFQ

Command	Purpose
`Router(config-if-atm-member)#` **`service-policy output`** *`policy-map`*	Enables CBWFQ and attaches the specified service policy-map to the VC being created or modified.

Note The **service-policy output** and **random-detect-group** commands are mutually exclusive; you cannot apply a WRED group to a VC for which you have enabled CBWFQ through application of a service policy. Moreover, before you can configure one command, you must disable the other if it is configured.

Configuring a VC to Use Flow-Based WFQ

SUMMARY STEPS

1. Device(config)# **policy-map** *policy-map*
2. Device(config-pmap)# **class class-default** *default-class-name*
3. Device(config-pmap-c)# **fair-queue** *number-of-dynamic-queues*
4. Do one of the following:
   - Device(config-pmap-c)# **queue-limit** *number-of-packets*

- Device(config-pmap-c)# **random-detect**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Device(config)# **policy-map** *policy-map*	Specifies the name of the policy-map to be created or modified.
Step 2	Device(config-pmap)# **class class-default** *default-class-name*	Specifies the default class so that you can configure or modify its policy. **Note** You can include other classes in the same policy-map as the one that contains the flow-based WFQ class. Packets not otherwise matched are selected by the default class-default class match criteria.
Step 3	Device(config-pmap-c)# **fair-queue** *number-of-dynamic-queues*	Specifies the number of dynamic queues to be reserved for use by flow-based WFQ running on the default class. **Note** By default--that is, even if you do not configure the class-default class with the **fair-queue** policy-map class configuration command and you do not configure it with the **bandwidth** policy-map class configuration command--the default class is defined as flow-based WFQ.
Step 4	Do one of the following: • Device(config-pmap-c)# **queue-limit** *number-of-packets* • Device(config-pmap-c)# **random-detect**	Specifies the maximum number of packets that can be queued for the class. Enables WRED. The class policy will drop packets using WRED instead of tail drop.

Attaching a Policy-Map to a Standalone VC and Enabling WFQ

Command	Purpose
`Device(config-if-atm-vc)#` **`service-policy output`** *`policy-map`*	Enables WFQ for the VC by attaching the specified policy-map containing the class-default class to the VC being created or modified.

Attaching a Policy-Map to an Individual VC and Enabling WFQ

Command	Purpose
`Device(config-if-atm-member)#` **`service-policy output`** *`policy-map`*	Enables WFQ for the VC bundle member by attaching the specified policy-map containing the class-default class to the VC bundle member.

Monitoring per-VC WFQ and CBWFQ

Command	Purpose
`Device# show policy-map interface interface-number [vc [vpi/] vci]]`	Displays the contents of packets inside a queue for a particular interface or VC.

Enabling Logging of Error Messages to the Console

Command	Purpose
`Router(config)# logging console level`	Limits messages logged to the console based on severity.

IP to ATM CoS Configuration Examples

Example Single ATM VC with WRED Group and IP Precedence

The following example creates a PVC on an ATM interface and applies the WRED parameter group called sanjose to that PVC. Next, the IP Precedence values are configured for the WRED parameter group sanjose.

```
interface ATM1/1/0.46 multipoint
 ip address 200.126.186.2 255.255.255.0
 no ip mroute-cache
 shutdown
pvc 46
 encapsulation aal5nlpid
 random-detect attach sanjose
!
random-detect-group sanjose
 precedence 0  200   1000 10
 precedence 1  300   1000 10
 precedence 2  400   1000 10
 precedence 3  500   1000 10
 precedence 4  600   1000 10
 precedence 5  700   1000 10
 precedence 6  800   1000 10
 precedence 7  900   1000 10
```

Example VC Bundle Configuration Using a VC Class

This example configures VC bundle management on a router that uses Intermediate System-to-Intermediate System (IS-IS) as its IP routing protocol.

Bundle-Class Class

At the outset, this configuration defines a VC class called bundle-class that includes commands that set VC parameters. When the class bundle-class is applied at the bundle level, these parameters are applied to all VCs that belong to the bundle. Note that any commands applied directly to an individual VC of a bundle in bundle-vc

mode take precedence over commands applied globally at the bundle level. Taking into account hierarchy precedence rules, VCs belonging to any bundle to which the class bundle-class is applied will be characterized by these parameters: aal5snap encapsulation, broadcast on, use of Inverse Address Resolution Protocol (ARP) to resolve IP addresses, and operation, administration, and maintenance (OAM) enabled.

```
router isis
 net 49.0000.0000.0000.1111.00
vc-class atm bundle-class
 encapsulation aal5snap
 broadcast
 protocol ip inarp
 oam-bundle manage 3
 oam retry 4 3 10
```

The following sections of the configuration define VC classes that contain commands specifying parameters that can be applied to individual VCs in a bundle by assigning the class to that VC.

Control-Class Class

When the class called control-class is applied to a VC, the VC carries traffic whose IP Precedence level is 7. When the VC to which this class is assigned goes down, it takes the bundle down with it because this class makes the VC a protected one. The QoS type of a VC using this class is vbr-nrt.

```
vc-class atm control-class
 precedence 7
 protect vc
 vbr-nrt 10000 5000 32
```

Premium-Class Class

When the class called premium-class is applied to a VC, the VC carries traffic whose IP Precedence levels are 6 and 5. The VC does not allow other traffic to be bumped onto it. When the VC to which this class is applied goes down, its bumped traffic will be redirected to a VC whose IP Precedence level is 7. This class makes a VC a member of the protected group of the bundle. When all members of a protected group go down, the bundle goes down. The QoS type of a VC using this class is vbr-nrt.

```
vc-class atm premium-class
 precedence 6-5
 no bump traffic
 protect group
 bump explicitly 7
 vbr-nrt 20000 10000 32
```

Priority-Class Class

When the class called priority-class is applied to a VC, the VC is configured to carry traffic with IP Precedence in the 4-2 range. The VC uses the implicit bumping rule, it allows traffic to be bumped, and it belongs to the protected group of the bundle. The QoS type of a VC using this class is ubr+.

```
vc-class atm priority-class
 precedence 4-2
 protect group
 ubr+ 10000 3000
```

Basic-Class Class

When the class called basic-class is applied to a VC, the VC is configured through the **precedence other** command to carry traffic with IP Precedence levels not specified in the profile. The VC using this class belongs to the protected group of the bundle. The QoS type of a VC using this class is ubr.

```
vc-class atm basic-class
 precedence other
 protect group
 ubr 10000
```

The following sets of commands configure three bundles that the router subinterface uses to connect to three of its neighbors. These bundles are called new-york, san-francisco, and los-angeles. Bundle new-york has four VC members, bundle san-francisco has four VC members, and bundle los-angeles has three VC members.

new-york Bundle

The first part of this example specifies the IP address of the subinterface, the router protocol--the router uses IS-IS as an IP routing protocol--and it creates the first bundle called new-york and enters bundle configuration mode:

```
interface atm 1/0.1 multipoint
 ip address 10.0.0.1 255.255.255.0
 ip router isis
 bundle new-york
```

From within bundle configuration mode, the next portion of the configuration uses two protocol commands to enable IP and Open Systems Interconnect (OSI) traffic flows in the bundle. The OSI routing packets will use the highest precedence VC in the bundle. The OSI data packets, if any, will use the lowest precedence VC in the bundle. If configured, other protocols, such as IPX or AppleTalk, will always use the lowest precedence VC in the bundle.

As the indentation levels of the preceding and following commands suggest, subordinate to bundle new-york is a command that configures its protocol and a command that applies the class called bundle-class to it.

```
 protocol ip 1.1.1.2 broadcast
 protocol clns 49.0000.0000.2222.00 broadcast
 class-bundle bundle-class
```

The class called bundle-class, which is applied to the bundle new-york, includes a **protocol ip inarp** command. According to inheritance rules, **protocol ip**, configured at the bundle level, takes precedence over **protocol ip inarp** specified in the class bundle-class.

The next set of commands beginning with **pvc-bundle ny-control 207**, which are further subordinate, add four VCs (called ny-control, ny-premium, ny-priority, and ny-basic) to the bundle new-york. A particular class--that is, one of the classes predefined in this configuration example--is applied to each VC to configure it with parameters specified by commands included in the class.

As is the case for this configuration, to configure individual VCs belonging to a bundle, the router must be in bundle mode for the mother bundle. For each VC belonging to the bundle, the subordinate mode is pvc-mode for the specific VC.

The following commands configure the individual VCs for the bundle new-york:

```
 pvc-bundle ny-control 207
  class-vc control-class
```

```
 pvc-bundle ny-premium 206
  class-vc premium-class
 pvc-bundle ny-priority 204
  class-vc priority-class
 pvc-bundle ny-basic 201
  class-vc basic-class
```

san-francisco Bundle

The following set of commands create and configure a bundle called san-francisco. At the bundle configuration level, the configuration commands included in the class bundle-class are ascribed to the bundle san-francisco and to the individual VCs that belong to the bundle. Then, the **pvc-bundle** command is executed for each individual VC to add it to the bundle. After a VC is added and bundle-vc configuration mode is entered, a particular, preconfigured class is assigned to the VC. The configuration commands comprising that class are used to configure the VC. Rules of hierarchy apply at this point. Command parameters contained in the applied class are superseded by the same parameters applied at the bundle configuration level, which are superseded by the same parameters applied directly to a VC.

```
bundle san-francisco
 protocol clns 49.0000.0000.0000.333.00 broadcast
 inarp 1
 class-bundle bundle-class
 pvc-bundle sf-control 307
  class-vc control-class
 pvc-bundle sf-premium 306
  class-vc premium-class
 pvc-bundle sf-priority 304
  class-vc priority-class
 pvc-bundle sf-basic 301
  class-vc basic-class
```

los-angeles Bundle

The following set of commands create and configure a bundle called los-angeles. At the bundle configuration level, the configuration commands included in the class bundle-class are ascribed to the bundle los-angeles and to the individual VCs that belong to the bundle. Then, the **pvc-bundle** command is executed for each individual VC to add it to the bundle. After a VC is added and bundle-vc configuration mode is entered, precedence is set for the VC and the VC is either configured as a member of a protected group (protect group) or as an individually protected VC. A particular class is then assigned to each VC to further characterize it. Rules of hierarchy apply. Parameters of commands applied directly and discretely to a VC take precedence over the same parameters applied within a class to the VC at the bundle-vc configuration level, which take precedence over the same parameters applied to the entire bundle at the bundle configuration level.

```
bundle los-angeles
 protocol ip 1.1.1.4 broadcast
 protocol clns 49.0000.0000.4444.00 broadcast
 inarp 1
 class-bundle bundle-class
 pvc-bundle la-high 407
  precedence 7-5
  protect vc
  class-vc premium-class
 pvc-bundle la-mid 404
  precedence 4-2
  protect group
  class-vc priority-class
 pvc-bundle la-low 401
  precedence other
```

```
   protect group
   class-vc basic-class
```

Example Per-VC WFQ and CBWFQ on a Standalone VC

The following example creates two class maps and defines their match criteria. For the first map class, called class1, the numbered access control list (ACL) 101 is used as the match criterion. For the second map class called class2, the numbered ACL 102 is used as the match criterion.

Next, the example includes these classes in a policy-map called policy1. For class1, the policy includes a minimum bandwidth allocation request of 500 kbps and maximum packet count limit of 30 for the queue reserved for the class. For class2, the policy specifies only the minimum bandwidth allocation request of 1000 kbps, so the default queue limit of 64 packets is assumed. Note that the sum of the bandwidth requests for the two classes comprising policy1 is 75 percent of the total amount of bandwidth (2000 kbps) for the PVC called cisco to which the policy-map is attached.

The example attaches the policy-map called policy1 to a PVC. Once the policy-map policy1 is attached to the PVC, its classes constitute the CBWFQ service policy for that PVC. Packets sent on this PVC will be checked for matching criteria against ACLs 101 and 102 and classified accordingly.

Because the **class-default** command is not explicitly configured for this policy-map, all traffic that does not meet the match criteria of the two classes comprising the service policy is handled by the predefined class-default class, which provides best-effort flow-based WFQ.

```
class-map class1
 match access-group 101
class-map class2
 match access-group 102
policy-map policy1
 class class1
  bandwidth 500
  queue-limit 30
 class class2
  bandwidth 1000
interface ATM1/1/0.46 multipoint
 ip address 200.126.186.2 255.255.255.0
 pvc 46
  vbr-nrt 2000 2000
  encap aal5snap
  service policy output policy1
```

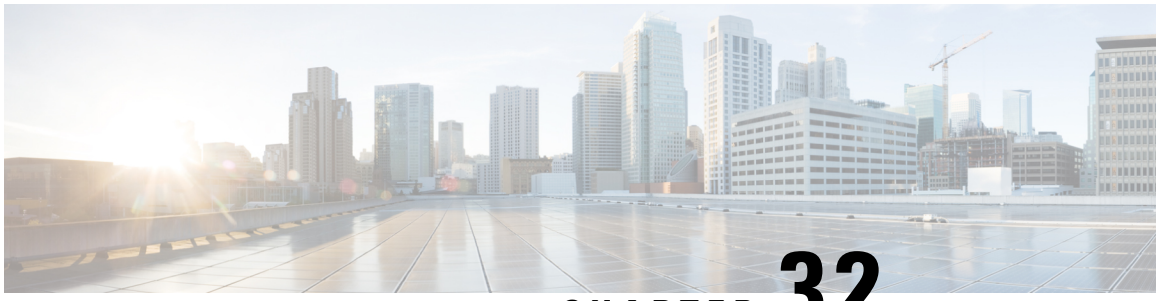
Example Per-VC WFQ and CBWFQ on Bundle-Member VCs

The following example shows a PVC bundle called san-francisco with members for which per-VC WFQ and CBWFQ are enabled and service policies configured. The example assumes that the classes included in the following policy-maps have been defined and that the policy-maps have been created: policy1, policy2, and policy4. For each PVC, the IP to ATM CoS **pvc-bundle** command is used to specify the PVC to which the specified policy-map is to be attached.

Note that PVC 0/34 and 0/31 have the same policy-map attached to them, policy2. Although you can assign the same policy-map to multiple VCs, each VC can have only one policy-map attached at an output PVC.

```
bundle san-francisco
 protocol ip 1.0.2.20 broadcast
 encapsulation aal5snap
 pvc-bundle 0/35
```

```
 service policy output policy1
 vbr-nrt 5000 3000 500
 precedence 4-7
pvc-bundle 0/34
 service policy output policy2
 vbr-nrt 5000 3000 500
 precedence 2-3
pvc-bundle 0/33
 vbr-nrt 4000 3000 500
 precedence 2-3
 service policy output policy4
pvc-bundle 0/31
 service policy output policy2
```



CHAPTER 32

QoS Scheduling

This chapter outlines the process of selecting the next packet to exit an interface and when it should happen (henceforth termed Scheduling). The topic of scheduling exploits the following commands: **priority**, **bandwidth**, **bandwidth remaining**, **shape** and **fair-queue**. Using these commands we can apportion bandwidth when congestion exists and ensure that applications receive the treatment they need to operate over the network.

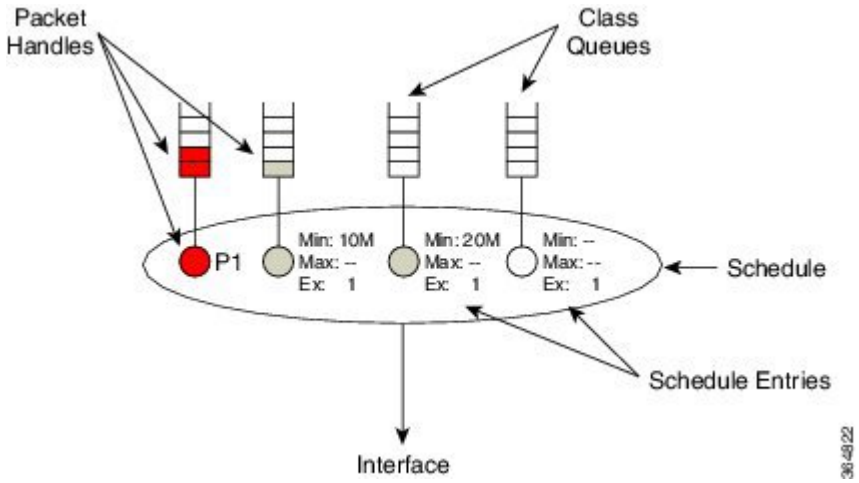
Specifically, this chapter will focus on flat policies attached to physical interfaces. The information presented here should ground your understanding of hierarchical scheduling concepts discussed in int following chapters.

About QoS Scheduling

Definitions

In this section we define core "scheduling" terms.

Figure 13: Scheduling Definitions



Packet Handle

When a router is prepared to forward a packet, it places a *packet handle*, representing that packet, in one of the egress queues. This handle holds information like the length of the packet and the location of the packet in memory.

Class Queues

When egress QoS is configured, a *class queue* is created for each class where we configure a queuing action. Similarly, we create an *implicit class-default queue* for any traffic not matching one of the explicitly-created queuing classes. If you configure a class with only non-queuing actions (e.g., a class with only marking configured), "matching" packets will be enqueued in the class-default queue.

Schedule

You should view a *schedule* (scheduler) as the decision maker. By selecting the packet handle, the schedule chooses which packet should next exit and when to send it. In the diagram above the "oval" represents a single schedule that selects a packet from one of the class queues.

Note An individual schedule is created for each interface.

Schedule Entry

For a schedule to choose between queues it needs to know each queues' expected treatment. We store this type of information in a *schedule entry*. For example, by configuring a queuing command (e.g. **bandwidth 10 Mbps**) you are setting the schedule entry.

The schedule entry also stores the internal state like the last time a packet was transmitted from that queue and the current packet handle, if any, from that queue.

Two types of schedule entries include the following: Priority Queues, on page 368, and Bandwidth Queues, on page 376.

How Schedule Entries are Programmed

In this section we provide a brief introduction to the parameters that are configured within a schedule entry. The actual commands will be covered in greater detail later in this chapter.

Firstly, a schedule entry is configured as either a *priority entry* or *bandwidth entry* (*priority queue* or *bandwidth queue*).

In the descriptions that follow you will see that priority entries can be further divided into *P1 entries* or *P2 entries*. You configure a P1 entry (the default) with either the **priority** or **priority level 1** command. Similarly, you use the **priority level 2** command to configure a P2 entry.

A bandwidth entry has three distinct parameters: *minimum rate* (Min), *maximum rate* (Max) and *excess weight* (drawn as "Ex" in illustrations).

Note The scheduler for a ASR 1000 Series Aggregation Services Router is often described as a three-parameter scheduler.

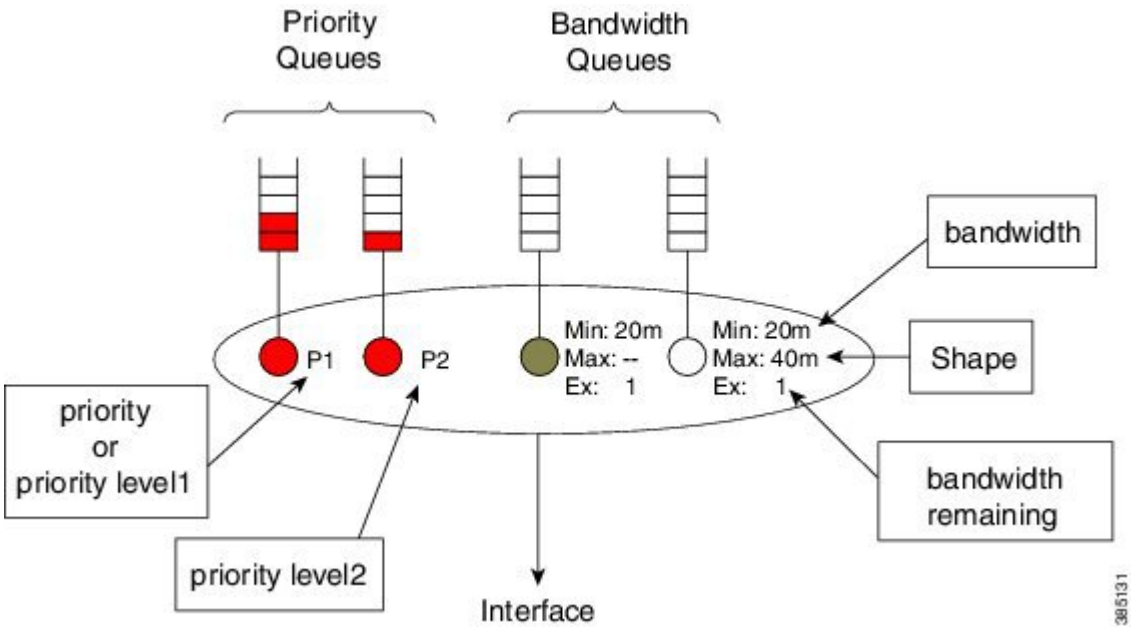
The **Min (minimum rate)** entry allocates a minimum bvandwidth guaranteed amount of throughput to a queue. The Min entry is configured with the **bandwidth** command and is not set unless explicitly configured. IOS configuration checking attempts to ensure that a schedule will always have sufficient bandwidth to honor any configured Min rates. Servicing queues based on monitoring throughput vs. a preconfigured target rate is sometimes referred to as *real time scheduling* (refer to Scheduler's Representation of Time, on page 385).

The **Max (maximum rate)** entry establishes a ceiling on the amount of throughput a queue can receive. The Max entry is configured using the **shape** command and is not set unless explicitly configured. Understand that Max sets a ceiling on the throughput of a queue but does not in itself guarantee any throughput to that queue.

The **Ex (excess weight)** entry mandates how queues will compete for any bandwidth available after Priority and Min guarantees have been met (*excess bandwidth*, or available bandwidth that is not guaranteed to, or not used by, priority and bandwidth guarantees). We configure Excess Weight with the **bandwidth remaining** command and unless explicitly configured, it defaults to 1. Excess *bandwidth sharing* is proportional to a queue's Excess Weight (sometimes referred to as *virtual time scheduling*, because no rates are configured and relative behavior alone is significant). For reflections on bandwidth sharing, see How Schedule Entries are Programmed, on page 359.

The following diagram summarizes what is presented above (the commands to set each schedule entry).

Figure 14: IOS Commands to Set Schedule Entries



Schedule Operation

How a schedule determines the packet sequence may be summarized as follows:

Note After each packet is forwarded, we return to step 1.

1. If the P1 queue is not empty, send the P1 packets.
2. If the PI queue is empty but the P2 queue is not, send the P2 packets.
3. Provided all priority queues are empty, the schedule services any queues with a minimum bandwidth guarantee (Min) and continues to service such queues until the guarantees are met. To ensure fairness, the scheduler will pick between queues with minimum guarantees by selecting the eligible queue, a queue that has not exceeded the bandwidth guarantee and has been waiting longest.
4. What if priority queues are empty and all bandwidth guarantees have been satisfied? Any excess bandwidth is distributed between queues that still require service until either all bandwidth is exhausted or a given queue has reached a maximum configured bandwidth. The Ex configured in that queue's schedule entry, dictates the share each queue will receive of this excess bandwidth.

Schedule Operation: Without a Shaper

The following example illustrates how a schedule operates and how it determines the bandwidth each queue will receive for a given offered load.

Before diving into the example we need to introduce the concept of *priority queue admission control*. In the previous description of schedule operation you will notice an absence of rates regarding how the schedule deals with priority queues; the schedule simply selects the priority queue whenever it contains a packet.

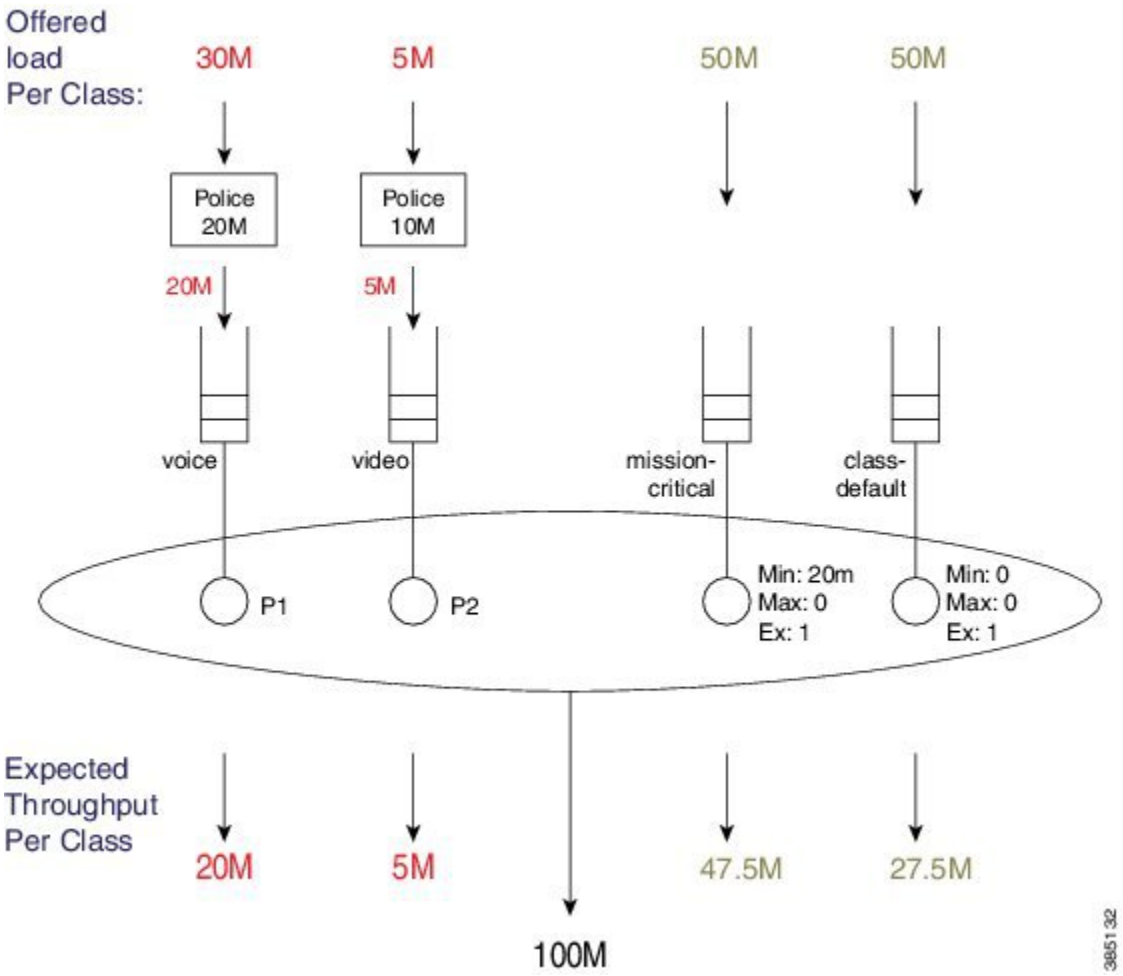
To prevent a priority queue (class) from starving other queues of service, we can use a policer to limit the consumable bandwidth. Such a policer restricts the rate at which packets can be enqueued in that queue.

In the following example, we attach a policy to a 100 Mbps interface:

```
policy-map scheduling-example
  class voice
    priority level 1
    police 20m
  class video
    priority level 2
    police 10m
  class mission-critical
    bandwidth 20000
  class class-default
```

Note Bandwidth is configured in Kbps. While **police** and **shape** commands support a postfix to specify the unit, the **bandwidth** command does not.

Figure 15: Scheduling Operation



The loads offered to each class are shown at the top of the figure: 30M, 5M, 50M, and 50M. We have applied policers (20M and 10M) to the priority queues.

30 Mbps is offered to the voice class, which first traverses a 20 Mbps policer, enqueuing 20 Mbps to the P1 queue. Because we always service this queue first, all 20 Mbps enqueued will be forwarded.

5 Mbps is offered to the video class (which all transits the 10 Mbps policer) and 5 Mbps is enqueued to the video queue. As 80 Mbps (100 Mbps - 20 Mbps) bandwidth is still available, all 5 Mbps will be forwarded.

After servicing priority queues, we advance to any queues with an explicit Min bandwidth guarantee. The mission-critical class has a Min of 20 Mbps so it will receive at least that amount of throughput.

The available excess bandwidth is 55Mbps (100 Mbps - 20 Mbps - 5 Mbps - 20 Mbps). Both the class-default and mission-critical classes have default excess weights of 1, so each receives an equal share of the available bandwidth, (55Mbps/2 =) 27.5 Mbps.

The mission-critical class will observe a total throughput of 47.5 Mbps (20 Mbps + 27.5 Mbps).

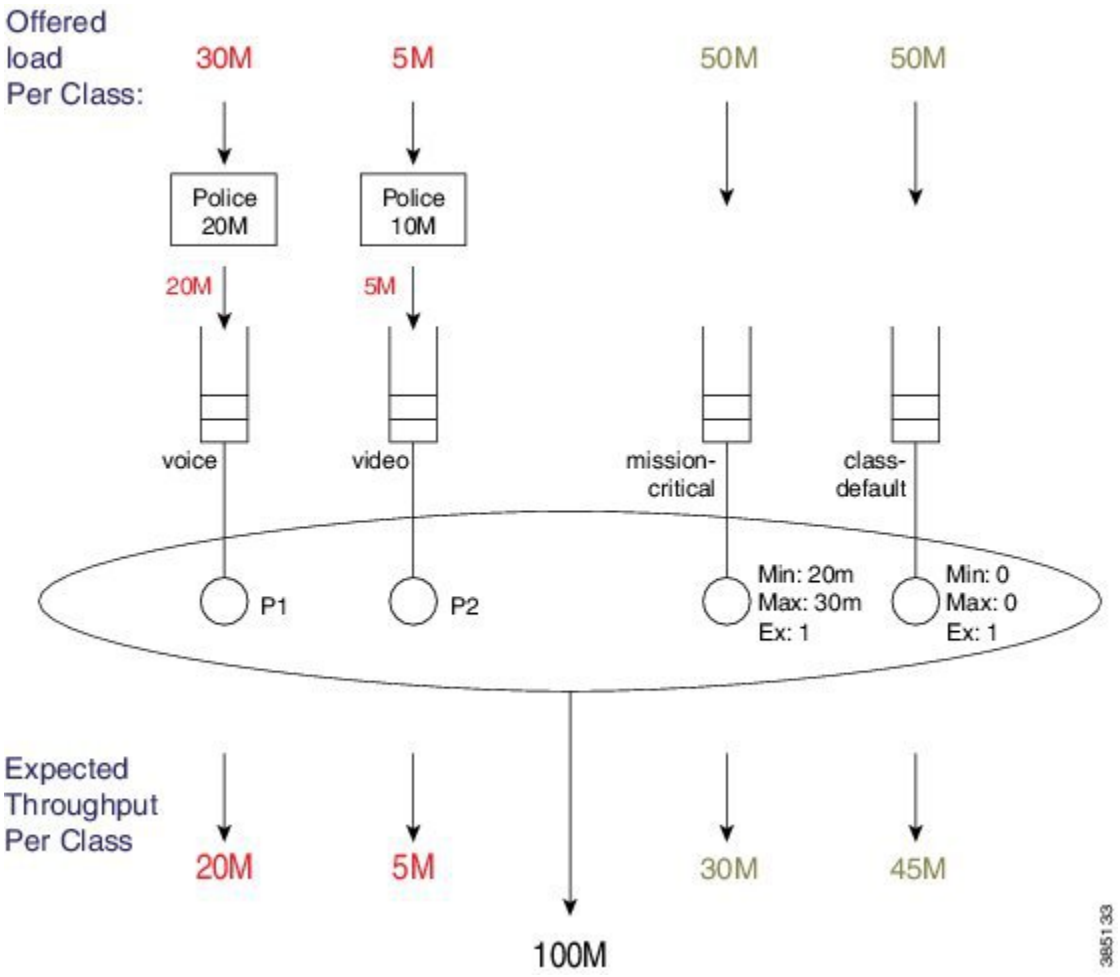
Schedule Operation: With a Shaper

Let's modify the configuration slightly - we will add a Max value (configure a shaper) to the mission-critical class:

```
policy-map scheduling-example
  class voice
    priority level 1
    police 20m
  class video
    priority level 2
    police 10m
  class mission-critical
    bandwidth 20000
    shape average 30m
```

Note We excluded class-default in the policy definition - it is always there whether or not we explicitly define it.

Figure 16: Scheduling Operation



The loads offered to each class is exactly as before: 30M, 5M, 50M and 50M.

30 Mbps is offered to the voice class, which first passes through a 20 Mbps policer, enqueing 20 Mbps to the P1 queue. We always service this queue first, so all 20 Mbps enqueued will be forwarded.

5 Mbps is offered to the video class (which all passes through the 10 Mbps policer) and 5 Mbps is enqueued to the P2 queue. As 80 Mbps (100 Mbps - 20 Mbps) bandwidth is still available, all 5 Mbps will be forwarded.

After servicing priority queues, we advance to any queues with an explicit Min. The mission-critical class has a bandwidth guarantee of 20 Mbps so it will receive at least that amount of throughput.

The available excess bandwidth is 55 Mbps (100 - 20 - 5 - 20 Mbps). Both the class-default and mission-critical classes have default Ex's 1, so each receives an equal share of the available bandwidth. From the Excess bandwidth sharing "rule," where in bandwidth is proportional to a queue's Ex, each class receives a 27.5 Mbps share. (For more information on this "rule," refer to How Schedule Entries are Programmed, on page 359.)

Based on the bandwidth guarantee and bandwidth sharing, the mission-critical queue would receive 47.5 Mbps (20 + 27.5 Mbps). However, the queue cannot use this much bandwidth because the Max configured shape rate is set to 30 Mbps (recall that Max is set to 0 in the previous example). Consequently, the queue uses 30 Mbps (out of the 47.5 Mbps received from bandwidth sharing) and the additional 17.5 Mbps of bandwidth returns to the excess pool.

As class-default is the only queue still requesting bandwidth, it has no competition and can consume this extra 17.5 Mbps, increasing its total throughput to 45 Mbps.

Note This example demonstrates how bandwidth is never wasted - scheduling will continue to sort through eligible queues and apportion bandwidth until one of the following applies:

- Each queue is empty.
- All Max values have been reached.
- All bandwidth has been consumed.

Configuring Rates and Burst Parameters

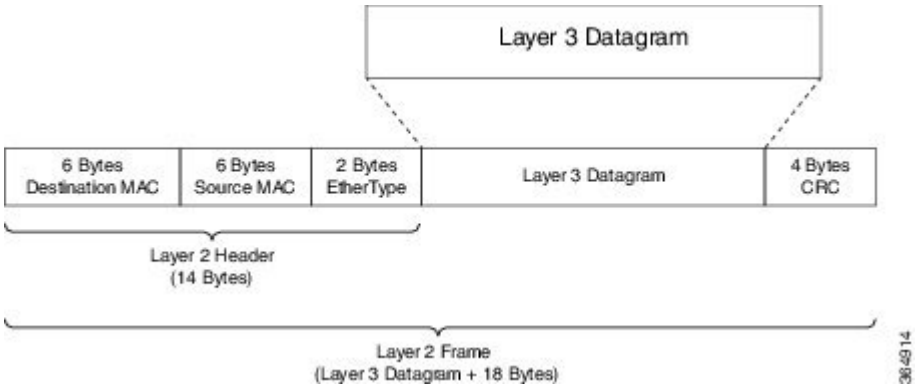
What's Included in Scheduling Rate Calculations (Overhead Accounting)

In the discussion of Schedule Operation, you will notice that Min and Max are configured in bits per second. But what do these rates include? The short answer is that a schedule includes the Layer 3 datagram and Layer 2 header lengths but neither CRC nor inter-packet overhead.

Layer 3 Datagram

To clarify, let's imagine transporting an IP datagram over a GigabitEthernet link.

Note Henceforward, we will refer to a "schedule's perception of the packet length" as the *scheduling length*.

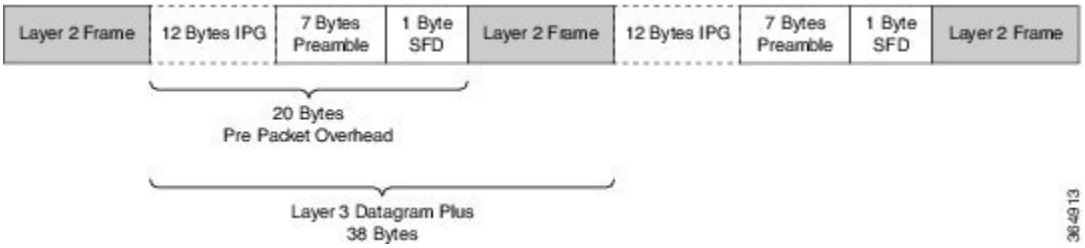


Ethernet Overhead

To transport the datagram on a GigabitEthernet link we first need to encapsulate it correctly in an Ethernet frame. This process adds 14 bytes of Layer 2 header and an additional 4 bytes of CRC (i.e., total of 18 bytes for encapsulation).

Consider what happens when this Layer 2 frame is transmitted over the physical medium. Ethernet requires a minimum *inter packet gap* (IPG) equal to the transmit time for 12 bytes of data, 7 bytes of preamble, and a *single-byte start-of-frame delimiter* (SFD), for a total pre-packet overhead of 20 bytes:

Figure 17: Ethernet Overhead



So, if you send multiple Ethernet frames sequentially, the total per-packet overhead for each Layer 3 datagram is an additional 38 bytes (encapsulation (18 bytes) + Ethernet inter-packet overhead (20 bytes)). For example, if you were to send 100 byte IP datagrams at line rate on a GigabitEthernet link, the expected throughput in packets per second would be:

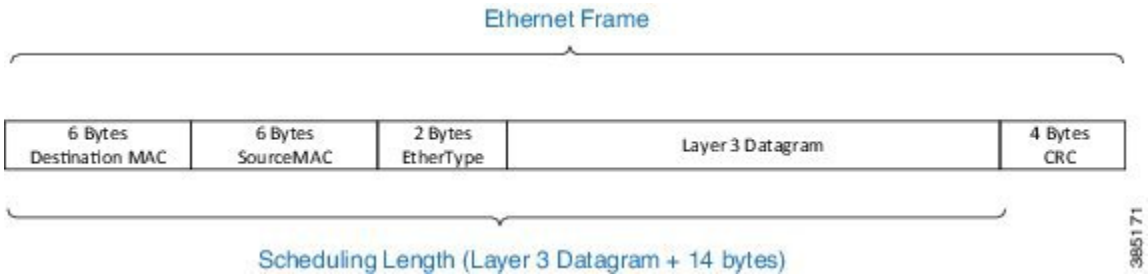
```
Linerate / Bits Per Byte / ( Layer 3 length + Per Packet Overhead) = Packets Per Second
```

```
1 Gbps / 8 / ( 100 + 38 ) = 905, 797 pps
```

Scheduling Length

From the scheduler's viewpoint, the packet's length is Layer 3 datagram + Layer 2 header (14 bytes on a GigabitEthernet interface):

Figure 18: Scheduling Length



Now consider a 500-Mbps shaper configured on a GigabitEthernet interface. (Recall that shaping is the process of imposing a maximum rate of traffic while regulating the traffic rate in such a way that downstream devices are not subjected to congestion.) As in the previous example, we will send all 100-byte IP datagrams to the scheduler, resulting in a "scheduling length" of 114 bytes (100 byte (datagram) + 14 byte (Ethernet Layer 2 header)). According to the following formula, the anticipated throughput would now be:

```
  Shaper Rate / Bits per Byte / (Layer 3 length + Layer 2 header length)
= Packets Per Second
```

```
  500 Mbps / 8 / (100 + 14) = 548,246 pps
```

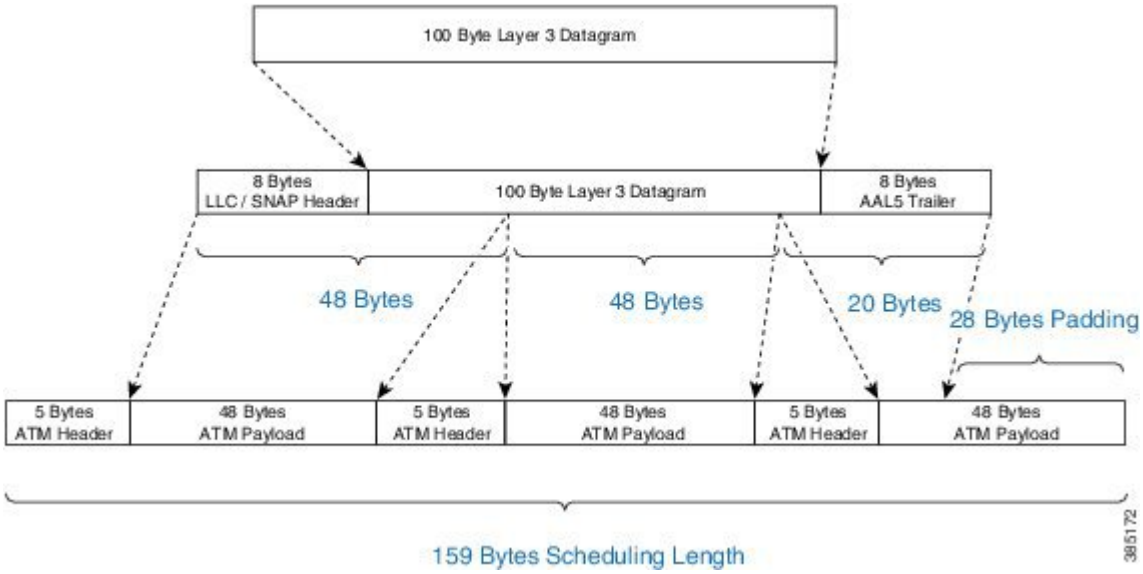
Observe that 100% of linerate (all 100 byte datagrams) was 905,797 packets per second but shaping to 500 Mbps (all 100 byte datagrams) yielded a throughput of 548,246 packets per second. Obviously, this is considerably more than 50% of physical capacity. When specifying rates to apportion bandwidth, be aware that rates do not include all overhead required to transport that packet.

Scheduler on an ATM Interface

When a queuing policy (scheduling policy) is attached to an ATM VC, scheduling rates in that policy are inclusive of all cell tax. This differs from a policer configured in such a policy that only includes the AAL5 header.

For example, consider a 100-byte datagram sent over an ATM VC configured with AAL5 SNAP encapsulation. A router will add an 8-byte LLC/SNAP header to the datagram, yielding a *policing length* of 108 bytes (analogous to a scheduling length of 114 bytes for a 100 byte IP datagram. (See What's Included in Scheduling Rate Calculations (Overhead Accounting), on page 364.) (For further details on policing length, refer to Priority Policing Length, on page 374.)

Figure 19: Scheduler on ATM Interface



To convey the packet, the router must also add an 8-byte AAL5 trailer (to the policing length) and then spilt the packet into ATM cells. To transport this packet we require 3 ATM cells, each carrying 48 bytes of the packet. We pad the third cell such that it also has a 48-byte payload.

Each of these 3 cells is 53 bytes in length (48 bytes packet + 5 byte ATM Header), which means that the scheduling length of the 100-byte datagram would be 159 bytes (3 cells x 53 bytes per cell).

Scheduler on a Logical Interface

In the Policing chapter we discuss how policer overhead accounting may differ depending on whether the policy is attached to a physical or logical interface (see Policer on Logical Interface, on page 596). This situation does not apply to a scheduler. Although a policy is attached to a logical interface (a tunnel interface) we must complete all processing and add any necessary headers before we enqueue the packet to egress a physical interface. Because we know the final length of the packet at the time of enqueue, we can set the scheduling length accordingly at that time.

Scheduler Overhead Accounting Adjustment

In prior sections, we described what is included by default in scheduler rate calculations. Occasionally, however, a user might want behavior to differ from the default.

For example, we hear that users want to express rates as physical bandwidth that would be consumed on the link. For an Ethernet interface, you would need to include the 4-byte CRC and 20-bytes inter-packet overhead required by each packet.

We also hear from service providers who want to charge their customers for traffic throughput at Layer 3 rates. The datagram's length remains constant as a packet traverses different interface types or encapsulating protocols, making it easier for users to understand. In this instance, we would not include the Layer 2 header length in shape rate calculations.

Note Changing overhead accounting may impact the network elsewhere. For example, if we use a policer for network admission control, we typically configure a shaper on customer premises equipment connecting to that network. The shaper and policer should have the same view of what is included in CIR.

Scheduler Account Option

The scheduler account option (the **account** keyword) allows you to specify a number of bytes that should be added or removed from the default "scheduling length" per packet to achieve the desired behavior. You can add or subtract at most 63 bytes per packet. This option is supported on the **shape** and **bandwidth** commands.

In the following example, we apply a shaper on an Ethernet interface and we want to include all overhead such that the shaper will cap throughput at 50% of the actual physical bandwidth. By adding 24 bytes per packet we "cover" the 4 byte CRC and 20-bytes inter-packet overhead:

```
policy-map ethernet-physical-example
  class class-default
    shape average percent 50 account user-defined 24
```

Note With overhead accounting, we must account for *hierarchical policies*. If a parent shaper is configured with the account option, any child shapers or bandwidth guarantees will also inherit the same adjustment as specified in the parent policy.

In the chapter on hierarchical scheduling, we will observe how to use shapers to condition traffic for remote links and to use child polices for apportioning bandwidth within that shape rate. In that use case, the encapsulation on the remote link may differ from the encapsulation on the sending device (e.g. an enterprise hub router connected to the network with an Ethernet interface sends traffic to a branch connected with a T1 interface). If the T1 link were using HDLC encapsulation each datagram would have 4 bytes of Layer 2 headers on that link. On the Ethernet, however, each packet would have 14 bytes of Layer 2 headers. The account option can be used to shape and schedule packets as they would appear on that remote link. That is, remove 14 bytes from the scheduling length as Ethernet headers are no longer present and then add 4 bytes to the scheduling length to represent the HDLC Layer 2 overhead.

Overhead Accounting Adjustment (Predefined Options)

In addition to specifying a number of bytes to add or subtract (see the following table), the CLI also offers some predefined options with which you can specify remote encapsulation. The current predefined options are based on broadband use cases, assuming that we send (or receive) traffic on an Ethernet interface to a DSLAM elsewhere in the network. Although we are encapsulating in Ethernet frames that include Dot1Q or Q-in-Q, the DSLAM receives some form of ATM encapsulation. We want the shaper to condition traffic to mirror how it would appear after DSLAM. In each case we would also add cell-tax to the scheduling length.

Imagine that we forwarding Dot1Q encapsulated packets on an Ethernet interface. Imagine further than a downstream DSLAM will:

- receive the packets
- strip the Ethernet and Dot1q headers
- perform AAL5-Mux 1483 routed encapsulation.

Referring to the previous table, DSLAM will remove 18 bytes of Ethernet/Dot1q and add a 3-byte LLC header, generating a -3 bytes change in the scheduling length. (For a schematic, refer to What's Included in Scheduling Rate Calculations (Overhead Accounting), on page 364.)

As the DSLAM is sending over an ATM network, it would add an 8-byte AAL trailer and then split the resulting PDU into 53-byte cells. The ATM value "yes" (with reference to the table) indicates that the router will calculate this cell tax and include that extra overhead in the scheduling length.

```
policy-map atm-example
  class class-default
    shape average 50m account dot1q aal5 mux-1483routed
```

Example - Predefined Overhead Accounting

Imagine we are forwarding Dot1Q encapsulated packets on an Ethernet interface. Imagine further than a downstream DSLAM will:

- receive the packets
- strip the Ethernet and Dot1q headers
- perform AAL5-Mux 1483 routed encapsulation.

Referring to the previous table, DSLAM will remove 18 bytes of Ethernet/Dot1q and add a 3-byte LLC header, generating a -3 bytes change in the scheduling length. (For a schematic, refer to What's Included in Scheduling Rate Calculations (Overhead Accounting), on page 364.)

As the DSLAM is sending over an ATM network, it would add an 8-byte AAL trailer and then split the resulting PDU into 53-byte cells. The ATM value "yes" (with reference to the table) indicates that the router will calculate this cell tax and include that extra overhead in the scheduling length.

```
policy-map atm-example
  class class-default
    shape average 50m account dot1q aal5 mux-1483routed
```

Priority Queues

Priority queues represents a type of schedule entries that enable you to avoid any unnecessary delay in forwarding packets. Through the priority semantic, we can guarantee low latency treatment for applications that are latency and (or) jitter sensitive. As an example, consider Voice over IP (VOIP). Typical VOIP phones have a 30 mS *de-jitter buffer*, allowing them to tolerate a maximum of 30 mS jitter end-to-end across the network.

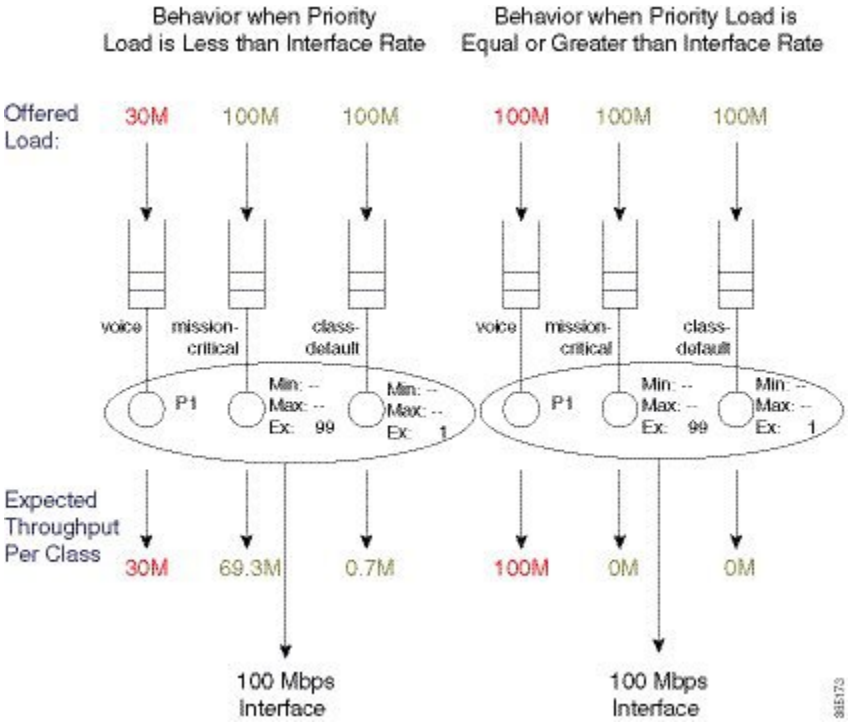
When you configure a priority queue, you can select one of three ways to control the bandwidth that might be consumed by that traffic class: unconstrained priority queue, conditional policer, or (un-conditional) always-on policer.

Unconstrained Priority Queue

One way to control bandwidth consumption is with an *unconstrained priority queue* (absolute priority queue), which is a priority queue configured without any limit on the amount of bandwidth that may be consumed by the priority class. To illustrate, consider this example configuration as well as the configured schedule entries in the following figure. Note the lack of a policer for admission control to the priority queue.

```
policy-map absolute_pq_example
  class voice
    priority
  class mission-critical
    bandwidth remaining percent 99
  class class-default
    bandwidth remaining percent 1
```

Figure 20: Unconstrained Priority Queue



In the example on the left, the actual interface bandwidth capacity (100 Mbps) exceeds the load to the priority queue of the voice class (30 Mbps). This leaves 70 Mbps of excess bandwidth to apportion based on the excess weight (Ex) ratio (99:1; set by the **bandwidth remaining** command).

In the example on the right, the priority load has been increased to 100 Mbps. Because this leaves no excess bandwidth, other queues are starved of service (an expected throughput of 0M).

The take-home is that without admission control on a priority class, a class might consume the entire interface bandwidth and so starve all other service queues. This can cause mission-critical applications to suffer. Moreover, if control messages are in the starved service queue, network instability might result.

So, use unconstrained priority queues with caution. To ensure the priority queue is unable to starve others of service, you might want to consider using alternative bandwidth control systems like Call Admission Control (CAC).

Note You cannot use minimum bandwidth guarantees (as set by the **bandwidth** command) in conjunction with unconstrained priority queues. If the priority queue is capable of consuming all available bandwidth it follows that you can't guarantee any of that bandwidth to other classes. IOS will reject any such configurations.

Priority Queue with Conditional Policer

Another way to control bandwidth consumption is to enter a value with the **priority** command. This represents a way to handle queue admission control with a conditional policer.

Conditional priority rate limits traffic with a policer only if congestion exists at the parent (policy-map or physical interface) level. This state exists provided more than the configured maximum rate of traffic attempts to move through the class (and/or interface).

The key element is that a conditional policer will only drop packets if the schedule is congested. That is, it will only drop packets when the offered load exceeds the available bandwidth (interface bandwidth in the context of a flat policy-map attached to a physical interface).

A conditional priority class can use more than its configured rate, but only if contention with other classes in the same policy is absent.

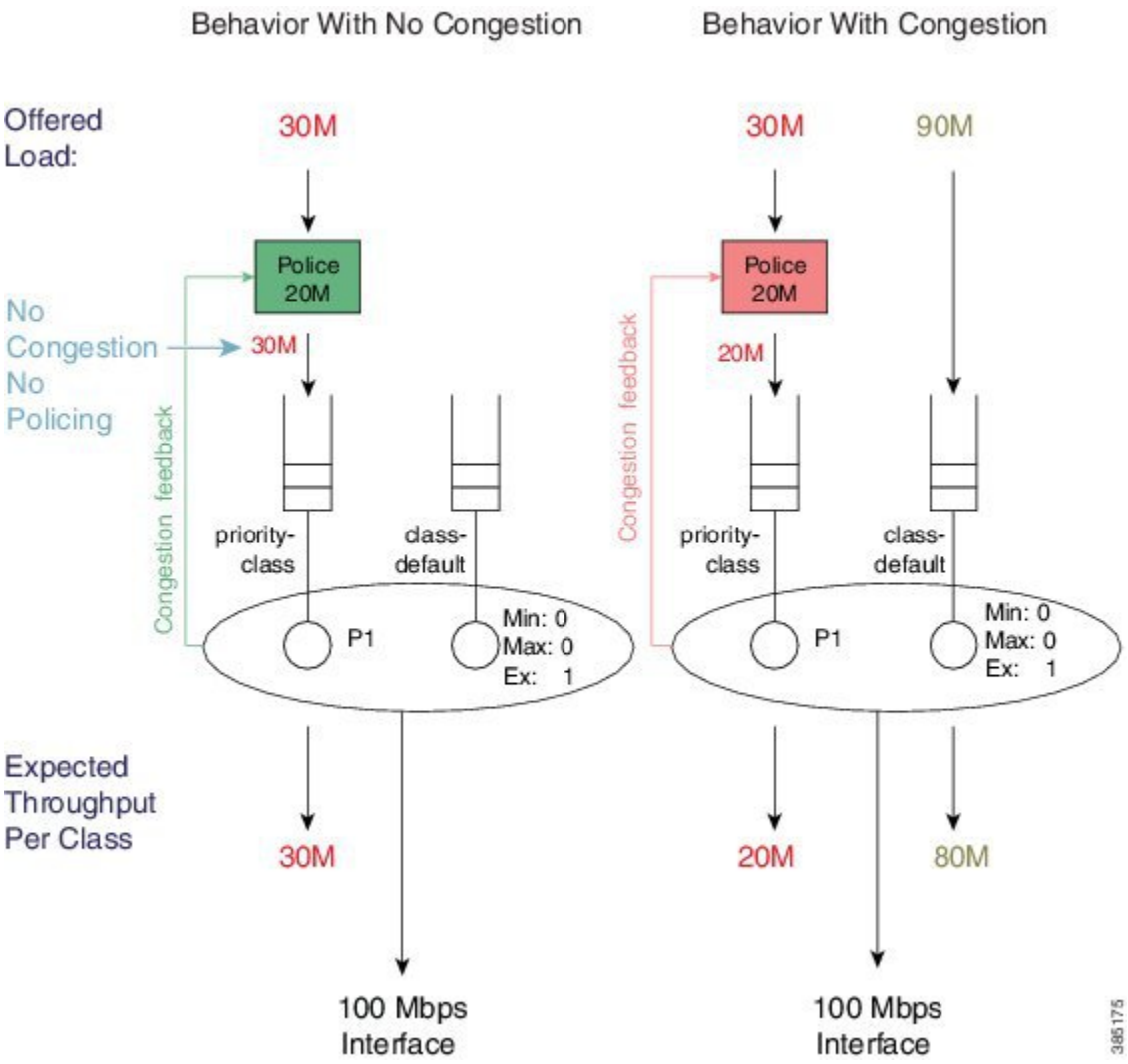
The schedule provides congestion feedback to the policer, which will count every packet in the rate it observes but will suppress the drop action unless congestion exists. So, if no congestion exists, the priority queue may consume whatever bandwidth is available but when congestion occurs the queue will be policed to the configured rate.

```
policy-map conditional_policer_example
  class priority-class
    priority 20000
```

The priority value is configured in kps and although configured with the **priority** command, it does not alter a schedule entry. (Recall that a schedule entry is where we store a queue's expected treatment.) Instead, it configures a policer that will be executed before packets may be enqueued.

The following diagram shows how a conditional policer would function with and without congestion. In this example, we have attached the previous configuration to a 100 Mbps interface.

Figure 21: Priority Queue with Conditional Policer



In the schedule depicted on the left no congestion exists. As the congestion feedback reports no congestion and the policer will enqueue the entire 30 Mbps offered to that class.

In the schedule depicted on the right where congestion exists, the policer will enforce the 20 Mbps rate configured with the **priority** command.

Be aware that a conditional policer has advantages and disadvantages:

Advantage	A priority class may use all bandwidth not currently used by other classes.

Disadvantage	You cannot carefully plan for priority capacity throughout your network if you don't know the forwarding-capacity of a particular interface. A *true priority service* should have low latency (no queue build-up) and no drops, end-to-end. You experience inconsistent behavior depending on whether or not an interface is congested. If you under-provision the police rate, you may observe intermittent problems with applications using that class and diagnosing the issue might be very difficult.

Note You cannot use conditional policers and policer overhead accounting adjustment concurrently.

Priority Queue with Always on (Unconditional) Policer

The third way to control bandwidth consumption is to use an explicit always-on (i.e., unconditional) policer for queue admission control.

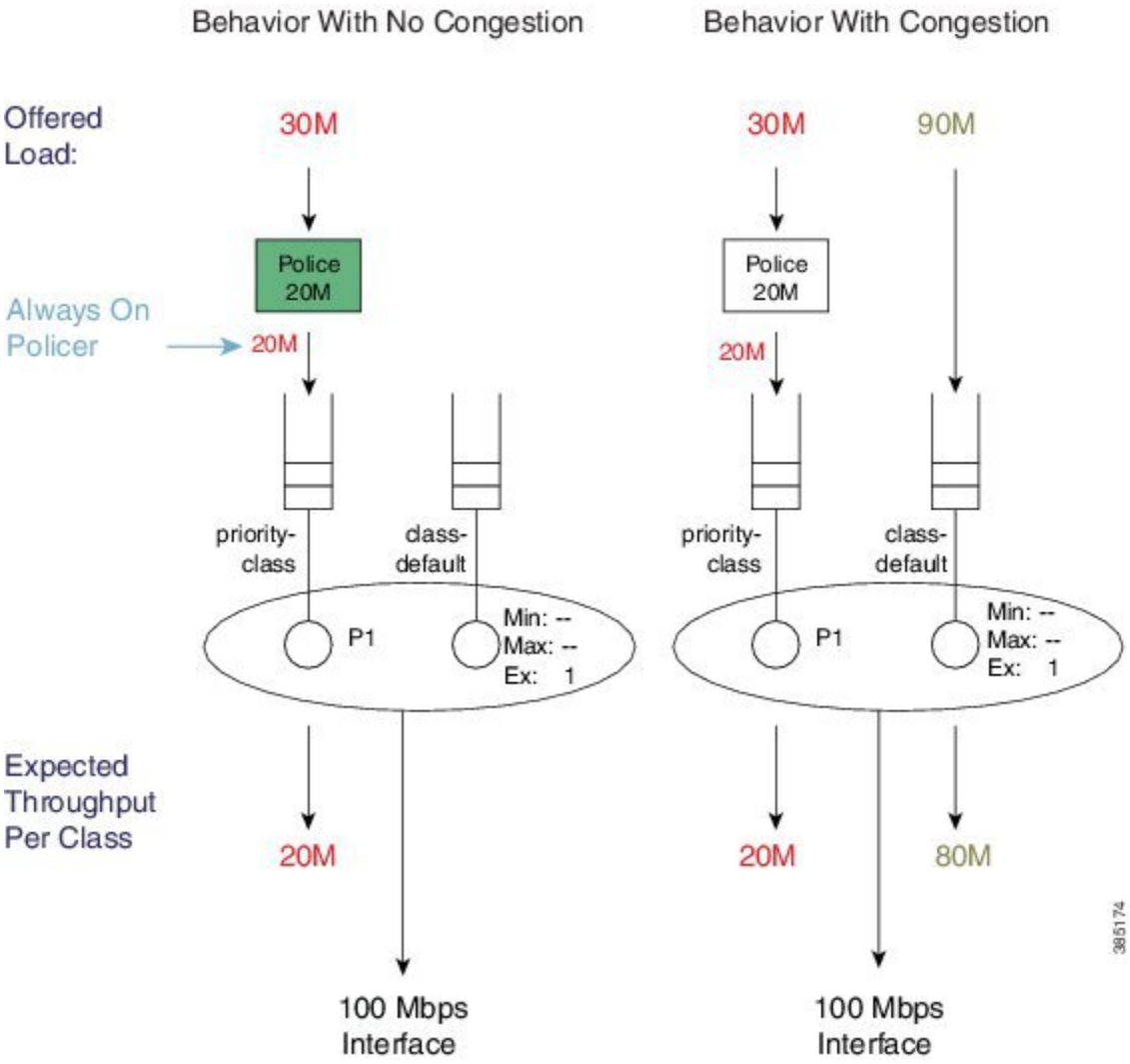
When you configure a priority class with an explicit policing rate, traffic is limited to the policer rate regardless of congestion conditions. That is, even if bandwith is available, the priority traffic cannot exceed the explicit rate.

The following example shows how such a configuration might look:

```
policy-map always_on_policer
  class priority-class
    priority
    police cir 20m
```

The diagram below shows the behavior of such a policer.

Figure 22: Always on Policer



When you configure a priority class with an explicit policing rate, this rate is always enforced. That is, even with sufficient bandwith, priority traffic cannot exceed the explicit rate. This means that you have *deterministic behavior* in your priority service. If a user complains of poor application performance you can look for policer drops in your network and determine if insufficient bandwidth is allocated to the priority service. Applications should have the same experience regardless of whether or not congestion exists.

Priority Queue Burst Considerations

In previous sections we describe how to perform queue admission control for the priority queue using a conditional or always-on policer. Specifically we employ a *single-rate two-color policer*. From the policing chapter we know that policers are implemented using a token bucket scheme that allows for some burst (Single-Rate, Two-Color Policer, on page 591). Controlling this (burst) allowance is crucial when you use policers in this way.

Bursts that are allowed by the policer may result in a build-up of the priority queue, which will generate latency for a packet that is added to the end of that queue. The packet must wait for the transmission of all

preceding packets in the queue before it too can be transmitted. The amount of latency depends on the *serialization delay*, the time taken to transmit those packets on the physical medium.

As multiple packets from a given flow arrive, they may experience different conditions in the priority queue. One might arrive and be enqueued behind a number of packets already waiting and thus experience some latency. The next packet in that same flow may arrive to an empty priority queue and be scheduled immediately. What this means is that any potential latency from priority queue congestion is also potential *jitter* (jitter, potential the variation in the latency of received packets).

The *default burst allowance* for policers in IOS is set at 250 mS for always on policers and 200 mS for conditional policers. If a policer can allow us to enqueue a burst, it follows that these numbers can be almost directly translated into potential jitter. In the introduction to the priority semantic (see Priority Queues, on page 368), we indicated that voice applications can typically tolerate about 30 mS of jitter and 150 mS latency end-to-end across a network. Given the former, we usually try to apportion some of this budget to each node in the network. A simple guideline is to allow a burst tolerance (and thereby potential jitter) of 5-10 mS on any single node.

For example, envisage a priority queue configured with a queue-admission policer at a rate of 2 Mbps and a burst allowance of 5 mS. Calculate the number of bytes we can transmit in 5 mS:

Burst Target
= Police Rate / 8 Bites per Byte * 5 mS
= 2 Mbps / 8 * .005 = 1250 bytes

For an always on policer, the configuration for this example would look like:

```
policy-map always_on_policer_burst_example
  class voice
  priority
  police cir 2000000 1250
```

For a conditional policer, the configuration example would look like:

```
policy-map conditional_policer_burst_example
  class voice
  priority 20000 1250
```

Priority Policing Length

In the section What's Included in Scheduling Rate Calculations (Overhead Accounting), on page 364 we introduced the concept of scheduling length, which is how a scheduler "views" packet length when it is evaluating conformance to a rate. In the Policing chapter we also introduced the similar concept of policing length (What's Included in the Policer-Rate Calculation (Overhead Accounting), on page 595). As the rate configured on a priority queue is a policing rate, we will use the policing length when determining conformance to that rate. When a policy is attached to a physical interface, as described in this chapter, the policing and scheduling lengths are identical. To alter the policing length, you can use the policer overhead accounting feature.

Note The **account** keyword is supported with always-on policers but not conditional policers.

Multi-Level Priority Queuing

The Multi-Level Priority Queues (MPQ) feature allows you to configure multiple priority queues for multiple traffic classes by specifying a different priority level for each of the traffic classes in a single service policy-map.

In Schedule Operation, on page 360, we introduced that a priority queue may be P1 or P2. The original intent of this feature was to support voice and video in separate priority queues as each has differing traffic characteristics and jitter tolerance. In particular, voice has a smaller packet size (typically around 80 bytes for voice compared to 1400 bytes for video) and tighter jitter requirements (typically 30 mS whereas non-interactive video may be 100's of mS). So, we would use a P1 queue for voice and a P2 queue for video traffic.

Today many video applications use advanced adaptive codecs and separate the voice and video content into separate streams. Some argue that video traffic is now TCP-like in its behavior and does better in bandwidth queues. Interactive video on slower links may still require a P2 queue.

To configure multilevel priority queuing you must use the **level** keyword in the **priority** command. This feature is supported with conditional policers, always-on policers and absolute priority queues.

Here is an example of a multilevel priority queue with conditional policers:

```
policy-map multilevel-example2
  class voice
    priority level 1 5000 3125
  class video
    priority level 2 10000 12500
```

Here is an example of a multilevel priority queue configuration with always-on policers:

```
policy-map multilevel-example1
  class voice
    priority level 1
    police cir 5000000 3125
  class video
    priority level 2
    police cir 10000000 12500
```

Note If you do not explicitly configure a level, a priority queue will operate as a P1 queue. However, if you want to configure multilevel priority queuing, you must explicitly configure levels.

For example, the following configuration would be rejected - you need to explicitly configure the priority level in the voice class:

```
policy-map multilevel-rejection-example
  class voice
    priority
    police cir 5000000 3125
  class video
    priority level 2
    police cir 10000000 12500
```

Bandwidth Queues

Bandwidth queues enable you to apportion interface bandwidth for applications that lack strict latency requirements. Recall that the intent of scheduling is to ensure that all applications receive the necessary bandwidth and to utilize unused bandwidth by making it available for other applications.

You can reflect on *bandwidth sharing* as follows:

Guarantee bandwidth for applications so that they operate effectively. For example, you may decide that your email application is business critical and must continue to operate even during network congestion. If so, you would want to always guarantee some amount of available bandwidth to your business critical applications.

Determine which applications to sacrifice under congestion. For instance, you may decide that a social media application is not business critical; employees can use the network for such applications but not at the expense of business critical activities. If so, you can place these applications in a queue that is intentionally deprived of service during congestion.

As described in How Schedule Entries are Programmed, on page 359, bandwidth queue schedule entries have three distinct parameters Min, Max, and Ex set by **bandwidth**, **shape**, and **bandwidth remaining** commands, respectively. Let's take a closer look at these commands.

Bandwidth Command

The **bandwidth** command sets the Minimum bandwidth (Min) guarantee in a schedule entry at which a queue will be serviced. Given the exact bandwidth requirements of an application, this command provides a convenient way to ensure that an application receives exactly what it needs under congestion. Be aware that by default every entry will also have a configured Excess Weight, which can lead to some additional guaranteed service for the queue.

Bandwidth guarantees are configured in Kbit/sec and may be configured in increments of 1 Kbps. You can also configure the guarantee as a percentage of physical linerate: a percentage of the nominal interface rate displayed as bandwidth through the **show interface** command.

For example, the **show interface gigabit x/y/z** command on a GigabitEthernet interface would show a BW of 1000000 Kbit/sec and any percentage value would be a percentage of this nominal rate. So, if we configured **bandwidth percent 50** on a Gigabitethernet interface, it would set a Min value of 500 Mbps.

The **bandwidth** command accepts the **account** keyword, which enables you to adjust what overhead is included in rate conformance calculations. However, any configured **account** value must be consistent across a policy-map (all **bandwidth** and all **shape** commands in the policy-map must be configured with the same account value).

Caution Do no configure extremely low bandwidth guarantees on high speed interfaces.

Recall from How Schedule Entries are Programmed, on page 359, that we service queues with Min bandwidth guarantees before Excess queues. Furthermore, recall from What's Included in Scheduling Rate Calculations (Overhead Accounting), on page 364 that by default the scheduling length of a packet does not include all of the physical bandwidth that will be consumed to transport that packet (it does not include CRC or inter packet overhead). Given these two facts you should be careful not to guarantee more bandwidth than is actually available. Else, you may starve queues possessing only an excess weight of service.

Imagine that you configure a queue with a Min of 98 Mbps and attach the policy to a FastEthernet interface (100 Mbps). If we send all 100 byte frames, the *scheduling length* for each frame would be 96 bytes but the *actual bandwidth* consumed by each (including the required inter- packet overhead) would be 120 bytes.

From a scheduling length perspective, 98 Mbs would translate to 120 Bytes/96 Bytes * 98 Mbps = 122.5 Mbps of physical bandwidth usage:

actual bandwidth/scheduling length * Min = physical bandwidth usage

(100 bytes Frame + 20 bytes Per Packet Overhead)/(100 bytes Frame - 4 bytes CRC) * 98 Mbps = 120 bytes/96 bytes * 98 Mbps = 122.5 Mbps

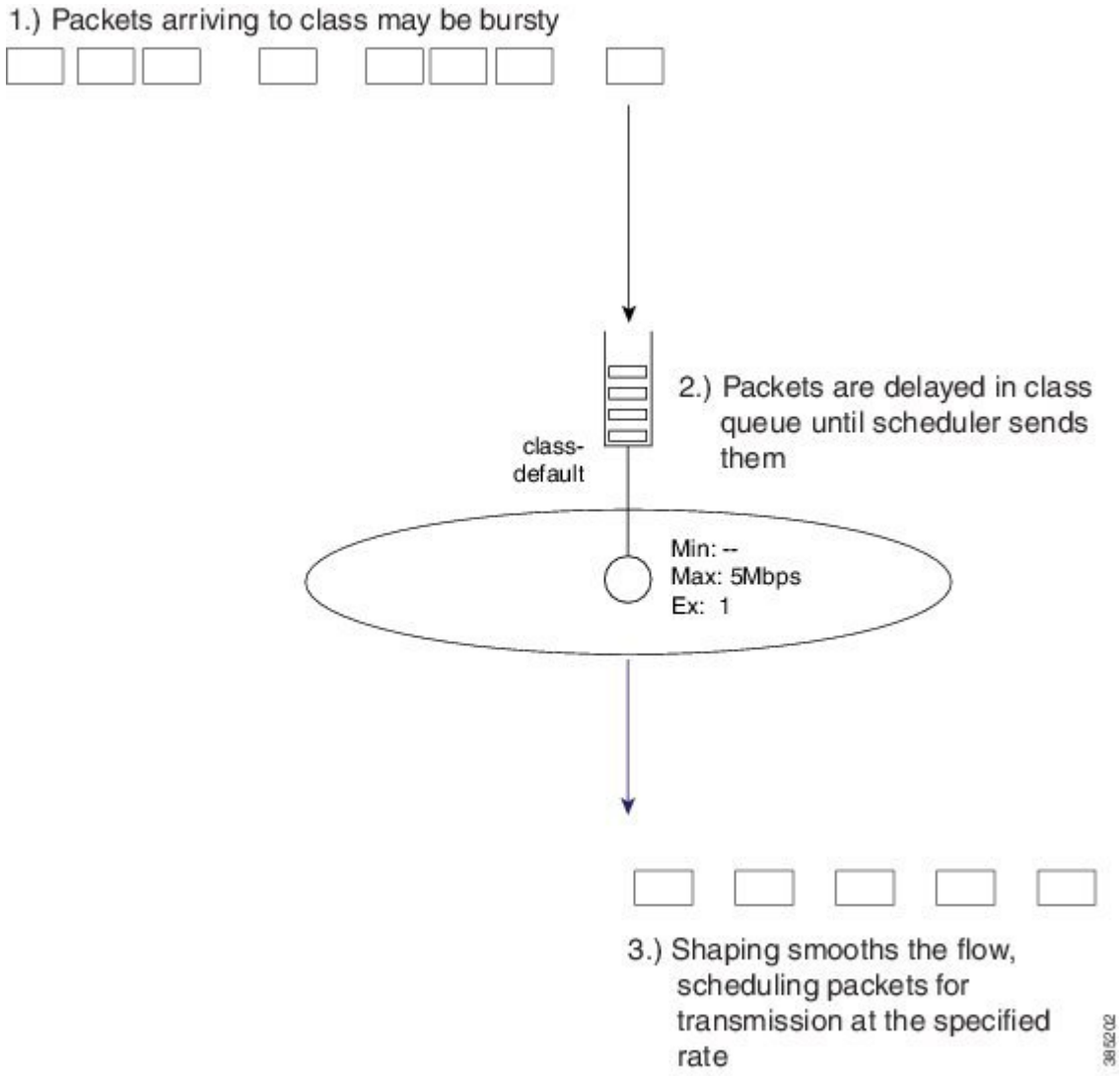
So, we cannot honor our promise! Generally, if the sum of your priority guarantees and Min bandwidth guarantees totals 75% or more of the physical bandwidth, consider whether you are starving other queues of service. In particular consider what might happen to class-default traffic as the default configuration for the class-default schedule entry is to configure an excess weight only.

Shape Command

The **shape** command sets the Max rate in a schedule entry at which a queue will be serviced. Setting the Max rate does not in itself guarantee any throughput to that queue; it simply sets a ceiling. If you create a class containing just the **shape** command it will also receive the default excess weight setting (‘1'), which determines the bandwidth share that class should receive.

Shaping is most commonly used in hierarchical policies. Occasionally, however, you might want to use shaping in flat policies. That is, you may have adaptive video in a bandwidth class that if unconstrained could expand its bandwidth usage to beyond what is physically available. Consequently, you might want to employ shaping to limit the expansion of that flow.

Figure 23: Single-shaped Queue



The diagram above shows an example of a single-shaped queue. For this simple example, the configuration would look as follows:

```
policy-map shape_example
  class class-default
    shape average 5m
```

As packets arrive they are added to the end of the queue for that class. The scheduler is pulling packets from the head of the queue at the specified rate. If the *arrival rate* (rate at which packets are arriving at the queue) exceeds the *service rate* (the rate at which packets are pulled from the queue) then packets will be delayed and must sit in the queue until all preceding packets are sent. In this simple example no other queues compete for bandwidth, so the service rate will equal the shape rate (5 Mbps).

From this simple example you can see that a shaper will "smooth" a stream. Typically, it will be a few small packets rather than a single packet released by the scheduler. The net result is as shown, a shaper meters the rate at which packets are forwarded.

Shape Average

The **shape average** command is the primary means of configuring a Max rate for a class.

You can configure the rate in bits per second or as a percentage of the interface (or parent shaper) rate. As with other scheduling commands, you can adjust the overhead included in scheduling calculations with the **account** keyword.

As an example, we can modify the previous configuration snippet to include CRC and inter-packet overhead on an Ethernet interface as follows (for more details see What's Included in the Policer-Rate Calculation (Overhead Accounting), on page 595):

```
policy-map shape_example
  class class-default
    shape average 5m account user-defined 24
```

Note The **shape average** command-line interface also includes options for Bc (bits per interval, sustained or committed) and Be (bits per interval, excess). (These options are remnants from the software implementation of shaping in IOS classic and have no effect on an ASR 1000 Series Aggregation Services Routers.)

On software implementations the processing overhead meant it was only feasible to perform the math involved in scheduling at some predetermined interval, typically a number of milliseconds.

Adjusting Bc was a way to further reduce scheduling frequency (and thereby processing overhead) at the expense of more burstiness in forwarded traffic. On the ASR 1000 Series Aggregation Services Routers, scheduling decisions are performed in dedicated hardware and (**so?**) frequent scheduling decisions does not incur a performance penalty. We have optimized the hardware to maximize the elimination of burstiness from the stream it forwards, obviating user input on Bc or Be.

Shape Peak

The **shape peak** command is supported on the ASR 1000 Series Aggregation Services Router but it offers no functionality beyond the **shape average** command. We support it to easily migrate configurations from existing IOS classic devices to ASR 1000 Series Aggregation Services Routers. With **shape peak** command, the router will look at the configured rate, Bc and Be and then calculate a target shape rate. This rate displays in the **show policy-map interface** command output and on the ASR 1000 Series Aggregation Services Router is programmed into the hardware schedule entry. If you are creating a new configuration, you should use the **shape average** command.

Bandwidth Remaining Command

The **bandwidth remaining** command configures the excess weight in a schedule entry and so determines a queues share of the excess bandwidth. Recall that excess bandwidth is defined as any bandwidth that is neither explicitly guaranteed to another queue by the **priority** or **bandwidth** command nor used by a queue to which it is guaranteed. (For details on excess weight, see How Schedule Entries are Programmed, on page 359.) By distributing excess bandwidth sharing in a deterministic manner (behavior entirely determined by initial state), we avoid wasting bandwidth. (For further discussion of bandwidth sharing, see Bandwidth Queues, on page 376.)

The **bandwidth remaining** command is also an effective way to guarantee bandwidth to queues. It is perfectly reasonable, and very common, to apportion all bandwidth using only excess bandwidth sharing.

The **bandwidth remaining** command has two variants - **bandwidth remaining ratio** and **bandwidth remaining percent**. in either case, you are setting the same excess bandwidth parameter in a schedule entry. The rationale for two forms will make sense when we discuss hierarchical policies. In the context of a flat policy attached to a physical interface, however, you can choose whichever form simplifies provisioning.

Note Both variants (as similar to other scheduling commands) support the **account** keyword.

Bandwidth Remaining Ratio

Concerning the **bandwidth remaining ratio** command, the first thing you need to understand is that every bandwidth queue schedule entry will have a default Excess Weight (Ex) of one ('1') provided you do not explicitly set the value. (For example, upon creation, the schedule entry for the class-default queue will have an Ex of 1.) Having a deterministic and easy to understand default removes any ambiguity when designing a QoS scheme.

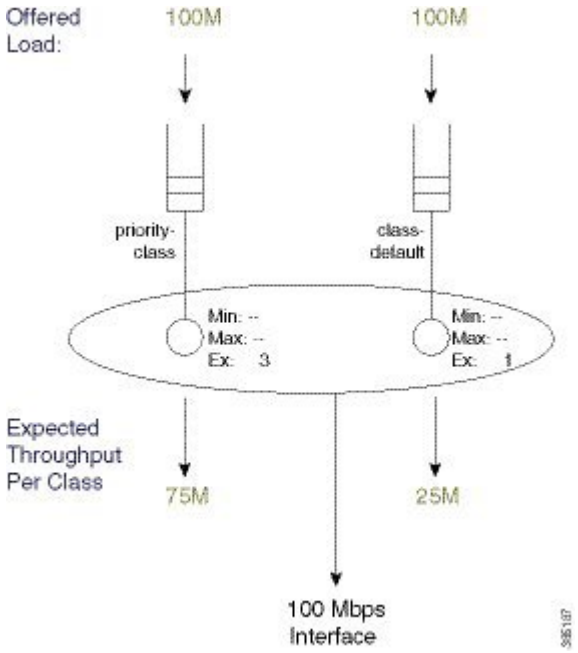
Consider the following policy-map example:

```
policy-map BRR-Example1
  class mission-critical
    bandwidth remaining ratio 3
```

This policy has 2 queues, one for the mission-critical class we explicitly create with a scheduling command and one for the implicit class-default.

Let's now attach this policy to a 100 Mbps interface and offer 100 Mbps to each queue. The scheduling hierarchy and expected throughput per class would be as shown:

Figure 24: Splitting Bandwidth Explicitly Assigned by Ratio



Now let's modify the policy by adding an explicit class with the **shape** command. Recall from the command page that the **shape peak** command sets the Max of the schedule entry for that queue. Because Ex has not been explicitly configured, it will default to 1.

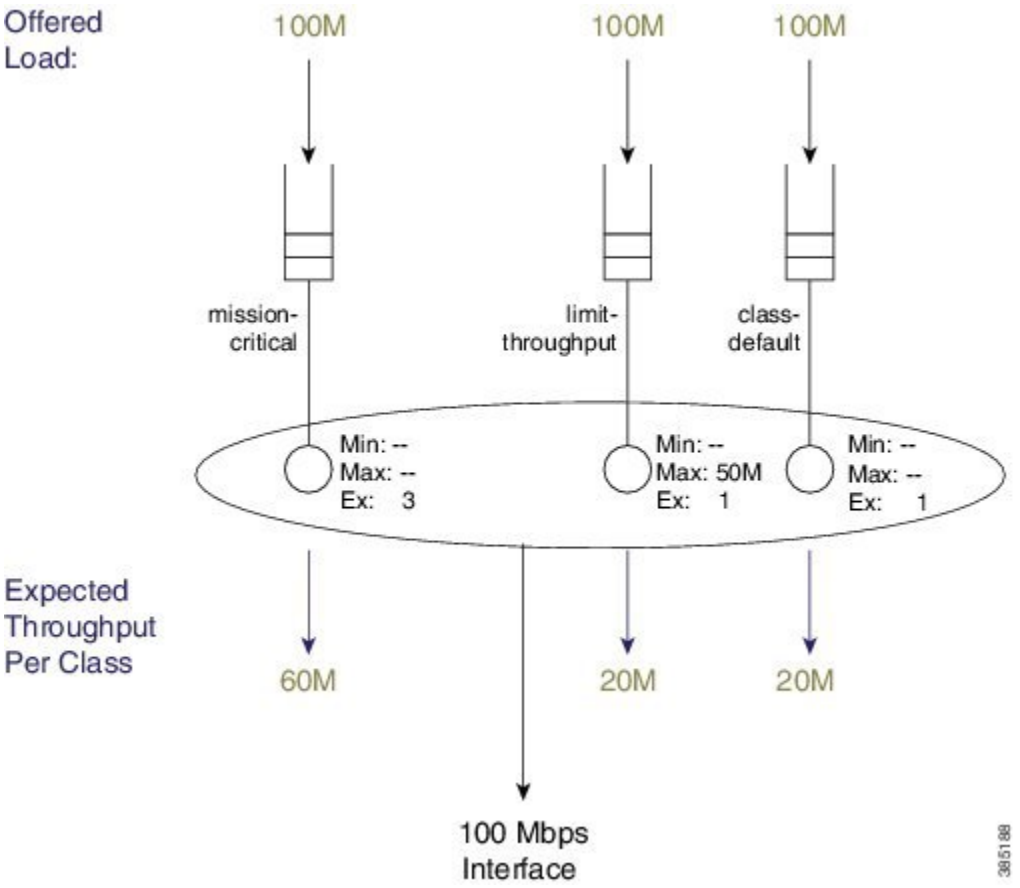
Note A Max entry does not guarantee any share of the bandwidth to a queue, it simply sets a ceiling on the possible throughput for that queue.

The policy could look like this:

```
policy-map BRR-Example1
  class mission-critical
    bandwidth remaining ratio 3
  class limit-throughput
    shape average 50m
```

If we attach this policy to a 100 Mbps interface and offer 100 Mbps to each class, the scheduling hierarchy and expected throughput would look as follows:

Figure 25: Modifying Excess Weight of Explicit Classes with bandwidth remaining ratio Command



The expected throughput (60M, 20M, and 20M) reflects the ratio of Ex values: 3, 1 , and 1. The key point is that modifying the excess weight using the **bandwidth remaining ratio** command will only alter the entry for the class you are explicitly modifying.

The bandwidth remaining ratio ranges from 1 to 1000 so we can achieve considerable variance between the service rate for different queues.

Bandwidth Remaining Percent

The **bandwidth remaining percent** command is another way to modify the Excess Weight (Ex) in a bandwidth queue's schedule entry. Obviously, with a percent-based scheme, the sum of excess weights across all bandwidth queues must total 100. We achieve this by distributing (equally) any percentage (not explicitly assigned) across class-default and any other queues that are not configured explicitly.

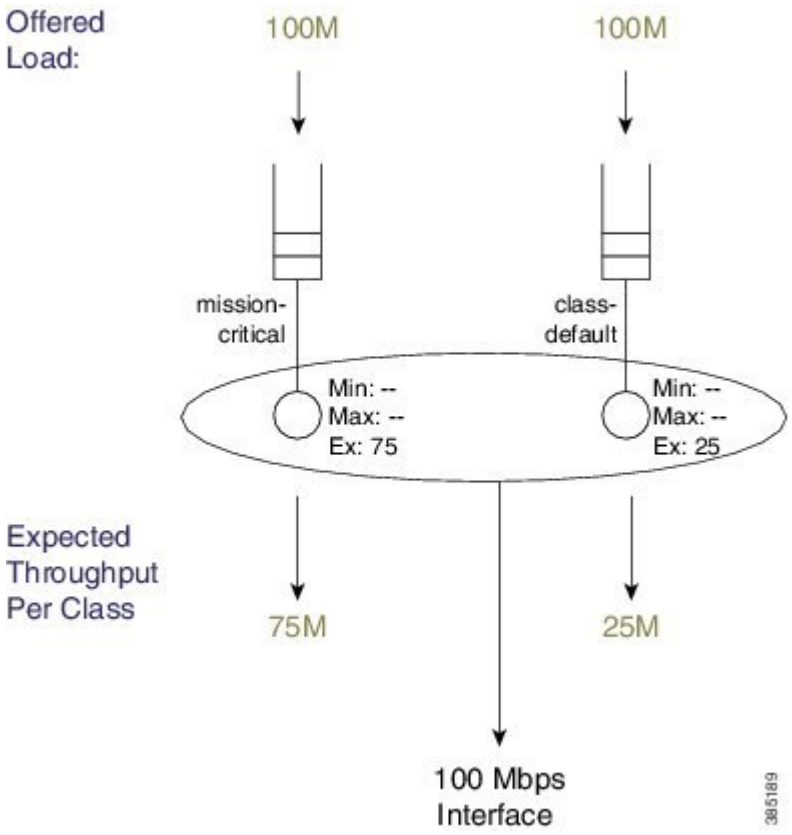
For details on this command, refer to the command page for bandwidth [remaining percent].

Consider the simplest example, which is equivalent to the first example in bandwidth remaining raito:

```
policy-map BRP-Example1
  class mission-critical
    bandwidth remaining percent 75
```

The scheduling hierarchy and expected throughput per class will look as follows:

Figure 26: Splitting Bandwidth Explicitly Assigned by Percent



Notice how the Ex of class-default was changed (from "1," by default) even though it was not explicitly configured.

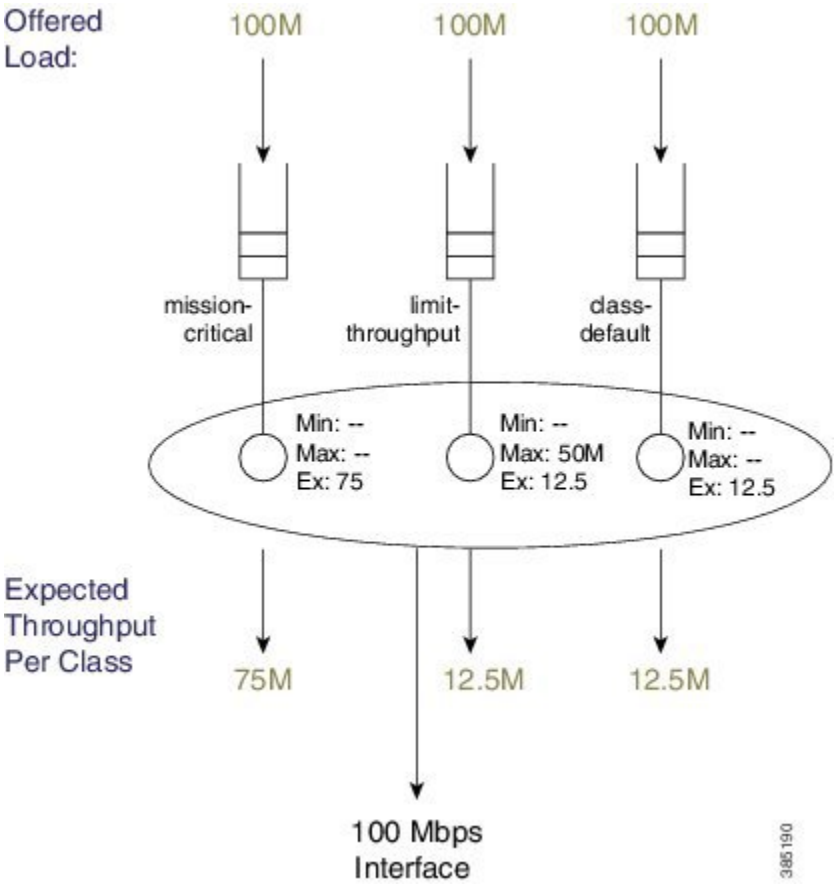
Now let's add a queuing class with no explicit bandwidth remaining configuration - again we'll add a class with just a shaper (see the figure "Splitting Bandwidth Explicitly Assigned by Ratio" in bandwidth remaining ratio:

```
policy-map BRP-Example2
  class mission-critical
    bandwidth remaining percent 75
  class limit-throughput
    shape average 50m
```

This example highlights the behavior of splitting percentage across class-default and any classes that are not explicitly assigned.

The hierarchy and throughput will now look as follows:

Figure 27: Splitting Bandwidth Percentage Across class-default and Unassigned Classes with an Added Shaper



Two-Parameter versus Three-Parameter Scheduling

Earlier we described how the schedule entry for each bandwidth queue has three parameters to control queue service: Min, Max and Ex. (See How Schedule Entries are Programmed, on page 359.) This is why we categorize the scheduler implementation on the ASR 1000 Series Aggregation Services Router as a *three-parameter scheduler*.

In an existing IOS classic implementation, we provide a simpler *two-parameter scheduler*. Instead of distinct entries for Min and Ex, each schedule entry has only a single weight. Whether you used the **bandwidth** or

bandwidth remaining command you were configuring the same single weight. To grasp the difference, let's look at an example focusing on the **bandwidth** command.

The policy-map for this example will look as follows:

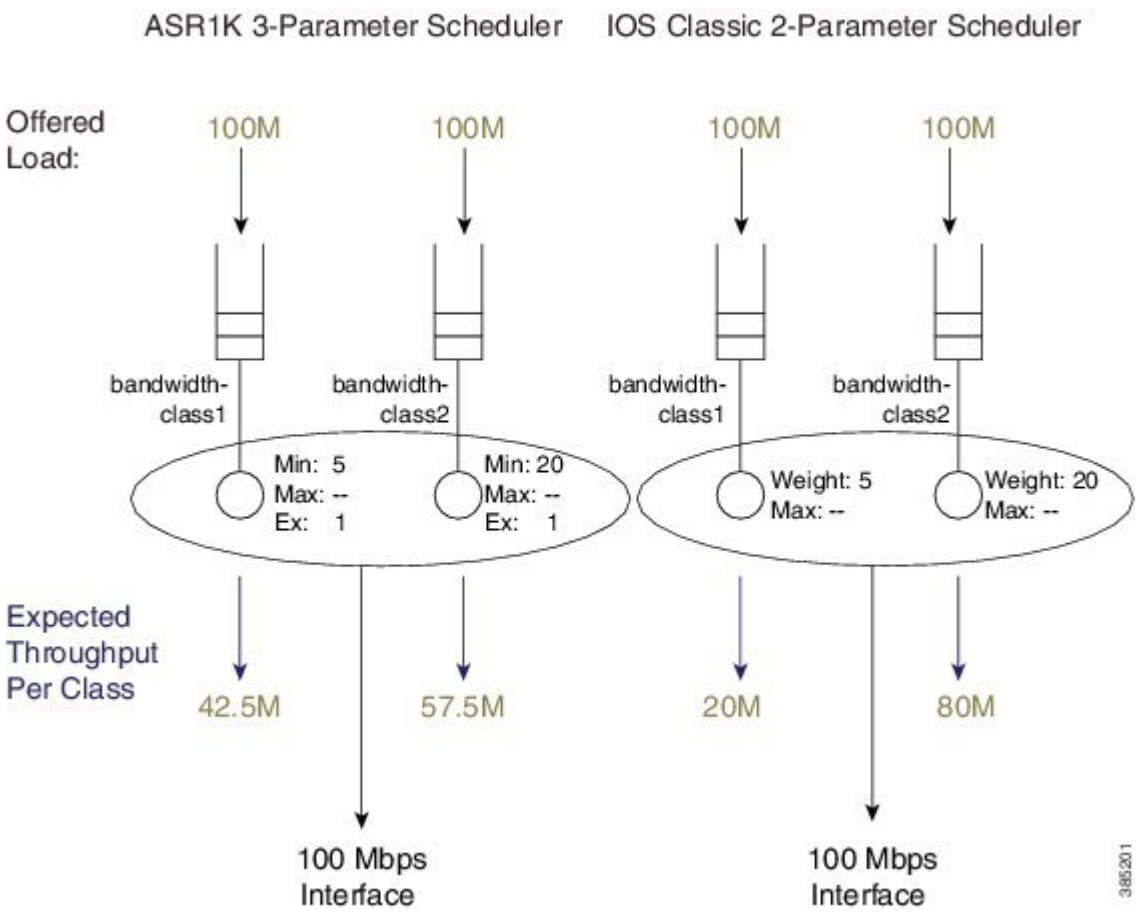
```
policy-map bandwidth-example
  class bandwidth-class1
    bandwidth percent 5
  class bandwidth-class2
    bandwidth percent 20
```

Now consider the policy-map attached to a 100 Mbps interface and offer 100 Mbps to each queue. The following figure shows how the schedule configuration and expected throughput would appear both on an ASR 1000 Series Aggregation Services Router (on the left) and on a router running an IOS Classic image (e.g., a Cisco 7200; on the right).

25 Mbps is assigned to the ASR 1000 Series Aggregation Services Router three-parameter scheduler and we use it to honor Min guarantees, which leaves 75 Mbps of excess bandwidth. This excess is shared equally between the two queues based on the default excess-weight of '1' that each queue will receive.

If we apply the same configuration on a two-parameter scheduler, the configured bandwidth values will dictate a single-weight parameter in the schedule entry. The concept "Min scheduling versus excess bandwidth sharing" does not apply here. Instead, for each entry, all bandwidth sharing hinges on the single weight.

Figure 28: Same Configuration running on ASR 1000 with IOS XE and Router running IOS Classic



Looking at this example you see that the same configuration on an ASR 1000 Series Aggregation Services Router running IOS XE can yield significantly different behavior from a router running an IOS classic image.

Note To achieve identical behavior to an router running IOS Classic, you can use a configuration using only excess bandwidth sharing. Changing **bandwidth percent** statements in an IOS Classic configuration to **bandwidth remaining percent** statements in IOS XE is an easy way to migrate existing configurations.

Schedule Burstiness

Possible sources of burstiness in scheduling include: packet batching and the scheduler's representation of time.

Packet Batching

This source is intentional and should not cause concern. As implemented in hardware, we cap the number of decisions a schedule can make per second. If you were to send all small packets, say 64-byte frames, a schedule may struggle to maintain if it is making decisions, packet by packet, for a fast interface like 10 Gbps. To ease this burden, the hardware will batch small packets (up to about 512 bytes from the same queue) and let the scheduler treat them as a single decision.

So, if a single packet of 512 bytes were at the head of the queue we would send that to the schedule as a single packet. On the contrary, if five 64-byte packets were at the head of the queue we would batch the packets as a single packet from the scheduler's perspective. That is, we would pull all five packets from the queue simultaneously and forward them back to back on the wire as a single burst. As the size of a single MTU greatly exceeds that of a burst, the later negligibly impacts downstream buffering or jitter for other queues.

Scheduler's Representation of Time

The second potential source of burstiness arises from how a schedule in hardware tracks time. If you mix very small rates (say 100K and less) and very large rates (say 100M and higher) in the same policy-map you may experience unexpected burstiness in scheduling of traffic from the queue configured with the high rate.

When you use *real-time scheduling* (using either the **bandwidth** or **shape** command) you are specifying rates in bits per second. This means that each schedule entry must have a concept of real time and must be monitoring service rate vs that real time. The representation of time must be uniform across all entries in a given scheduler.

Consider an 8-Kbps shaper (8000 bits/sec = 1000 bytes/sec).

Sending 64-byte packets would be (equivalent to) sending one packet every (64-byte packet * 64/1000 =) 64 mS.

Sending 1500-byte packets would be (equivalent to) sending one packet every 1500 byte * 1500/1000 =) 1.5 sec.

We want to represent anything ranging from 64 mS to many seconds. To do this we count time and establish that every counter increment represents 10.5 mS of real time.

Now consider a 10-Gbps shaper. In 10.5 mS we would expect to send 8,750 1500-byte packets:

10,000,000,000 b/sec * .0105 sec = 105,000,000 bits, which equals 105,000,000/8/1500, or 8750 1500-byte packets

This is a huge data burst. If we were counting time in 10.5 mS increments, whenever the clock (advances) we would need to send that burst. In contrast, if .65 mS represented *real time*, we would expect to send 542 1500-byte packets (a far more manageable situation).

The representation of time is driven by the lowest rate configurable within a policy-map. The following table shows the granularity of time chosen vs. rate that is configured in a policy-map. (The details are accurate for ESP-20 but only similar for all variants of ASR1K hardware.)

Reading the table you observe that if all rates in your policy-map are 116K or greater, then any burst introduced by this representation of time would last less than a millisecond and therefore insignificant. If you configure shape or bandwidth rates less than 116K on a fast interface you may want to ensure there are no unintended consequences. (e.g., if all rates in your policy-map range from 29K - 57K, then any burst introduced by this representation of time would be 2.6 mS in duration!) Such consequences could include downstream devices dropping if they have insufficient buffering to receive such a burst, WRED dropping packets, or downstream policers dropping packets due to exceeding their burst allowance.

Minimum Guaranteed Service Rate for a Queue

For any queue you can calculate a minimum guaranteed service rate (the service that a queue will receive if all other queues are congested). In some prior examples we have shown an offered rate of 100% to each queue - the expected throughput in those examples is the minimum guaranteed service rate. Knowing this rate might inform you on how your applications will behave under severe congestion. That is, when the network is systemically overloaded, can you expect your application to run?

One particularly useful number you can calculate from the guaranteed rate is the delay a packet would experience if the egress queue were full. For example, let's consider an oversubscribed video queue, whose policy-map looked as follows:

```
policy-map min-service-rate-example
  class priority-class
    priority
    police cir 1m 1250
  class video
    bandwidth 1000
  class mission-critical
    bandwidth 2000
```

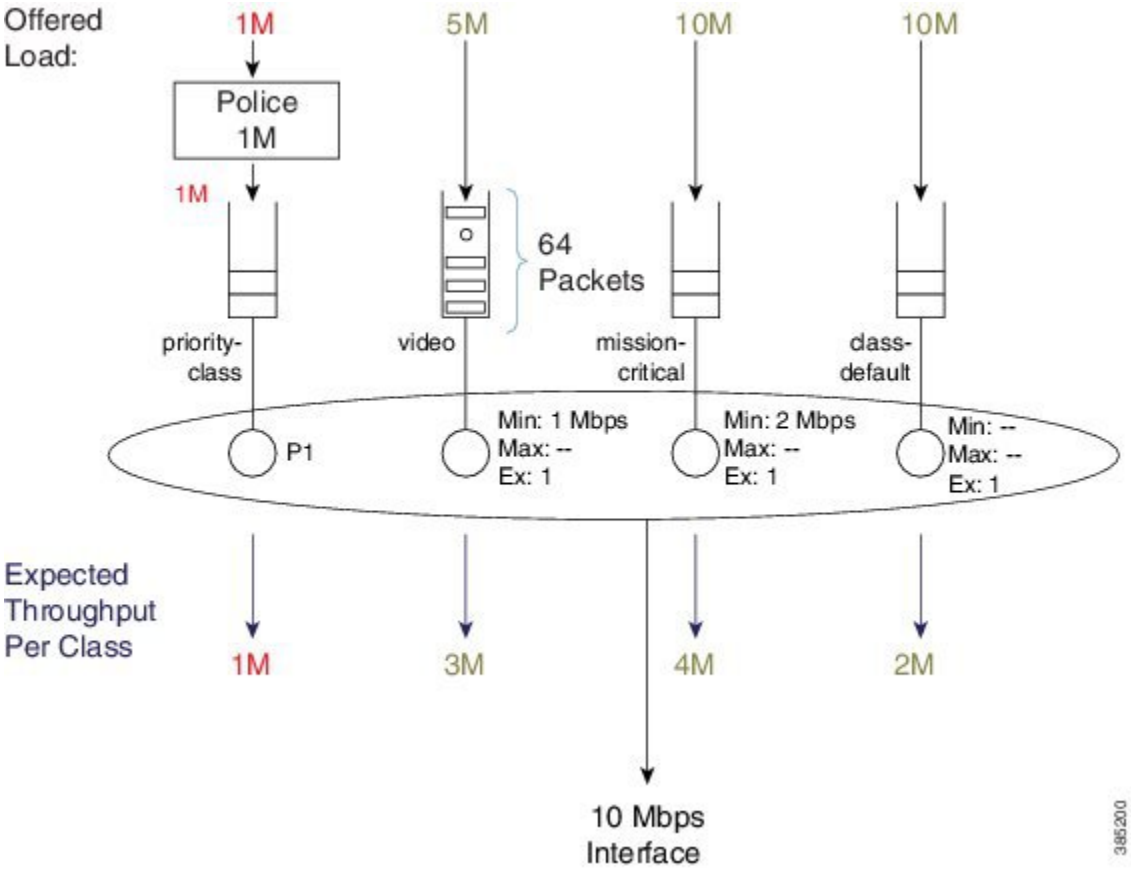
In this example we attach the policy-map to a 10 Mbps Ethernet interface and the offered load to each class as shown in the following analysis. (We will abide by the guidelines in Schedule Operation, on page 360.)

The video class is serviced at its minimum guaranteed service rate of 3 Mbps (1 Mbps (Min configured with the **bandwidth** command) + 2 Mbps (its always guaranteed share of excess bandwidth). Looking more closely at the calculations involved:

10 Mbps - 1 Mbps(for P1) - 3 Mbps(Min guaranteed for video and mission-critical) = 6 Mbps

6 Mbps excess bandwidth(after accounting for P1 and Min guarantees)/3 equal shares for all queues = 2 Mbps

Figure 29: Minimum Service Rate and "Experience" of Latency



With the minimum guaranteed service rate, you can now calculate "the experience" of latency packets arriving at the video queue. Using the default queue-limit of 64 packets (under over-subscription, we would expect the queue to fill and contain 64 packets) a new packet would be either dropped or placed at the tail of the queue (if the packet arrived just when another was pulled from the video queue).

Given this queue for video traffic, we could expect the average packet size to be about 1400 bytes (roughly the size of an MPEG I-frame), generating 716,800 bits as the amount of data buffered and awaiting transmission:

64 packets * 1400 bytes/packet * 8 bits/byte = 716,800 bits

Given a minimum rate of 3 Mbps, we would require 239 mS to drain this queue:

716,800 bits / 3 Mbps = 0.239 Seconds (239 mS)

As you can see, a minimum guaranteed service rate can help you envisage (and so predict) the behavior of your applications under congested conditions.

Pak Priority

Pak priority designates a scheme to protect some critically important control packets (interface keepalives, BFD packets, some routing protocol hellos etc.) that are vital for network stability. In this section, we will describe these packets and outline how they are scheduled.

Note The name, pak_priority, is unfortunate and might cause confusion because the control packets are not (actually) queued in a priority queue.

With pak_priority, we attempt to ensure guaranteed delivery and not guaranteed low latency) of the control packets. They are marked with an internal pak_priority flag, when first generated in the control plane. This flag does not propagate outside the router and is only used to ensure we give special treatment to the packet as we send it towards the egress interface.

Observe that we set the DSCP of IP encapsulated control packets to CS6 protecting them at other devices in the network where they must traverse. On the router that generates the control packet, the pak_priority designation dictates further protection beyond what you might configure for CS6 packets.

The following table lists the packets and protocols we mark with the internal pak_priority flag.

Packets and Protocols Marked with the pak_priority Flag

Table 36: Control Packets Marked with pak_priority Flag

Level Marked	Packets and Protocols
Layers 1 and 2	
	ATM Address Resolution Protocol Negative Acknowledgment (ARP NAK)
	ATM ARP requests
	ATM host ping operations, administration and management cell(OA&M)
	ATM Interim Local Management Interface (ILMI)
	ATM OA&M
	ATM ARP reply
	Cisco Discovery Protocol
	Dynamic Trunking Protocol (DTP)
	Ethernet loopback packet
	Frame Relay End2End Keepalive
	Frame Relay inverse ARP
	Frame Relay Link Access Procedure (LAPF)
	Frame Relay Local Management Interface (LMI)
	Hot standby Connection-to-Connection Control packets (HCCP)

Level Marked	Packets and Protocols
	High-Level Data Link Control (HDLC) keep-alives
	Link Aggregation Control Protocol (LACP) (802.3ad)
	Port Aggregation Protocol (PAgP)
	PPP keep-alives
	Link Control Protocol (LCP) Messages
	PPP LZS-DCP
	Serial Line Address Resolution Protocol (SLARP)
	Some Multilink Point-to-Point Protocol (MLPP) control packets (LCP)
IPv4 Layer 3	
	Protocol Independent Multicast (PIM) hellos
	Interior Gateway Routing Protocol (IGRP) hellos
	OSPF hellos
	EIGRP hellos
	Intermediate System-to-Intermediate System (IS-IS) hellos, complete sequence number PDU (CSNP), PSNP, and label switched paths (LSPs)
	ESIS hellos
	Triggered Routing Information Protocol (RIP) Ack
	TDP and LDP hellos
	Resource Reservation Protocol (RSVP)
	Some L2TP control packets
	Some L2F control packets
	GRE IP Keepalive
	IGRP CLNS
	Bidirectional Forwarding Protocol (BFD)

Levels of Protection for pak_priority Packets

First level

A policer or WRED will not drop packets with this designation.

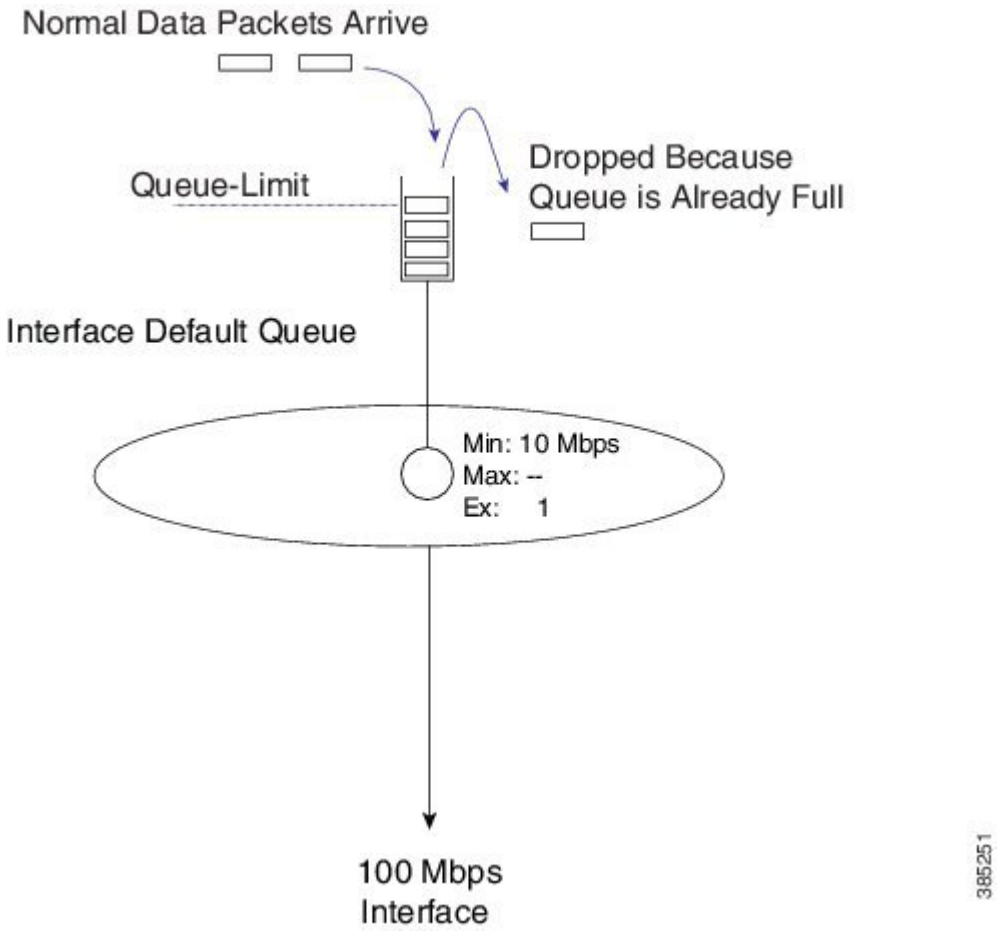
Second level

We will enqueue pak_priority packets even if an egress queue is already full. To grasp this let's first look at a physical interface that has no QoS configured, where we still need a queue and (therefore) a schedule to pull packets from that queue.

Without QoS configured on the interface, we have a single first-in first-out (FIFO) queue (referred to as the *interface default queue*). (Do not confuse with a *class-default queue*).

In the diagram below, normal data packets arrive when the queue is full. Notice that the schedule entry has the typical Min and default Ex values (10% of the interface rate and 1). Because only one queue exists, these values have no effect; without competition, one queue will receive all the available bandwidth.

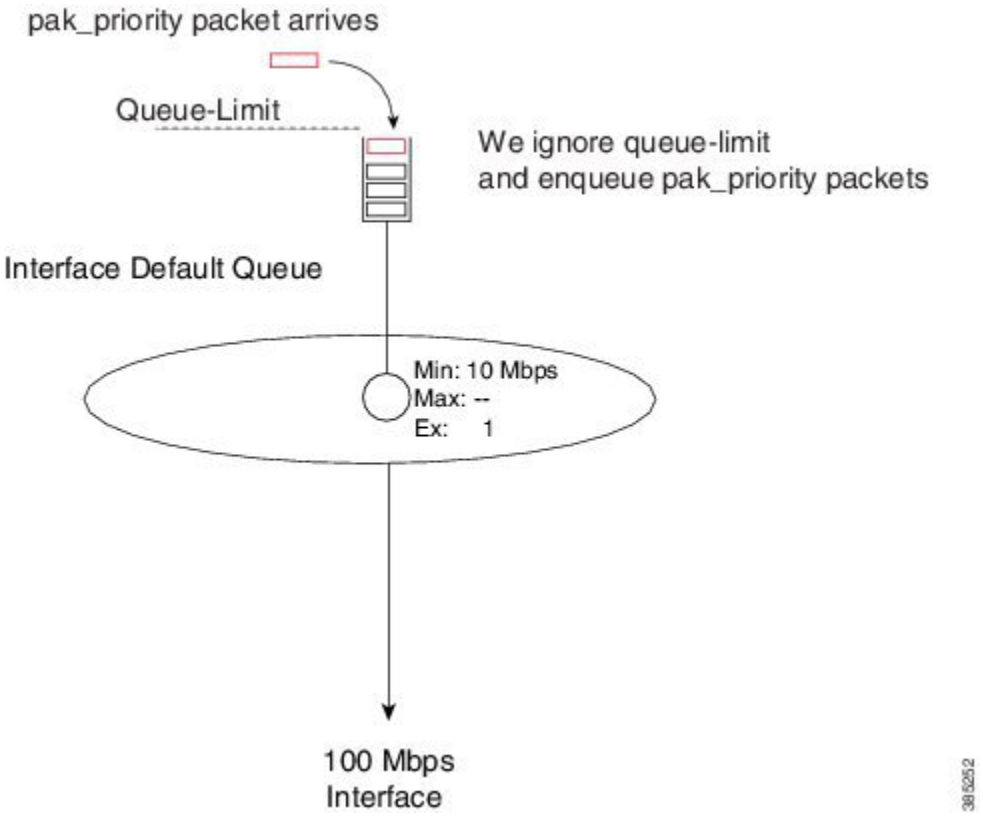
Figure 30: Normal Packet Arrives when Queue is Full



As with every queue, a queue-limit determines how much data we can buffer before the queue is considered full.

If a pak_priority packet arrives while the queue is full, we ignore the queue-limit and enqueue it; the packet must wait until all leading packets are sent. On most ASR 1000 Series Aggregation Services Router platforms, the default queue-limit is 50 mS. So, we may delay the pak_priority packet for 50 mS but we do guarantee its delivery.

Figure 31: pak_priority Packet Arrives when Queue is Full, and we Ignore queue-limit

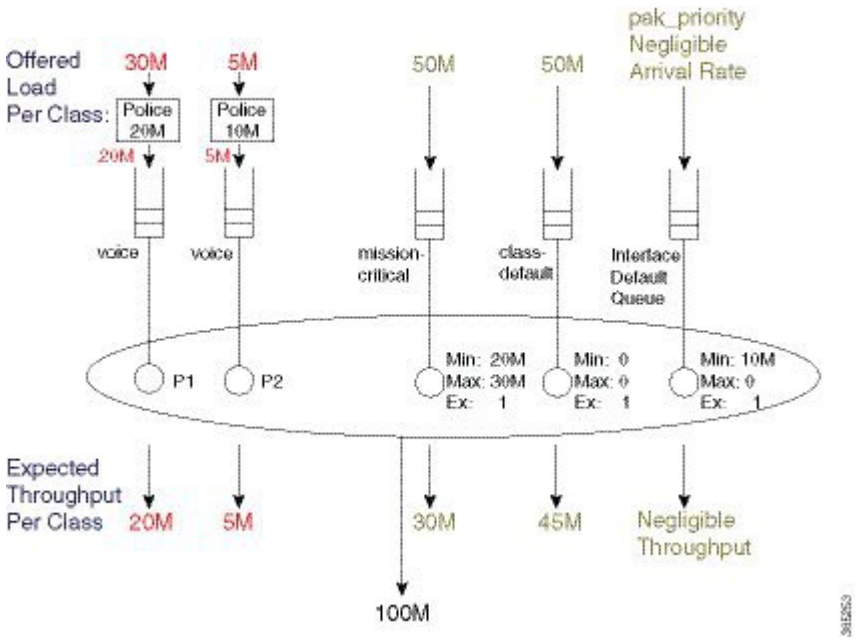


The discussion (of pak priority) changes slightly when you configure QoS on an interface. To illustrate, let's reconsider one of the very first examples in this chapter (see Schedule Operation: With a Shaper, on page 362).

```
policy-map scheduling-example
  class voice
    priority level 1
    police 20m
  class video
    priority level 2
    police 10m
  class mission-critical
    bandwidth 20000
    shape average 30m
```

We didn't show it previously and as you will now observe below, when you configure a policy we do not remove the interface default queue.

Figure 32: Repurposing Interface Default Queue as a Dedicated pak_priority Queue



Pak_priority packets are still enqueued in the interface default queue. Essentially, the queue has been repurposed as a dedicated pak_priority queue.

Looking at the diagram you can now see the importance of the Min value (set at 10% of linerate). This value ensures that other queues (with a Min service configured) cannot deprive this queue of service.

However, although we configure a Min of 10% of linerate we will never consume this bandwidth. We only mark a very small number of critical packets as pak_priority so the rate at which they arrive is negligible. We overpromise on the Min with the understanding that it will not impact behavior of other queues. Were we to mark too many packets "pak_priority," this scheme would not work.

With routing protocols we mark Hello packets but not routing updates with pak_priority. So, you should create a bandwidth queue for CS6 packets.

The Hello packets will pass through the interface default queue and the routing updates will use the newly-created bandwidth queue.

An exception is BGP, where we do not mark Hello packets as pak_priority. Why? BGP Hello packets and updates share the same TCP stream. Providing special treatment would cause TCP packets to arrive out of sequence. This offers no benefit as you could not consume them.

Classifying a pak_priority packet (by matching DSCP or any other field) and attempting to move it to a bandwidth queue will not happen - the packet will still be enqueued in the interface default queue. However, you can classify the packet and move it to a designated priority queue.

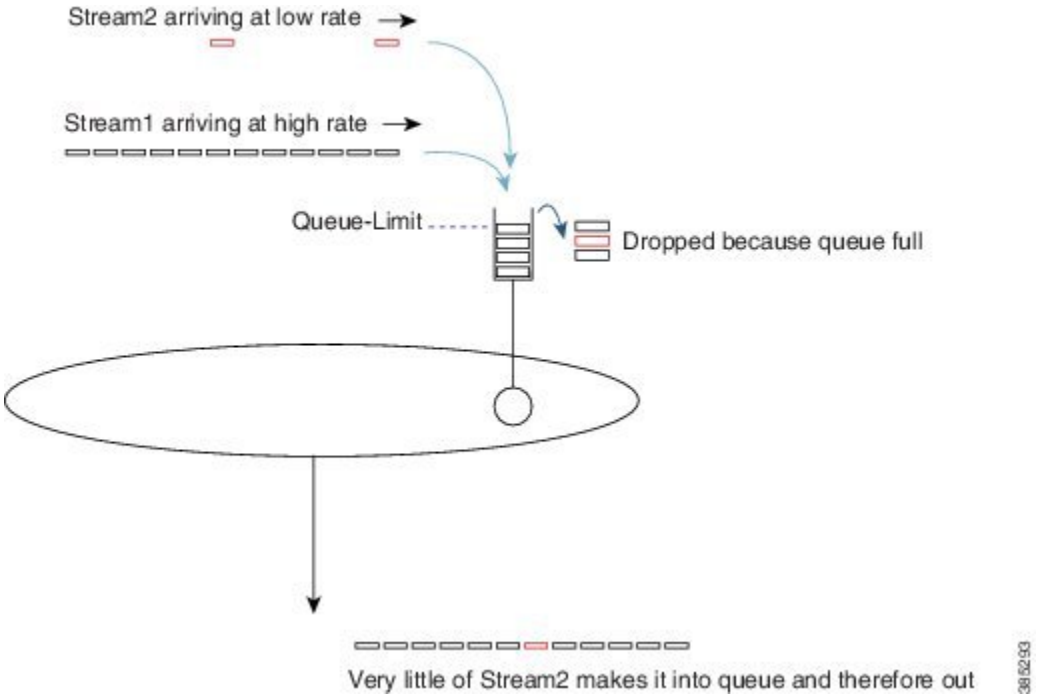
Flow-Based Fair Queuing

Flow-based fair queuing enables us to provide some fairness between flows that belong to the same class in a policy-map. It provides some protection for low speed (well behaved) streams of traffic competing with high-rate streams.

If an individual class is oversubscribed (the offered rate exceeds the service rate), one high-rate flow can starve low-rate flow(s) of service. To understand this you need to consider multiple streams targeting the same physical queue. After it fills, additional packets are dropped. Whenever the scheduler sends a packet from the queue, a single space will open at the queue's tail. We successfully enqueue whichever packet arrives next. If you look at the following example, you notice that a packet from Stream1 (the high rate stream) is more likely to arrive next. Fairness between treatment of the streams is absent! Whatever exits the interface is purely driven by what packet manages to make it into the queue.

Not only does the high rate stream obtain an unfair share of the class bandwidth, it also impacts latency for low rate streams. As successfully-enqueued packets are always at the tail of a full queue, they must wait for all other packets to drain before they can be transmitted.

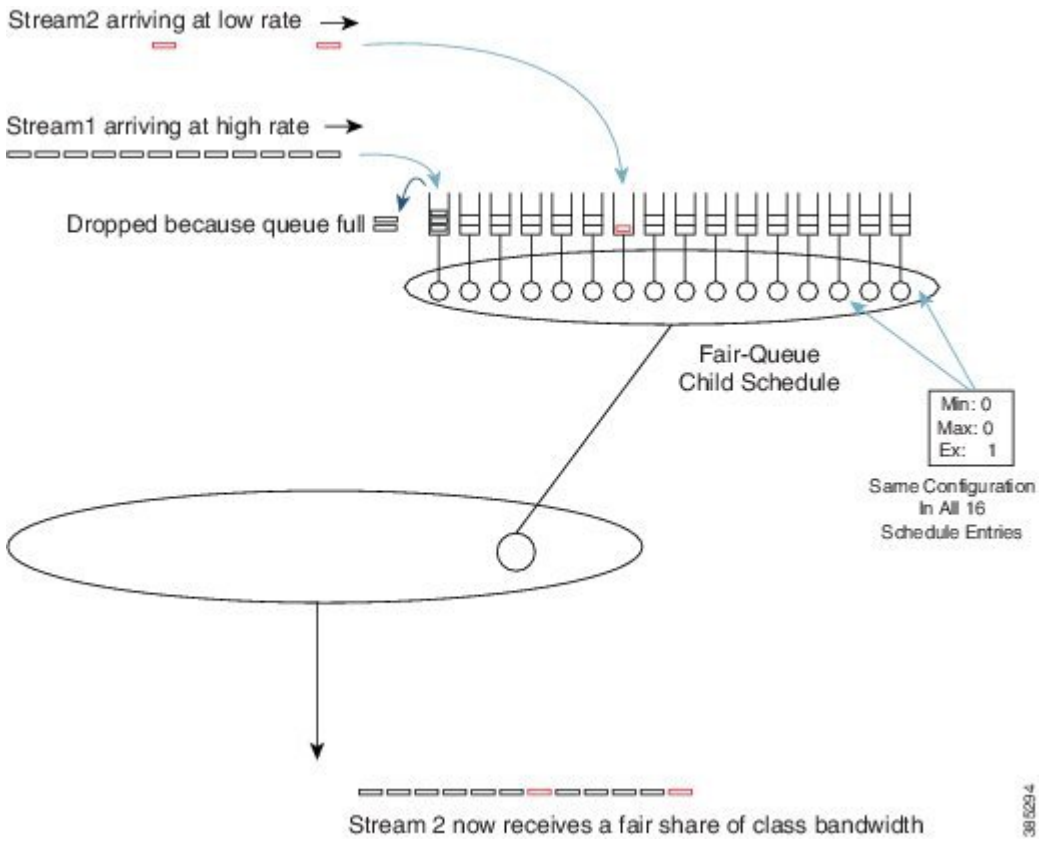
Figure 33: Latency without Flow-Based Fair Queuing



Flow-based fair queuing can alleviate this issue. When you issue the **fair-queue** command you direct the router to create 16 queues for that single class, which represents a simple form of *hierarchical scheduling*. .

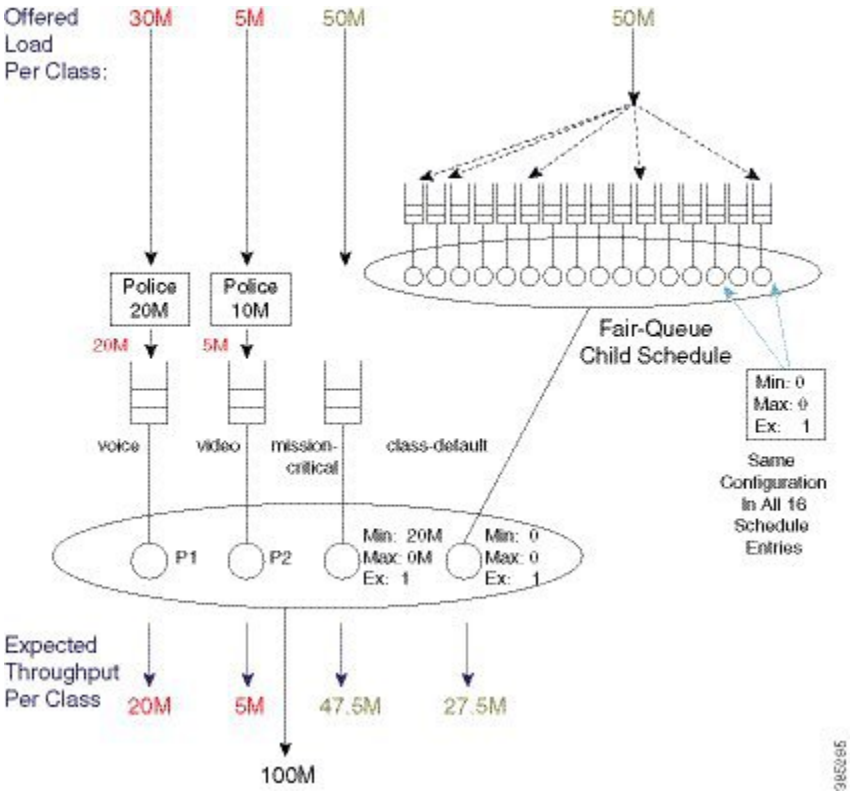
Consider the fair-queue child schedule a *pre-sorter*, providing sorting and fairness between multiple streams targeting the same class.

Figure 34: Pre-sorting and Fairness provided by Flow-Based Fair Queuing




```
policy-map fair-queue-example
  class voice
    priority level 1
    police 20m
  class video
    priority level 2
    police 10m
  class mission-critical
    bandwidth 20000
  class class-default
    fair-queue
```

Figure 35: Packet Flow with Flow-Based Fair Queuing



Verification

Use the **show policy-map interface** *interface* command to verify operation of scheduling. This command will show long term trends and a complete view of the policy configured.

The data plane sends statistics to the control plane every 10 seconds and control plane refreshs its own statistics every 10 seconds. This means that the values in output of **show policy-map interface** command updates every 10 seconds. Some counters that represent instantaneous state, such as current queue depth, may not be overly useful. It is possible to look directly at hardware counters if you really want true instantaneous state.

The following configuration is an example of **show policy-map interface***interface* command.

```
policy-map show_policy-example
 class voice
  priority level 1
  police cir percent 10 bc 5 ms
 class video
   priority level 2
  police cir percent 20 bc 10 ms
 class critical-data
  bandwidth percent 50
```

This policy has four classes, the three that are explicitly configured and the implicit class-default.

The output from the **show policy-map interface** command mirrors the configured policy. It has a section for each configured class. Within each class the output is consistently organized with a classification section and a section for each configured action.

Note that queuing information for priority classes is shown separately to other features (policers) in that class. This is since multiple priority classes may map into the same queue.

The following is an example of the output from the **show policy-map interface** command. Although it is one continuous block of output we break it into sections to highlight the structure of the output.

`Device#show policy-map interface g1/0/4`
` GigabitEthernet1/0/4`
 	
`  Service-policy output: show_policy-example`
 	
`queue stats for all priority classes:`
`  Queueing`
`  priority level 1`
`  queue limit 512 packets`
`  (queue depth/total drops/no-buffer drops) 0/0/0`
`  (pkts output/bytes output) 39012/58518000`
	This section shows queue information for the priority level 1 queue

`  queue stats for all priority classes:`
`  Queueing`
`  priority level 2`
`  queue limit 512 packets`
`  (queue depth/total drops/no-buffer drops) 0/0/0`
`  (pkts output/bytes output) 61122/91683000`
	This section shows queue information for the priority level 2 queue

`Class-map: voice (match-all)`
`  39012 packets, 58518000 bytes`
`  5 minute offered rate 672000 bps, drop rate 0000 bps`
`  Match:  dscp ef (46)`
	This section shows statistics for the class named voice.
	First shown is the classification statistics.

`Priority: Strict, b/w exceed drops: 0`
`  Priority Level: 1`
 	
`  police:`
`  cir 10 %, bc 5`
`  cir 100000000 bps, bc 62500 bytes`
`  conformed 39012 packets, 58518000 bytes; actions:`
`  transmit`
`  exceeded 0 packets, 0 bytes; actions:`
 	
`  drop`
`  conformed 672000 bps, exceeded 0000 bps`
	Priority level indicates the priority queue above that will be used by this class.
	The statistics for the policer used for queue admission control are also shown here.

`Class-map: video (match-all)`
`1376985 packets, 2065477500 bytes`
`5 minute offered rate 9171000 bps, drop rate 0000 bps`
`Match: dscp af41 (34)`
	This is start of section for class named video.
	Again classification statistics and criteria are shown first

`police:`
`cir 20 %, bc 10`
`cir 200000000 bps, bc 250000 bytes`
`conformed 1381399 packets, 2072098500 bytes; actions:`
`transmit`
`exceeded 0 packets, 0 bytes; actions:`
`drop`
`conformed 9288000 bps, exceeded 0000 bps`
`Priority: Strict, b/w exceed drops: 0`
`Priority Level: 2`
	This section shows actions configured in the class named video.
	The statistics for the queue admission control policer.
	Priority Level indicates packets from this class will be enqueued in the priority level 2 queue that is shown above.

`Class-map: critical-data (match-all)`
`45310 packets, 67965000 bytes`
`5 minute offered rate 719000 bps, drop rate 0000 bps`
`Match: dscp af11 (10)`
	This is start of section for class named critical-data.
	As always the classification statistics and criteria are shown first.

`Queueing`
`queue limit 2083 packets`
`(queue depth/total drops/no-buffer drops) 0/0/0`
`(pkts output/bytes output) 45310/67965000`
`bandwidth 50% (500000 kbps)`
	Since this class has the bandwidth action a queue is created for the class.
	This section shows the queue related configuration and statistics.

`Class-map: class-default (match-any)`
`51513 packets, 77222561 bytes`
`5 minute offered rate 194000 bps, drop rate 0000 bps`
`Match: any`
	This is the start of the section for class-default, the implicit class that exists in every policy.
	As always we first show the statistics for packets deemed to belong to this class.

`queue limit 4166 packets`
`(queue depth/total drops/no-buffer drops) 0/0/0`
`(pkts output/bytes output) 1371790/2057638061`
	This section shows the queue information for class-default.

As mentioned above the **show policy-map interface** command receives an update from the data plane every 10 seconds.

It is also possible to look directly at the data plane for a real time view of system behavior. The command will also allow you to verify the data plane is programmed as expected.

Counters for long term events such as packets enqueued or packets dropped will be cleared each time that information is pushed to the control plane, every 10 seconds.

Perhaps the most useful counter in the data plane show output is the instantaneous queue depth. As this is read each time you issue the command you can get realtime visibility into whether a queue is congested, is it sustained congestion or bursty behavior etc.

The **show platform hardware qfp active feature qos interface** *interface* is the command to view QoS configuration and statistics in the hardware data plane.

The following show example output from the command corresponding to the configuration and **show policy-map interface** example above.

You can see the output of the command again reflects the structure of the policy-map with a section for each class configured.

`Device#show platform hardware qfp active feature qos interface gig1/0/4`
`Interface: GigabitEthernet1/0/4, QFP interface: 11`
`  Direction: Output`
`    Hierarchy level: 0`
`    Policy name: show_policy-example`	

`      Class name: voice, Policy name: show_policy-example`
`        Police:`
`            cir:  100096000 bps,  bc: 63488 bytes`
`            pir: 0 bps,  be: 0 bytes`
`            rate mode: Single Rate Mode`
`            conformed: 0 packets, 0 bytes; actions:`
`              transmit`
`            exceeded: 0 packets, 0 bytes; actions:`
`              drop`
`            violated: 0 packets, 0 bytes; actions:`
`              drop`
`            color aware: No`
`            green_qos_group: 0, yellow_qos_group: 0`
`            overhead accounting: disabled`
`            overhead value: 0, overhead atm: No`
	Start of section for class named voice.
	Policer configuration and statistics for current 10 second interval

`Queue: QID: 175 (0xaf)`
`bandwidth (cfg) : 0`
`, bandwidth (hw) : 0`
`shape (cfg) : 0`
`, shape (hw) : 0`
`prio level (cfg) : 1`
`, prio level (hw) : 0`
`limit (pkts ) : 512`
`drop policy: tail-drop`
`Statistics:`
`depth (pkts ) : 0`
`tail drops (bytes): 0`
`, (packets) : 0`
`total enqs (bytes): 0`
`, (packets) : 0`
`licensed throughput oversubscription drops:`
`(bytes): 0`
`, (packets) : 0`
`Schedule: (SID:0x258)`
`Schedule FCID : 16`
`bandwidth (cfg) : 1050 Mbps`
`, bandwidth (hw) : 1050.01 Mbps`
`shape (cfg) : 1050 Mbps`
`, shape (hw) : 1050.01 Mbps`
	Queue information for class voice
	This depth is instantaneous queue depth – can be very useful

`Class name: class-default, Policy name: show_policy-example`
`Queue: QID: 176 (0xb0)`
`bandwidth (cfg) : 0`
`, bandwidth (hw) : 0`
`shape (cfg) : 0`
`, shape (hw) : 0`
`prio level (cfg) : 0`
`, prio level (hw) : n/a`
`limit (pkts ) : 4166`
`drop policy: tail-drop`
`Statistics:`
`depth (pkts ) : 0`
`tail drops (bytes): 0`
`, (packets) : 0`
`total enqs (bytes): 3420000`
`, (packets) : 2280`
`licensed throughput oversubscription drops:`
`(bytes): 0`
`, (packets) : 0`
	Start of section for class-default.
	Queue information for this class.
	Instantaneous depth and statistics for current 10 second interval

`Class name: video, Policy name: show_policy-example`
`Police:`
`cir: 200064000 bps, bc: 253952 bytes`
`pir: 0 bps, be: 0 bytes`
`rate mode: Single Rate Mode`
`conformed: 0 packets, 0 bytes; actions:`
`transmit`
`exceeded: 0 packets, 0 bytes; actions:`
`drop`
`violated: 0 packets, 0 bytes; actions:`
`drop`
`color aware: No`
`green_qos_group: 0, yellow_qos_group: 0`
`overhead accounting: disabled`
`overhead value: 0, overhead atm: No`
	Start of section for class named video.
	Admission control policer configuration and statistics are shown first.

`Queue: QID: 178 (0xb2)`
`bandwidth (cfg) : 0`
`, bandwidth (hw) : 0`
`shape (cfg) : 0`
`, shape (hw) : 0`
`prio level (cfg) : 2`
`, prio level (hw) : 1280`
`limit (pkts ) : 512`
`drop policy: tail-drop`
`Statistics:`
`depth (pkts ) : 0`
`tail drops (bytes): 0`
`, (packets) : 0`
`total enqs (bytes): 0`
`, (packets) : 0`
`licensed throughput oversubscription drops:`
`(bytes): 0`
`, (packets) : 0`
`Schedule: (SID:0x258)`
`Schedule FCID : 16`
`bandwidth (cfg) : 1050 Mbps`
`, bandwidth (hw) : 1050.01 Mbps`
`shape (cfg) : 1050 Mbps`
`, shape (hw) : 1050.01 Mbps`
	Queue information for class video
	Instantaneous queue depth and statistics for current 10 second interval

`Class name: critical-data, Policy name: show_policy-example`
`  Queue: QID: 177 (0xb1)`
`  bandwidth (cfg)  : 500000000`
`  , bandwidth (hw)  : 500000000`
`  shape (cfg)  : 0`
`  , shape (hw)  : 0`
`  prio level (cfg)  : 0`
`  , prio level (hw)  : n/a`
`  limit (pkts )  : 2083`
`  drop policy: tail-drop`
`  Statistics:`
`  depth (pkts )  : 0`
`  tail drops (bytes): 0`
`  , (packets)  : 0`
`  total enqs (bytes): 0`
`  , (packets)  : 0`
`  licensed throughput oversubscription drops:`
`  (bytes): 0`
`  , (packets)  : 0`
	Start of section for class named critical-data.
	Instantaneous queue depth and statistics for current 10 second interval.

Command Reference

Account

Account is not an independent command but rather an extension to scheduling commands that allows a user to specify overhead accounting for that command. Account is presented here to avoid replication in each of the scheduling commands.

Syntax description:

To configure a user defined number of bytes to be added to or subtracted from the scheduling length:

[**no**] **shape** | **bandwidth** *rate* **account user-defined** *value* [**atm**]

To specify encapsulation of a downstream device and automatically calculate the overhead accounting adjustment:

[**no**] **shape** | **bandwidth** *rate* **account dot1q** | **qing** *encapsulation*

Command Default:

By default Layer 3 Datagram and Layer 2 headers are included in scheduling calculations.

Usage Guidelines:

If the account option is used in one class containing scheduling actions in a policy-map, the account command with same values must be used in all classes containing scheduling actions. Similarly in a hierarchical policy-map the same account options must be configured in each level of the policy.

Bandwidth

The bandwidth command is used to guarantee a minimum service rate to a class.

Syntax description:

To configure in Kbps:

[**no**] **bandwidth** *rate* [**account** *account options*]

To configure as a percentage of visible bandwidth:

[**no**] **bandwidth percent** *value* [**account** *account options*]

Command Default:

By default there is no minimum bandwidth value configured in the schedule entry for a queue. Note that the default excess weight does guarantee some minimum service.

Usage Guidelines:

The **bandwidth** command may be useful if you have an application for which you know the minimum bandwidth requirements.

Bandwidth rates can be configured in 8Kbps increments and the ASR1K has been tested to achieve accuracy within 1% of those rates.

The **bandwidth** command is only supported in leaf schedules (class layer schedules). If you wish to apportion bandwidth in a parent policy you may use the **bandwidth remaining** command.

If you wish to replicate scheduling behavior of an IOS Classic platform (2 parameter scheduler) you may want to replace all **bandwidth percent** *value* commands in your configuration with **bandwidth remaining percent** *value* command.

Bandwidth remaining

The **bandwidth remaining** command is used to apportion excess bandwidth between classes. It may be configured as a simple weight or as a percentage of available bandwidth.

Syntax Description:

To configure as a simple weight:

[**no**] **bandwidth remaining ratio** *value* [**account** *account options*]

To configure as a percentage:

[**no**] **bandwidth remaining percent** *value* [**account** *account options*]

Command Default:

By default every bandwidth schedule entry, whether in leaf schedule or a parent schedule, is configured with an excess weight of 1. This is equivalent to **bandwidth remaining ratio** *1* being configured in that class.

Usage Guidelines:

Configuring bandwidth remaining as a weight supports values of 1 to 1000. This can allow more granular excess sharing than using the percent option.

Configuring **bandwidth remaining percent** *value* yields behavior similar to IOS classic which used a 2 parameter scheduler.

With a shape on parent / queue on child policy (parent has only class default) **bandwidth remaining ratio** *value* should be used to apportion bandwidth between logical interfaces where parent polices are attached

Fair-Queue

The **fair-queue** command is used to configure flow based fair-queuing in a class configured as a bandwidth queue.

Syntax Description:

fair-queue

Command Default:

By default a single fifo queue is configured for each bandwidth class.

Usage Guidelines:

Flow based fair-queuing is used to ensure a single greedy flow can't consume all the bandwidth allocated to a class.

All packets from any given flow are hashed into the same flow queue.

Flow-queuing should not be configured in a policy attached to a tunnel interface. Since all packets have the same outer header all packets are hashed to the same flow queue thus rendering the feature ineffective.

Priority

The **priority** command is used to give low latency and low jitter treatment to a class of traffic.

Syntax Description:

To configure an absolute priority queue (note should be used with explicit policer)

[**no**] **priority**

To configure an absolute priority queue with multi-level priority queuing (note this should be used with an explicit policer)

[**no**] **priority level** *1* | *2*

To configure a priority queue with conditional policer

[**no**] **priority** *rate in kbps* [**burst in bytes**]

or

[**no**] **priority percent** *rate* [**burst in bytes**]

To configure multilevel priority queueing with conditional policer

[**no**] **priority level** *1* | *2rate in kpbs* [**burst in bytes**]

or

[**no**] **priority level** *1* | *2* **percent***rate* [**burst in bytes**]

Command Default:

By default queues are not configured with priority treatment.

Usage Guidelines:

Priority queues should be used with some form of queue admission control (explicit policer or conditional policer) to avoid chance of starving other classes of service.

The policer conforming burst should be configured to an appropriate value for the application in the queue. The following is a configuration example

```
policy-map always_on_policer_burst_example
    class voice
      priority
      police cir 2000000 1250
```

It is not necessary to configure priority in the parent of a hierarchical policy as priority propagation will ensure packets marked as priority by a leaf schedule will receive priority treatment throughout the scheduling hierarchy.

Shape

Use the **shape** command to configure the maximum rate at which a queue may be serviced. Configuring a shaper does not guarantee throughput to a class, it simply puts an upper bound on the rate at which that class may be serviced.

Syntax Description:

[**no**] **shape average** *rate* [**unit**] [**confirming burst**] [**excess burst**] [**account** *options*]

or

[**no**] **shape average percent** *rate* [**confirming burst**] [**excess burst**] [**account** *options*]

Command Default:

By default there is no maximum rate configured in the schedule entry for a bandwidth queue.

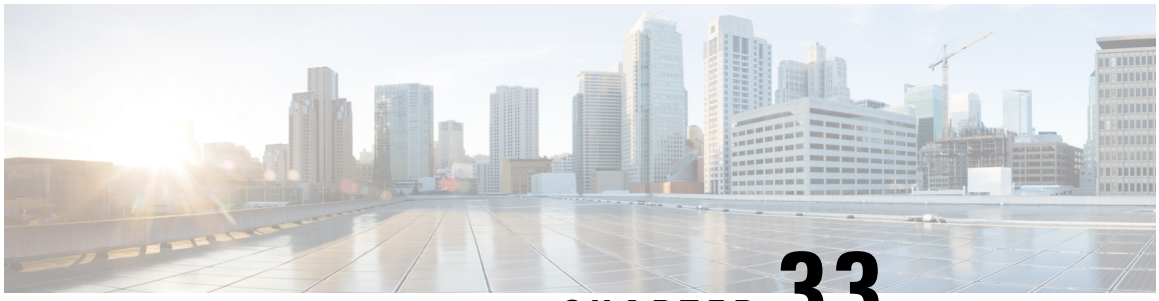
Usage Guidelines:

The **shape** command is most commonly used in a parent policy to limit the rate at which traffic is sent to a remote site.

When used in a parent policy the shape rate will be enforced for all traffic - priority and bandwidth.

The **shape** command has options for conforming and excess burst sizes but these values have no effect on XE platforms. Hardware scheduling on the ASR1K platform obviates the need to optimize burst parameters.

The **shape** command enforces a maximum rate at which a class may be serviced but does not in itself guarantee any throughput to that class. The **bandwidth remaining** command may be used along with the shape command to guarantee throughput.



CHAPTER 33

QoS Hierachical Scheduling

In this chapter we will see how commands and their semantics as covered in the scheduling chapter can be combined in different ways to achieve more complex outcomes.

Two distinct approaches are available to configure complex scheduling hierarchies: hierarchical policy-maps and policy-maps attached to logical interfaces. Here, we illustrate how either can achieve the same outcome and delineate the relative benefits of each approach.

About Hierarchical Schedules

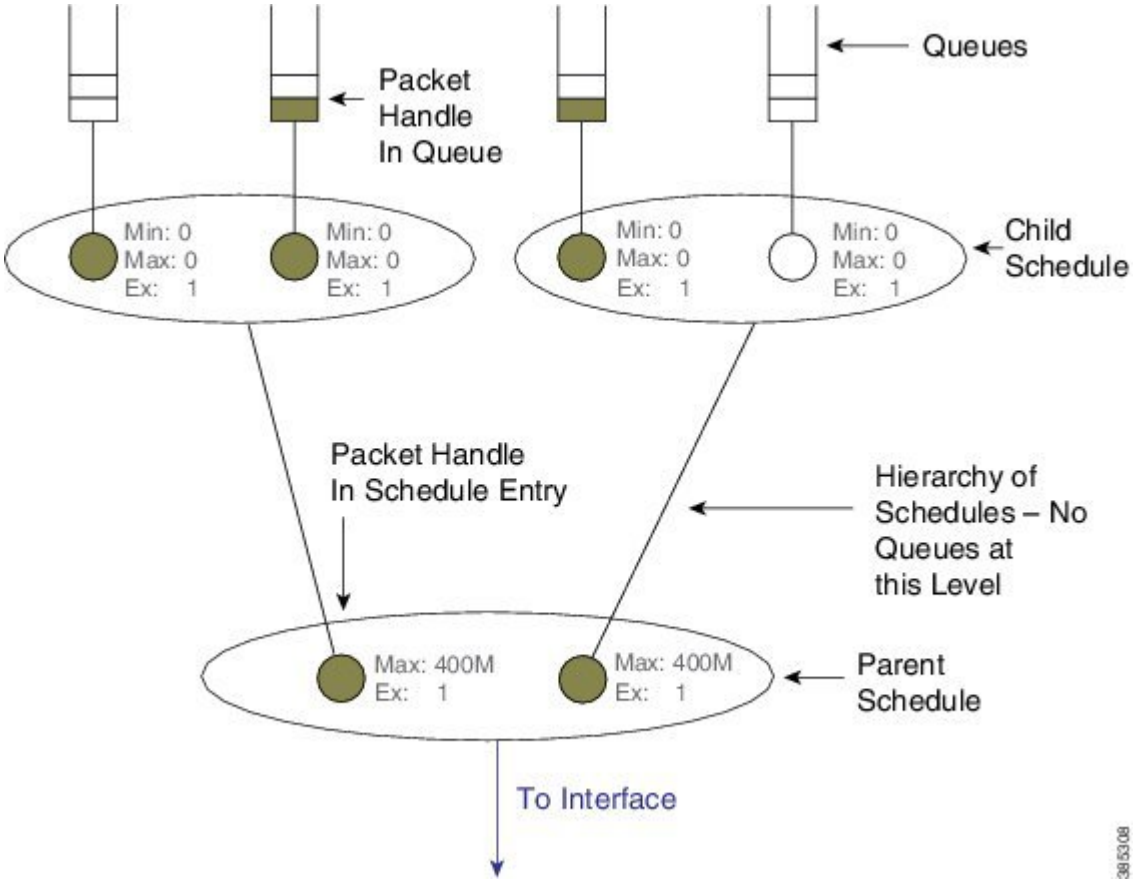
Definitions

We assume that you are now familiar with the role of a schedule and how a schedule entry contains information (packet handle, class queues, etc.) on how the child of that entry should be treated (see the Definitions, on page 357 discussion in the scheduling chapter). Here, we build upon that discussion.

The fundamental difference between what we show here and in the previous chapter is the *child schedule*, which may be a queue or another schedule. Hierarchical scheduling allows you to build complex structures with bandwidth sharing at multiple layers.

The following figure shows the basic hierarchical scheduling structure:

Figure 36: Hierarchical Scheduling Definitions



The first thing to notice from the diagram is that we implement a hierarchy of schedules and not a hierarchy of queues. This means that queues exist only at the *leaf layers* of the hierarchy and that packet handles (the packet representation vehicle) never move from queue to queue. Instead, a single packet handle is loaded into the *parent schedule* entry (provided a packet is waiting for transmission).

When detailing a scheduling hierarchy we describe schedules as parent or child (or indeed grandchild). These descriptions are relative. A parent schedule is one closer to the root of the hierarchy (closer to the interface). The child of a schedule could be either a schedule or a queue. We may also refer to schedules as a leaf or non-leaf schedule. A *leaf schedule* has solely queues as children; a *non-leaf schedule* will have at least one schedule as a child.

Looking at the diagram you can see that the schedule entry in the parent schedule (non-leaf)) has only two parameters per schedule entry. The Minimum Bandwidth parameter is only supported in leaf schedules – not in non-leaf schedules.

Scheduling Decisions - Root to Leaf

The following sequence of diagrams illustrates how the schedules in a hierarchy work in concert yet make local decisions when selecting the next packet to send through an interface.

Among the packets stored locally, the parent schedule will first decide on the most eligible packet to forward to the interface. After sending the associated packet handle, it will have a free spot in its own schedule entry – no packet handle from the child of that entry exists.

If the child is another schedule, the parent will send it a *pop* (a message that communicates "you pick your most eligible packet and send me that packet handle"). The child schedule will review the configuration of each entry, decide which packet should be sent next, and forward that packet handle to the parent schedule. The child will now have a free spot in its schedule entry. As the child is a queue no decision is necessary - the packet handle at the head of the queue will be loaded into the (child) schedule entry:

Figure 37: Scheduling Decision - Root to Leaf: Steps 1-2

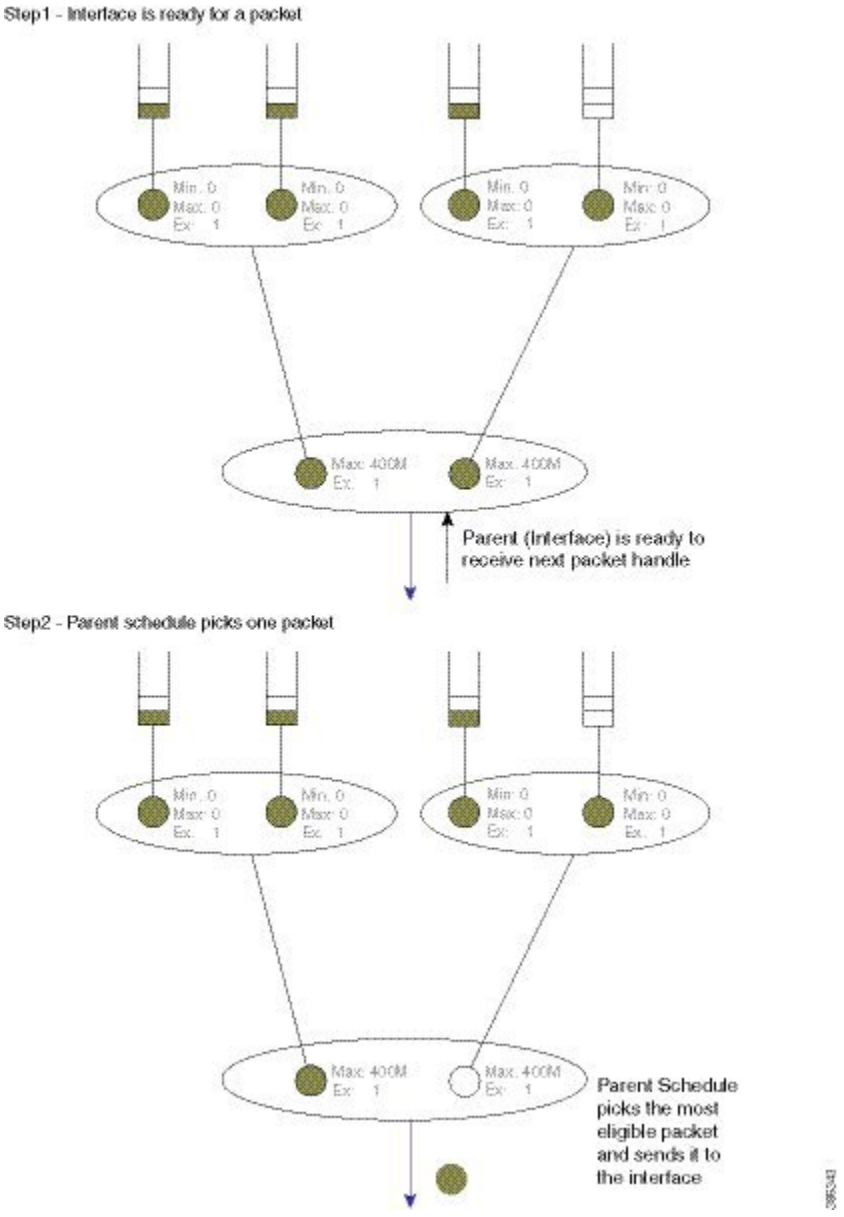


Figure 38: Scheduling Decision - Root to Leaf: Steps 3-4

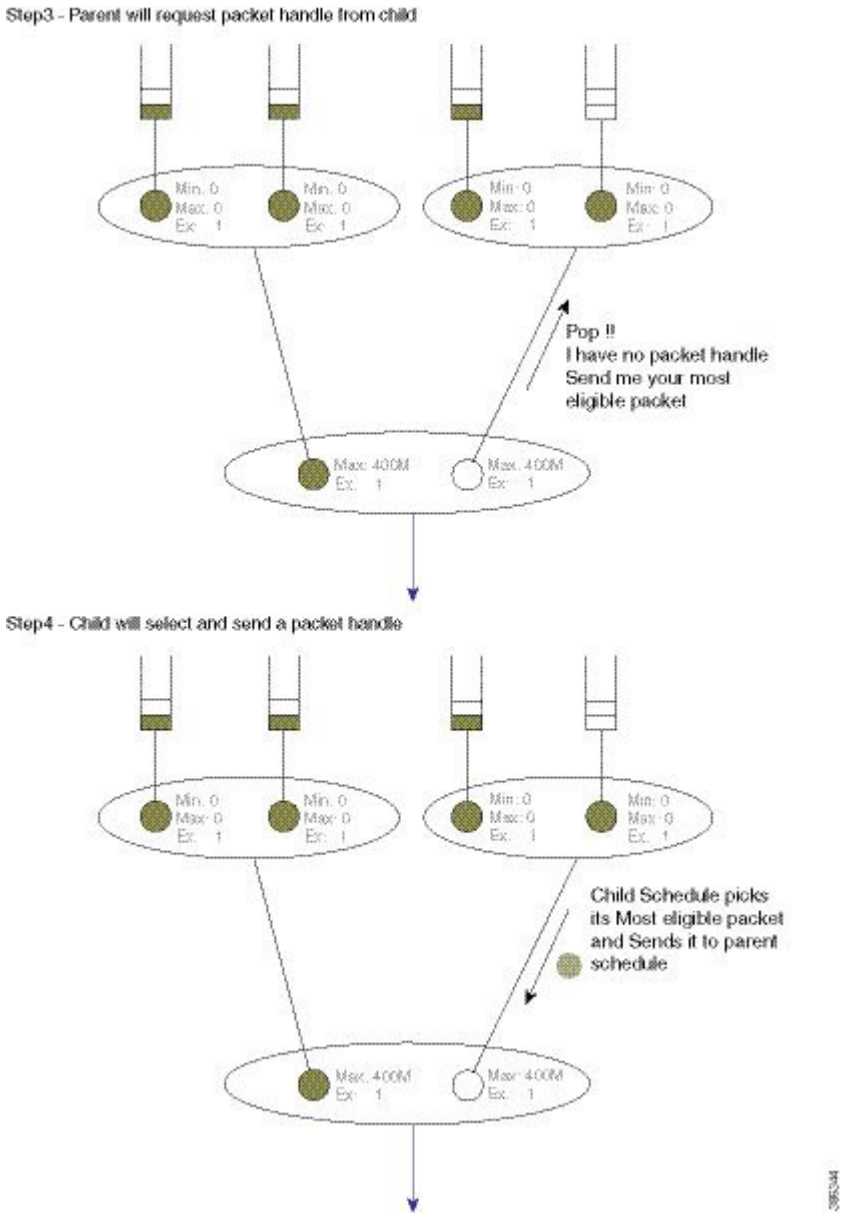
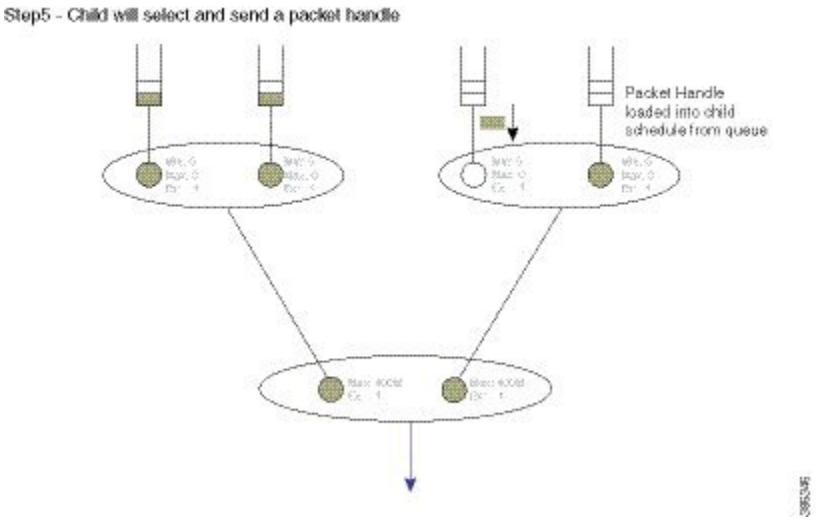


Figure 39: Scheduling Decision - Root to Leaf: Step 5

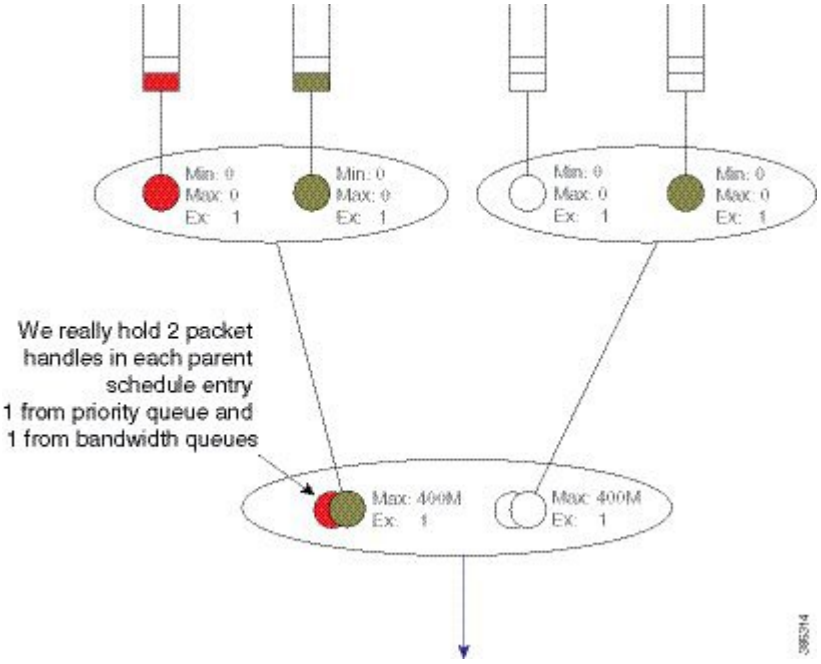


Concept of Priority Propagation

You will notice that thus far the descriptions have been somewhat simplistic in that they have only included bandwidth queues. In truth, for each child, a parent schedule can hold a *priority (queue) packet handle* and a *bandwidth (queue) packet handle* (we term this capability *passing lanes*). When a packet handle is sent from a child schedule to the parent we indicate whether it arose from a priority or a bandwidth class and we also indicate the priority level (we term this behavior *priority propagation*).

We will examine priority propagation later in this chapter. Here we merely introduce the concept so that the rules of hierarchical scheduling make sense:

Figure 40: Parent Schedule can hold Priority and Bandwidth Handles (Passing Lanes)



Observe in this hierarchy that priority service does not require configuration in the parent schedule entry; the (parent) schedule entry has only two parameters, Max and Excess Weight.

Hierarchical Scheduling Operation

In the scheduling chapter we describe how scheduling decisions are made for a flat policy attached to a physical interface. Here, we describe the scheduling rules for a leaf schedule (a scenario addressing a schedule with only queues as children). Those rules still hold.

We will now expand on that description to include the rules for the parent-child interaction:

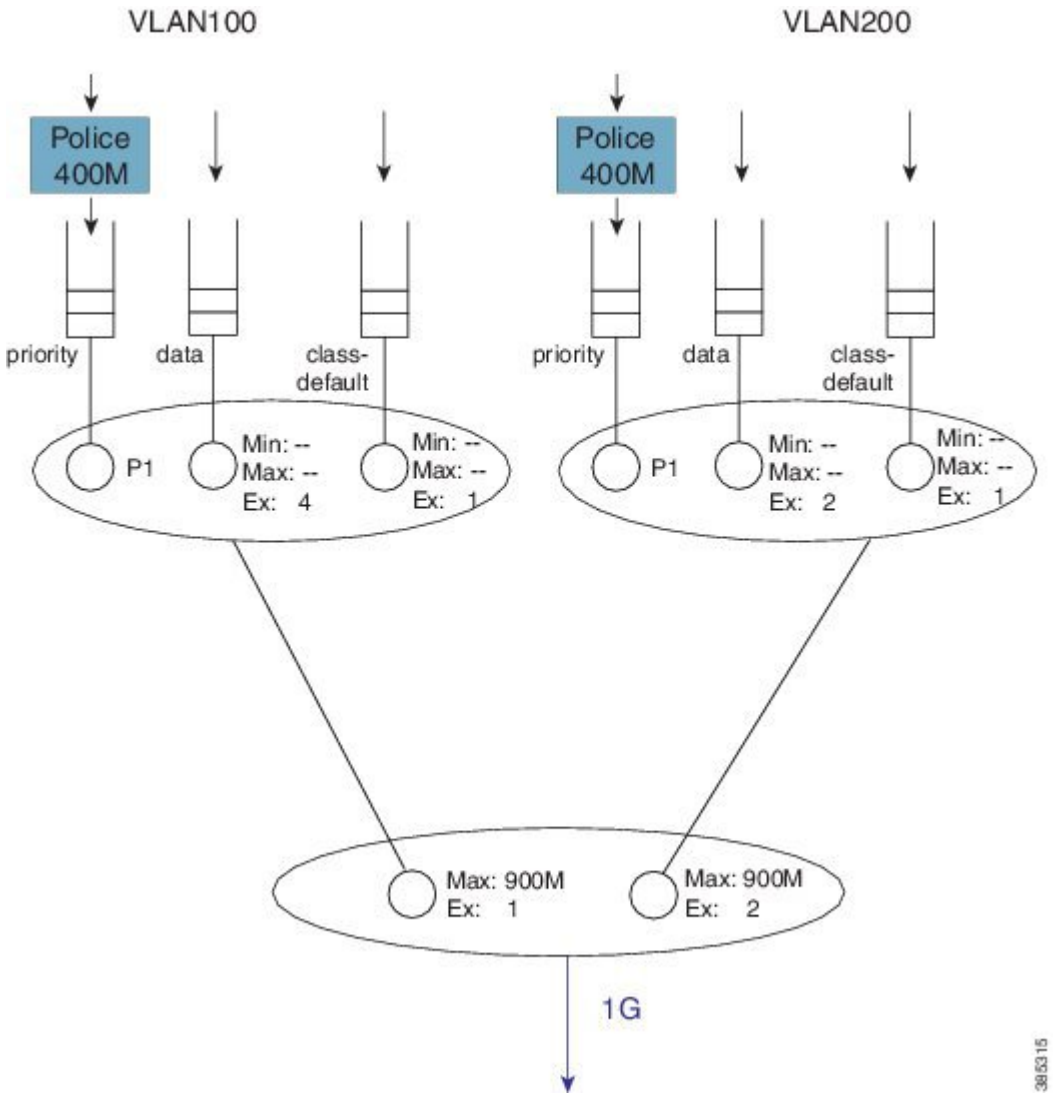
- Priority traffic counts towards Max (**shape** command) configured at the parent schedule.
- Priority traffic is unaltered by Ex (**bandwidth remaining ratio** command) configured at parent.
- Priority packets at the parent schedule will always be scheduled before bandwidth packets.
- Priority will be scheduled proportionally to the shape rate configured at parent. We include this point for completeness; it should not be a factor unless your priority load can oversubscribe the interface.
- Under priority propagation, a parent will know that a packet came from a priority queue but it will not know whether it was P1 (priority level 1) or P2 (priority level 2).
- Traffic from queues configured with the **bandwidth** or the **bandwidth remaining** commands are treated equally at the parent (*no min bandwidth propagation*). Henceforward, we refer to traffic from any bandwidth queue as *bandwidth traffic*.
- Excess weight configuration at the parent controls the fairness between bandwidth traffic from multiple children competing for any physical bandwidth not consumed by priority traffic.

To understand these rules, let's look at the following configuration example. Later, we will detail how a configuration is mapped into a datapath configuration. For now, the diagram and schedule entries shown in the diagram are sufficient to understand the behavior:

```
policy-map child100
  class priority
    priority
    police cir 400m
  class data
    bandwidth remaining ratio 4
!
policy-map parent100
  class class-default
    shape average 900m
    service-policy child100
!
policy-map child200
  class priority
    priority
    police cir 400m
  class data
    bandwidth remaining ratio 2
!
policy-map parent200
  class class-default
    shape average 900m
    bandwidth remaining ratio 2
    service-policy child200
!
int g1/0/4.100
  encaps dot1q 100
  service-policy out parent100
!
int g1/0/4.200
  encaps dot1q 200
  service-policy out parent200
```

The following diagram shows the scheduling hierarchy associated with the previous configuration. As described previously, the **shape** command in the parent policy(s) sets the Max parameter and the **bandwidth remaining ratio** command sets the Ex parameter in the schedule entry (rules 1 and 2). The latter defaults to 1 if not explicitly set:

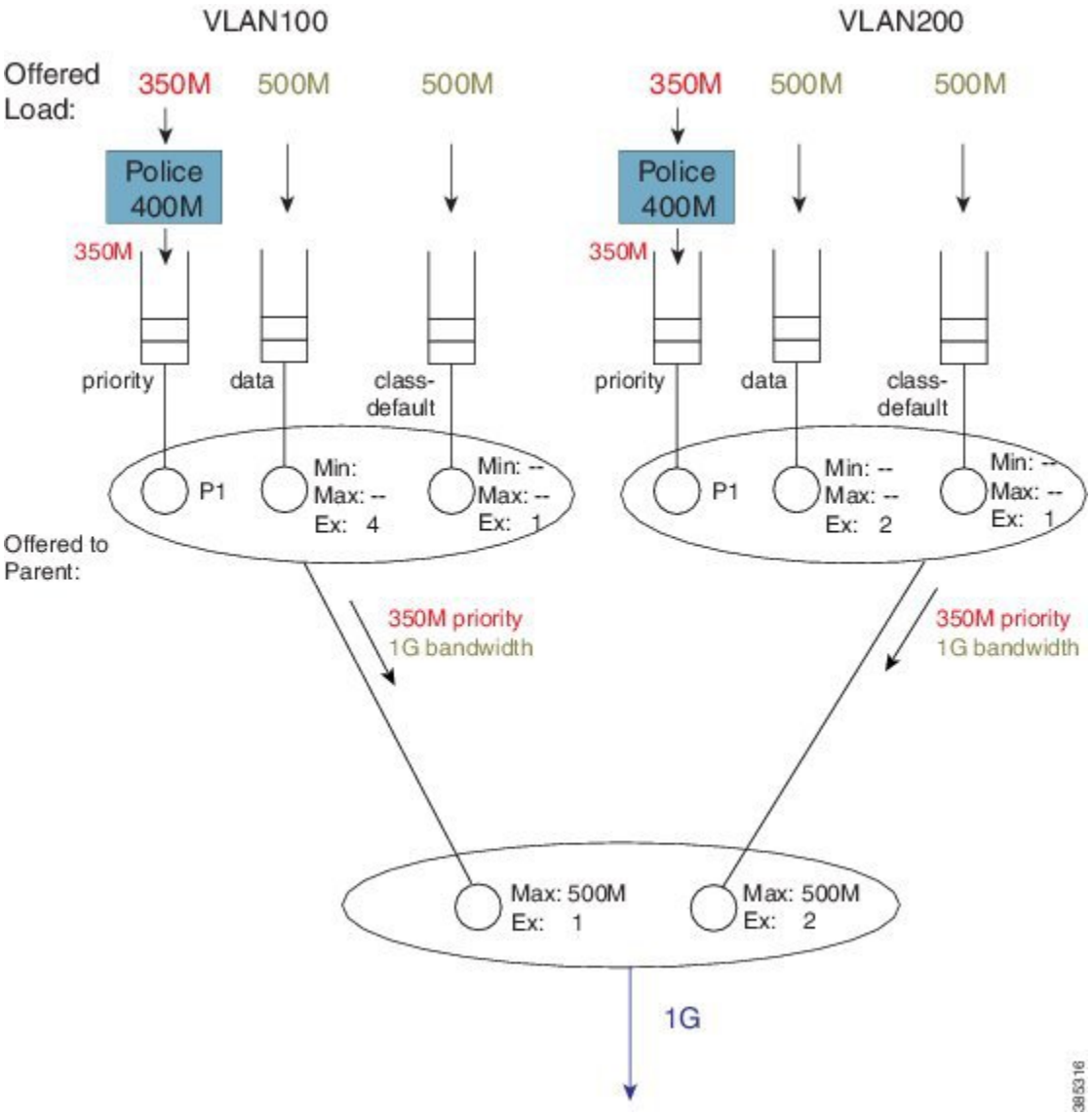
Figure 41: Scheduling Hierarchy Example - Forwarding the Entire Offered Priority Load



Let's now look at the expected throughput for an offered load.

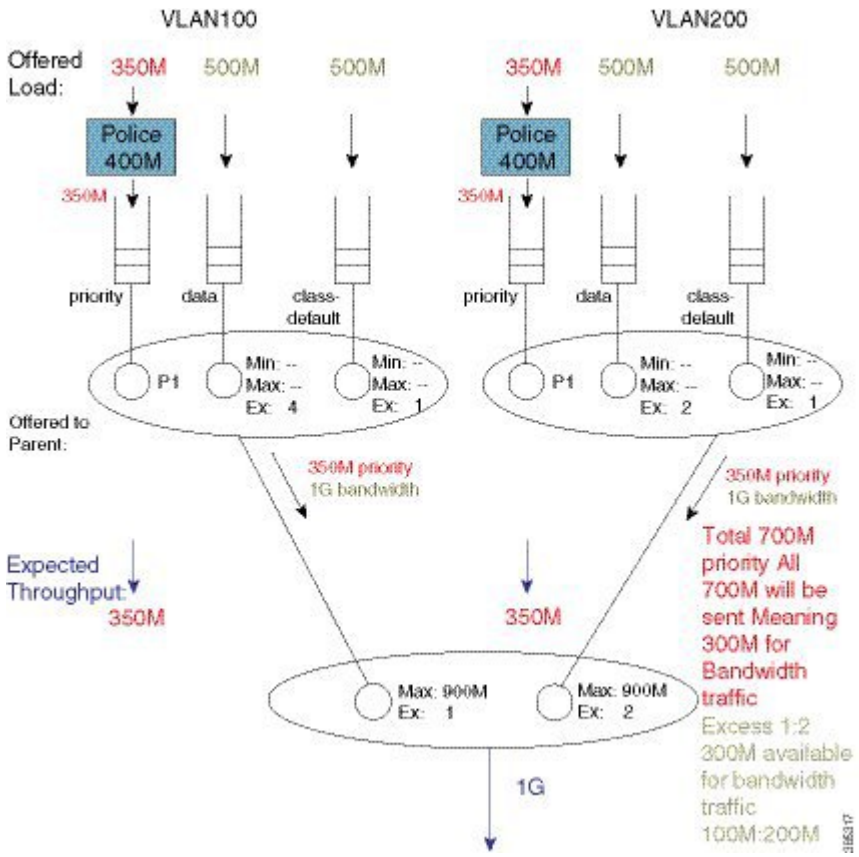
Note The following examples ignore overhead accounting – they are intended solely to illustrate how to calculate expected throughput independent of minor details.

Figure 42: Calculating What is Offered to the Parent from Each Child Schedule



In calculating expected throughput, the first step is to eyeball the offered load per class. The next step is to aggregate them and observe the total loads from priority and bandwidth classes that will be offered to the parent:

Figure 43: Calculating the Remaining Bandwidth for Bandwidth Queues



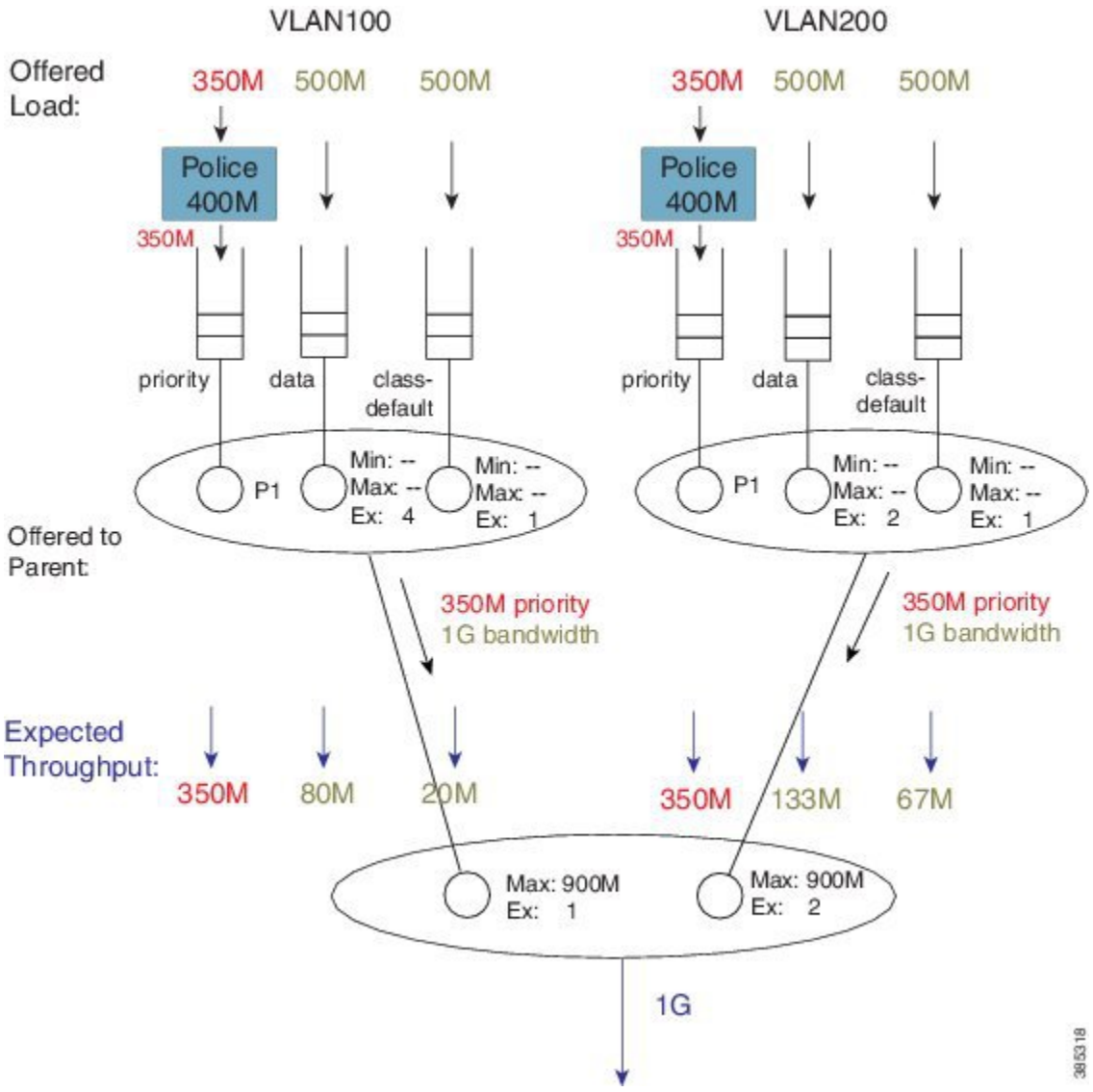
Each child schedule is offering 350 Mbps of priority traffic to the parent. Because the interface has 1 Gbps of available bandwidth it will forward the entire 700 Mbps offered priority load.

According to rule 3, we schedule priority traffic before bandwidth traffic. As the Max (rate) for each parent schedule entry exceeds the offered priority load from that entry's child schedule, we forward the entire 350 Mbps of traffic.

With the scheduled priority load (350 Mbps + 350 Mbps of traffic), we can now calculate the (remaining) bandwidth for bandwidth (queue) traffic (300 Mbps or 1 Gbps of total bandwidth available - 700 Mbps consumed by priority load).

The parent schedule will use the Ex configuration to apportion the 300 Mbps (remaining) bandwidth. With Ex values of 1 and 2, for VLAN100 and VLAN200, respectfully, the bandwidth will be shared 1:2. VLAN100 will receive 100 Mbps and VLAN200 will receive 200 Mbps of bandwidth traffic throughput:

Figure 44: Bandwidth Sharing based on the Excess Weights in the Child Schedule



To calculate how this 100 Mbps will be apportioned, we can now examine the bandwidth queue's schedule entries (in the schedule) for VLAN100.

No Min guarantees are configured (the **bandwidth** command is not supported in parent schedules), so all sharing hinges on the scheduled Ex values in the child schedule. Based on the settings (4 for class data and 1 for class class-default) the 100 Mbps will be shared 4:1 (class data receives 80 Mbps; class class-default receives 20 Mbps).

If we follow the same approach for VLAN200, the 200 Mbps available is split 2:1. Class data will receive 133 Mbps and class class-default will receive 67 Mbps.

You probably noticed that every class was oversubscribed. This means the expected throughput we calculated was also the minimum guaranteed service rate for each class. Under hierarchical scheduling, bandwidth sharing at the parent schedule ensures that we don’t waste bandwidth if any child schedule does not have packets

waiting for transmission. Similar to bandwidth sharing in flat policies, bandwidth unused by one child is available to others.

Priority Propagation

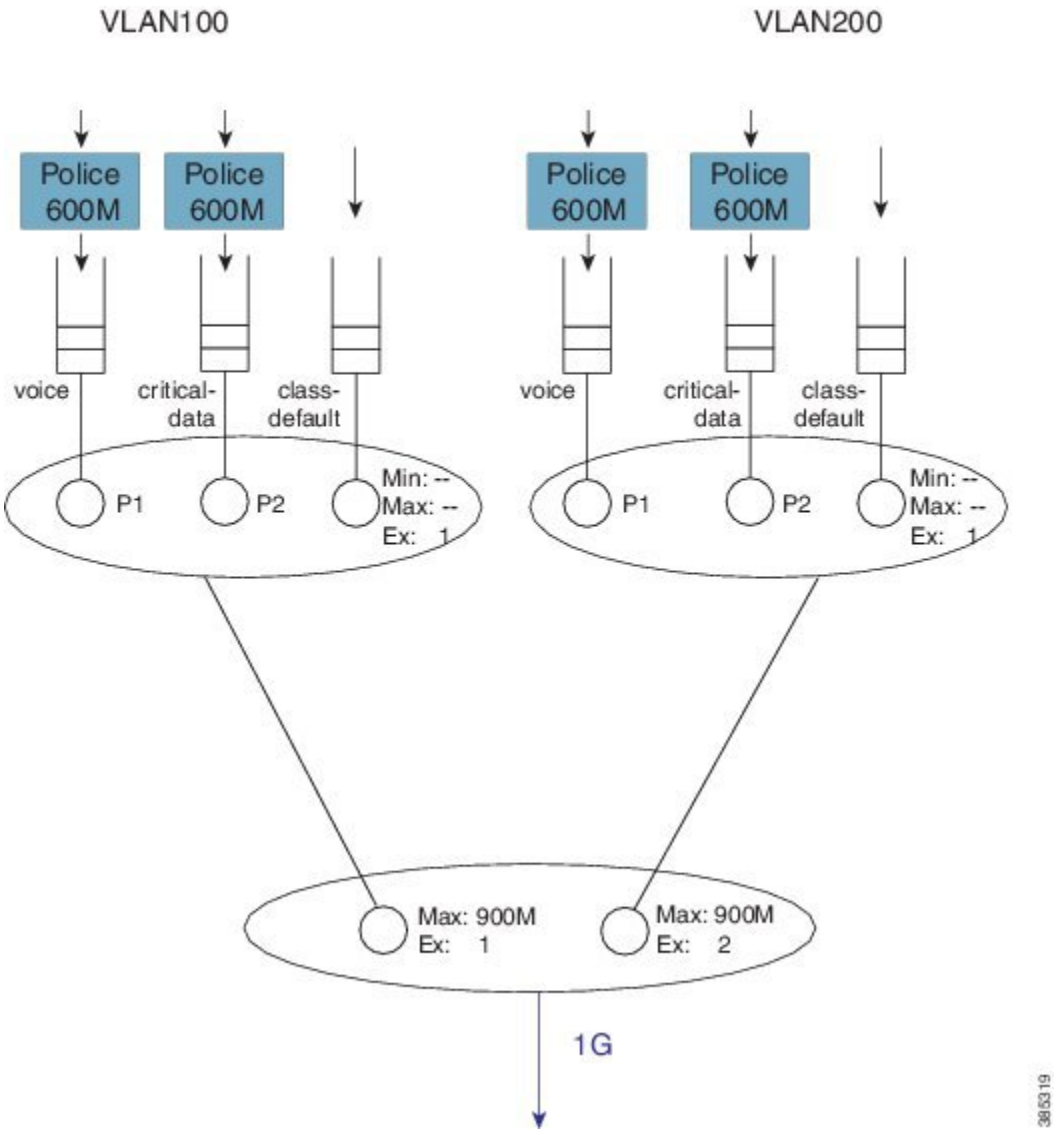
Regarding the Concept of Priority Propagation, on page 409, we will now use the following sample configuration to highlight a few points:

```
policy-map child
  class voice
    priority
    police cir 600m
  class video
    priority
    police cir 600m
!
policy-map parent100
  class class-default
    shape average 900m
    service-policy child
!
policy-map parent200
  class class-default
    shape average 900m
    bandwidth remaining ratio 2
    service-policy child
!
int g1/0/4.100
  encaps dot1q 100
  service-policy out parent100
!
int g1/0/4.200
  encaps dot1q 200
  service-policy out parent200
```

Note We are using the same child policy in both parent policy-maps. Unique policy-maps are unnecessary at any level; if the requirements match, you can share child or even parent policy-maps.

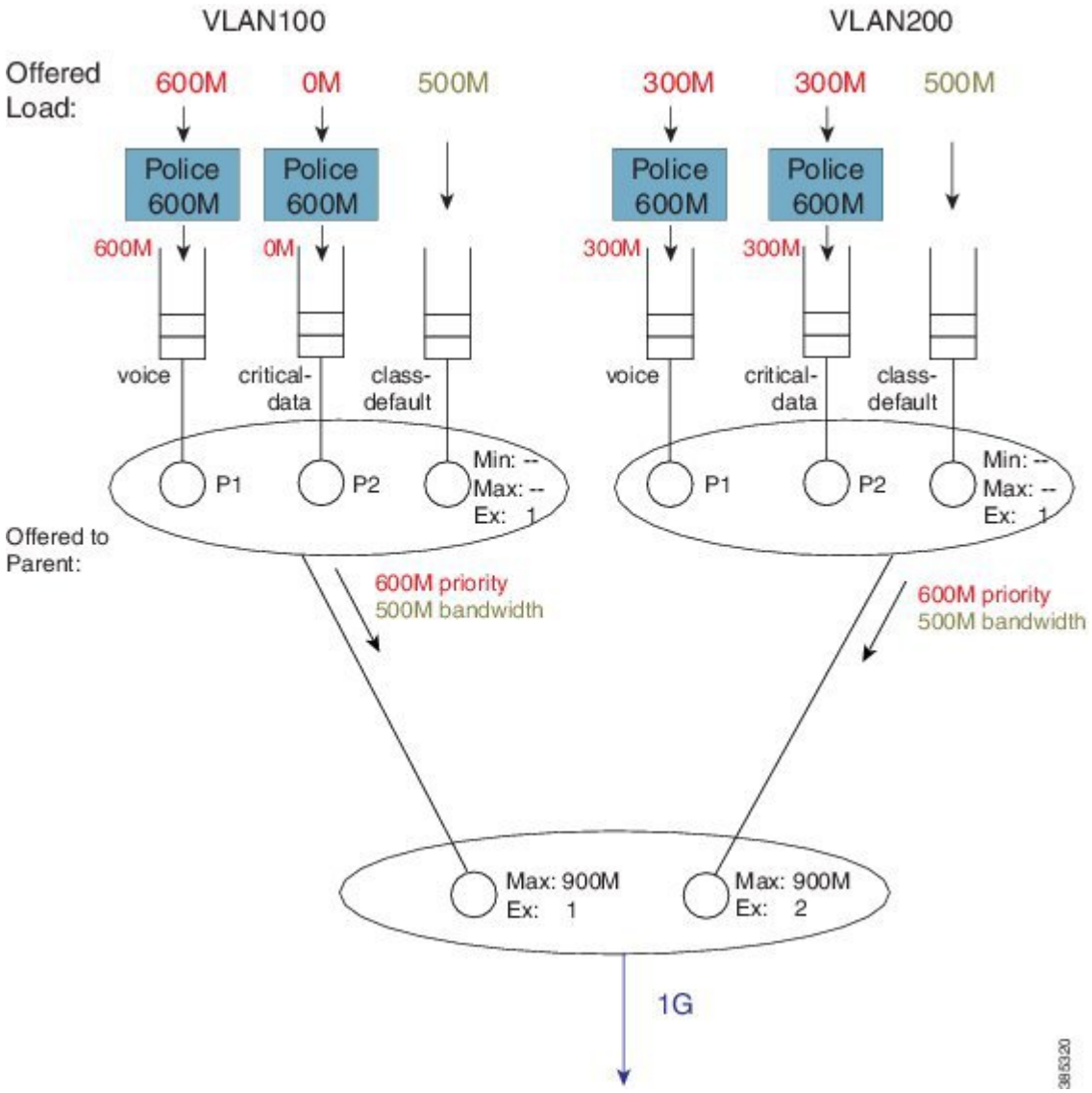
A hierarchy created for this configuration would look as follows:

Figure 45: Scheduling Hierarchy Example - Multi-level Priority Queuing in Child Schedule



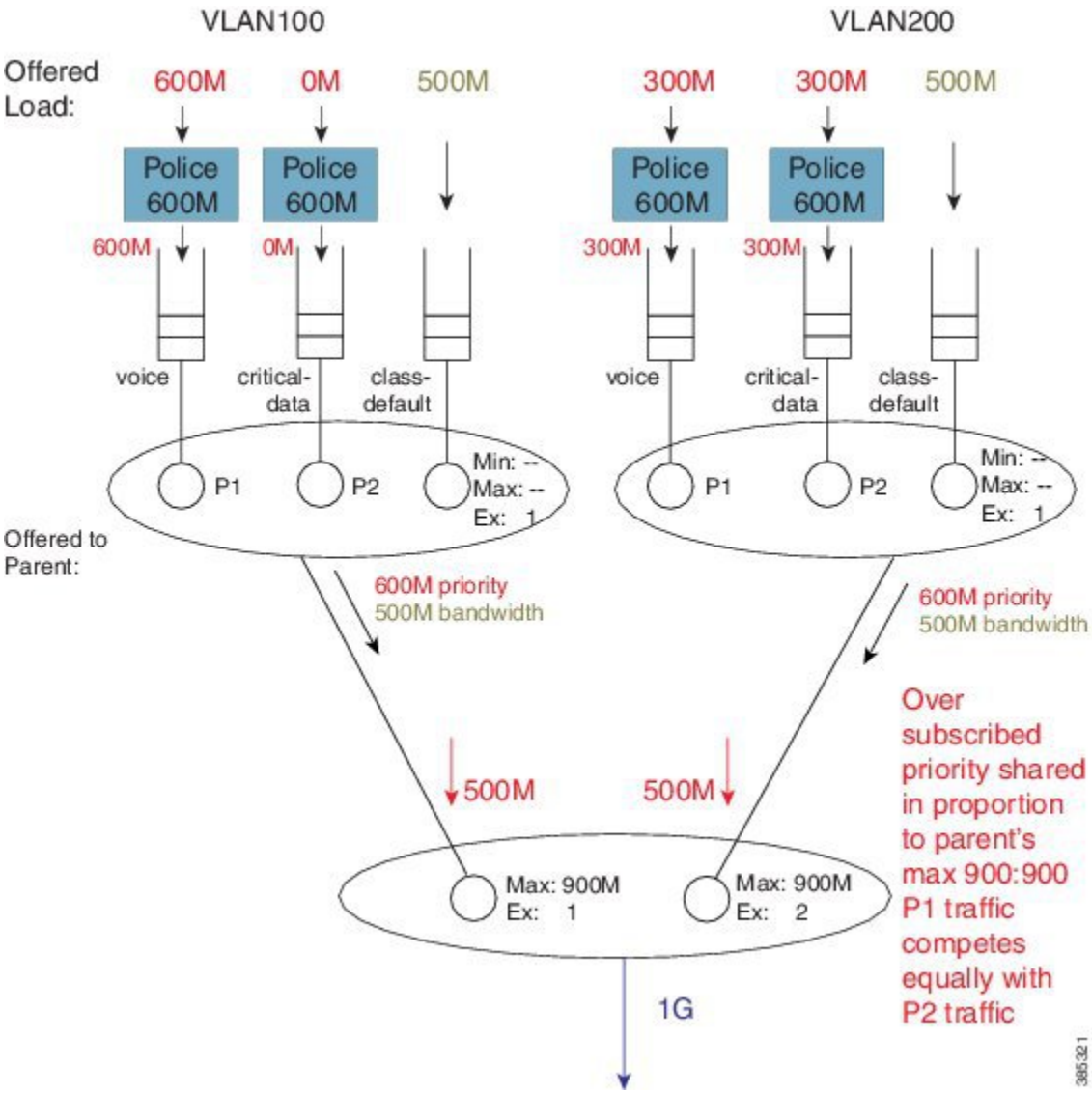
The scenario differs from that in the Concept of Priority Propagation, on page 409. We now have multi-level priority queuing in the child schedule (e.g., P1 [priority level 1] and P2 [priority level 2] classes). The following diagram shows the load offered to each class:

Figure 46: Multi-level Priority Queuing - Load Offered to each Class



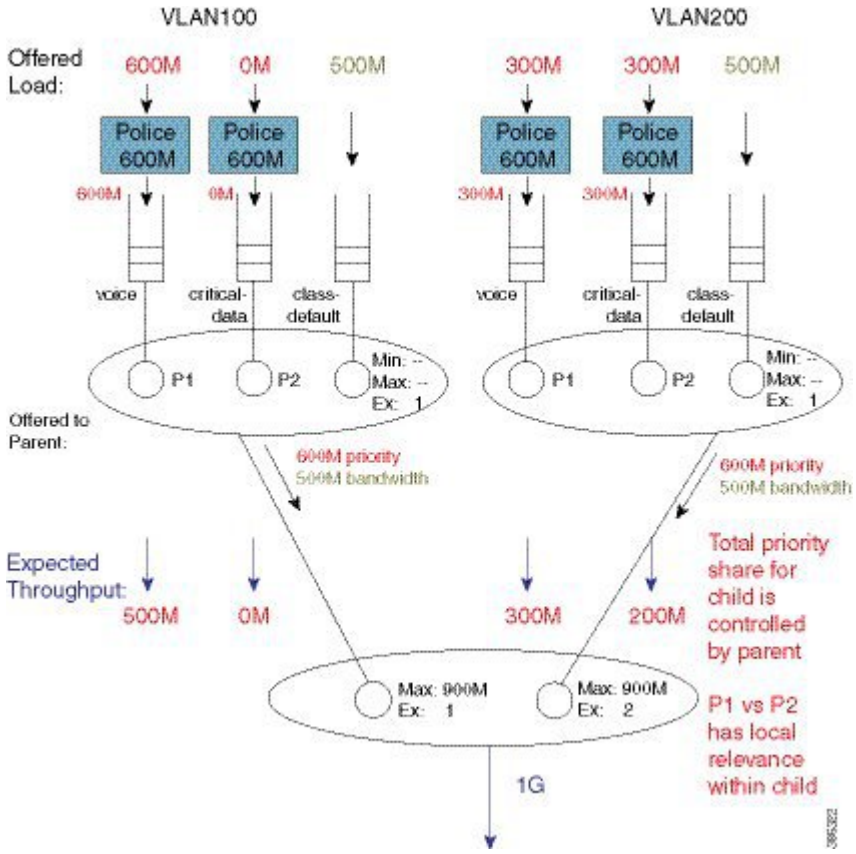
Now let's look at the total load offered from the priority and bandwidth queues (for each child) to the parent:

Figure 47: Oversubscribed Priority Queues shared relative to the Parent's Max ratio



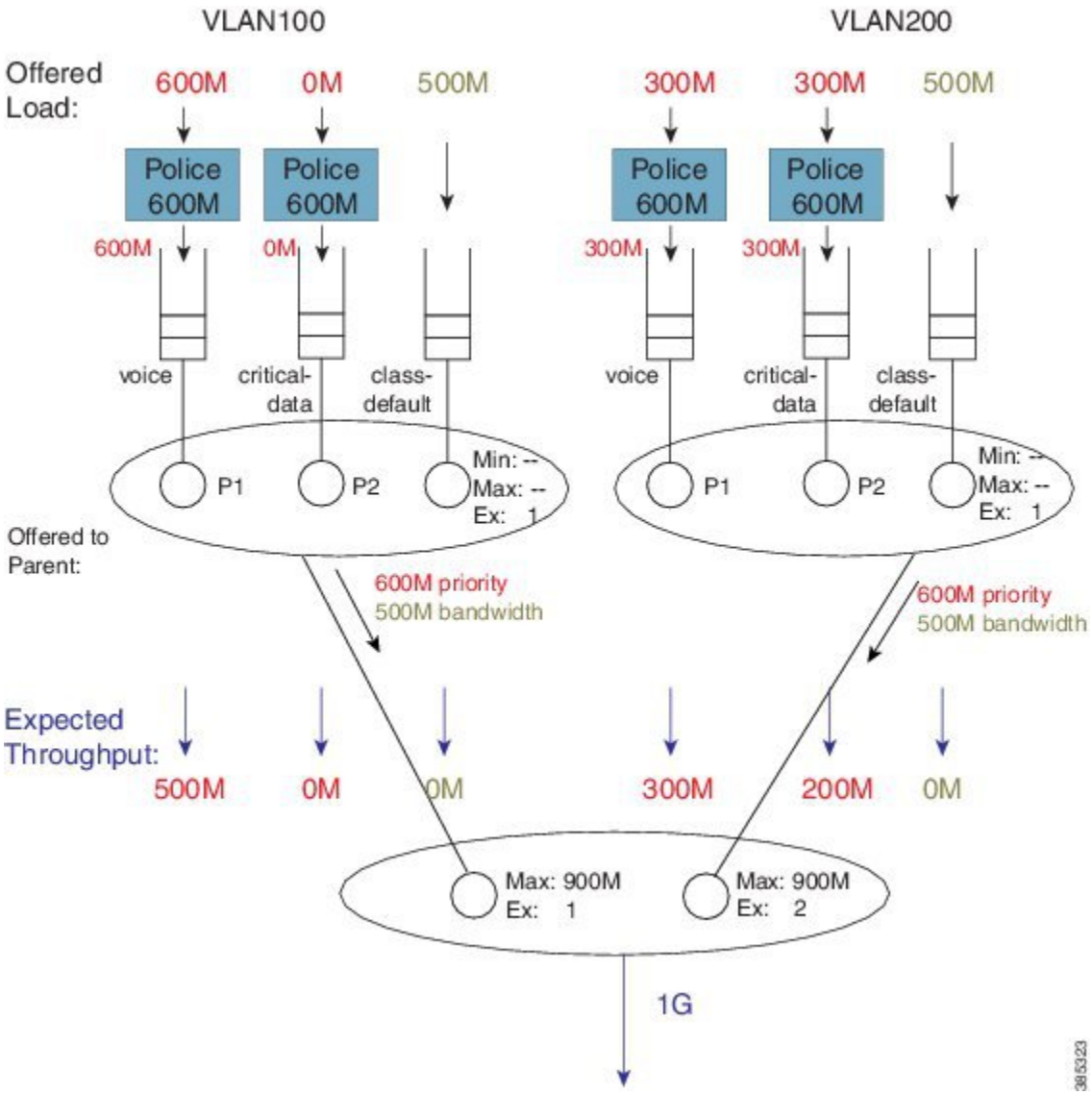
In rule 4 (see Hierarchical Scheduling Operation, on page 410) we stipulated that a parent will schedule an offered priority load *proportional to the shape rate configured in its schedule entry*. Here, each child has a Max rate ("shape" in the parent policy) of 900 Mbps and offers 600M priority traffic (i.e. 1.2 Gbps [600M + 300M + 300M traffic] when only 1 Gbps is available). The parent schedule will apportion 500 Mbps to each child. The key point to note here is that P1 from VLAN100 competes equally with P2 traffic from VLAN200. (Recall from rule 5 that priority propagation alerts the parent that a packet arose from a priority queue but does not indicate the priority level.)

Figure 48: Parent Controls Total Priority Share for Child



The parent schedule accepts 500 Mbps of priority load from VLAN200. The child schedule is responsible for apportioning bandwidth within that 500 Mbps. The child policy has P1 configured in the voice class, which means that the child schedule will always pick packets from that queue first (i.e., priority levels have local significance within a schedule). The expected throughput for the voice class in VLAN200 is 300 Mbps. The class critical-data will receive 200 Mbps (the unused share of the 500 Mbps – 300 Mbps in this example):

Figure 49: Child Schedule Apportions Bandwidth received from Parent Schedule



What about the expected throughput from the bandwidth queues? As the offered priority load exceeded the physical bandwidth available, nothing remained for the bandwidth queues. This example effectively highlights that priority classes can starve bandwidth queues completely. If control packets are not in priority queues, you might experience network instability. In fact, failure to place control packets in priority queues could be considered a misconfiguration!

Note Ensure that the physical bandwidth available exceeds the sum of all priority class policers, so that the latter can't starve others of service.

Please be aware that the concept of priority propagation does not end in the scheduling hierarchy. When we mark a packet as stemming from a priority class, that tag is carried to the egress interface. In egress carrier or

interface cards, we find multiple places where passing lanes enable priority packets to arrive at the interface as quickly as possible.

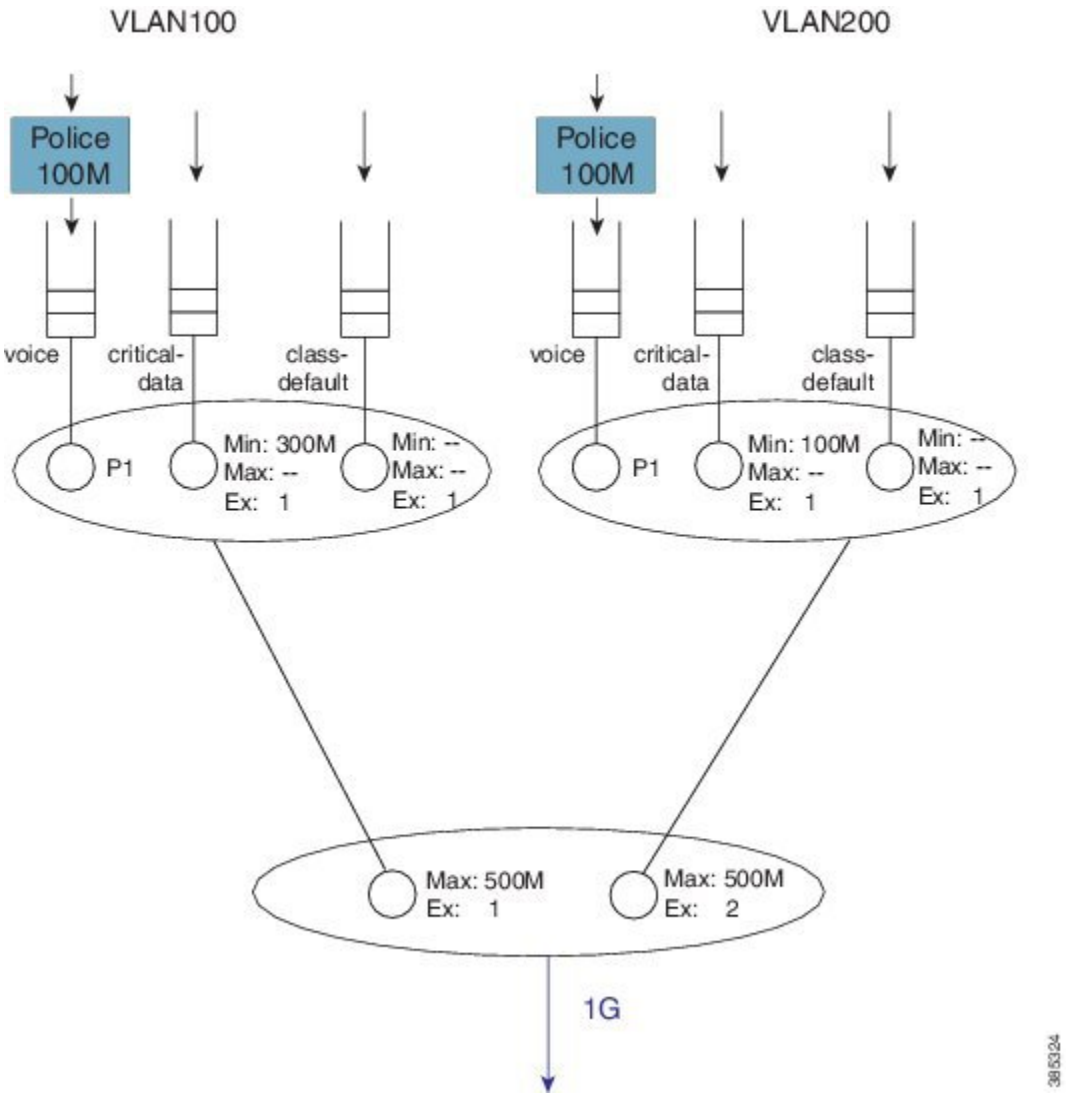
Bandwidth Command in Leaf Schedules

We have already stated that although the **bandwidth** command is not supported in parent schedules (and so a Min setting is absent), it is supported in leaf schedules. With the following configuration, we will explain the operation of the **bandwidth** command in a child policy-map. (Lines flagged with asterisks indicate how this configuration compares with that presented in priority propagation.

```
policy-map child100
  class voice
    priority
    police cir 100m
  class critical-data
    bandwidth 300000                  ****
!
policy-map child200
  class voice
    priority
    police cir 100m
  class critical-data
    bandwidth 100000                 ****
!
policy-map parent100
  class class-default
    shape average 500m
    service-policy child100
!
policy-map parent200
  class class-default
    shape average 500m
    bandwidth remaining ratio 2
    service-policy child200
!
int g1/0/4.100
  encaps dot1q 100
  service-policy out parent100
!
int g1/0/4.200
  encaps dot1q 200
  service-policy out parent200
```

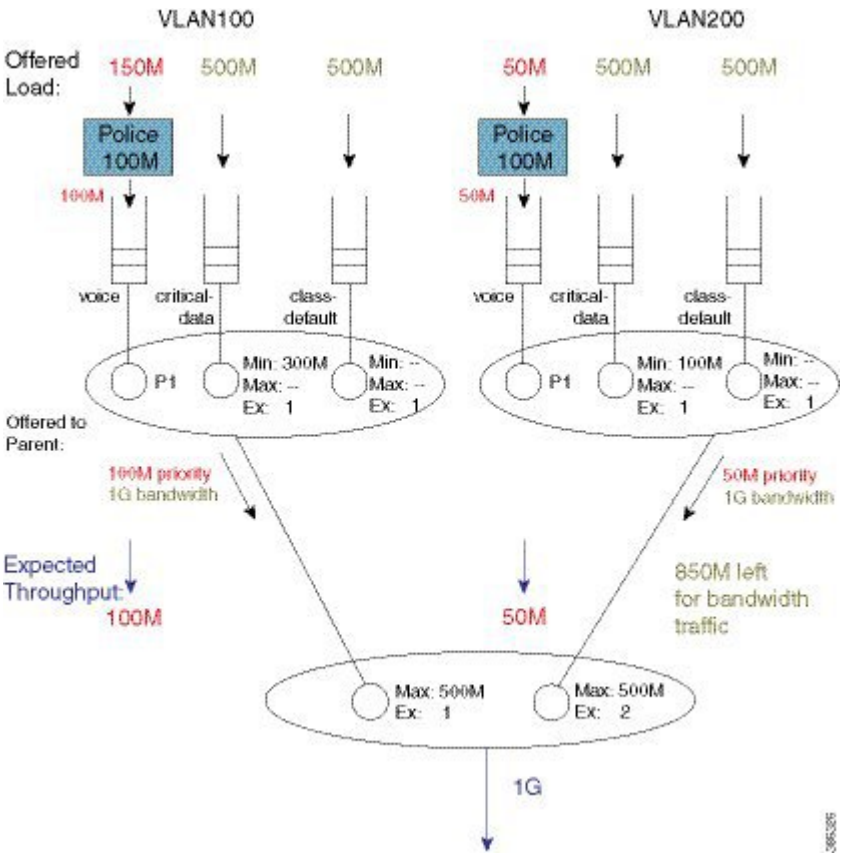
A hierarchy created for this configuration would look as follows:

Figure 50: Scheduling Hierarchy Example - Bandwidth Command Application in Leaf Schedules



To explain the operation of this hierarchy let's consider the following offered loads (to each class):

Figure 51: Load Offered to each Class

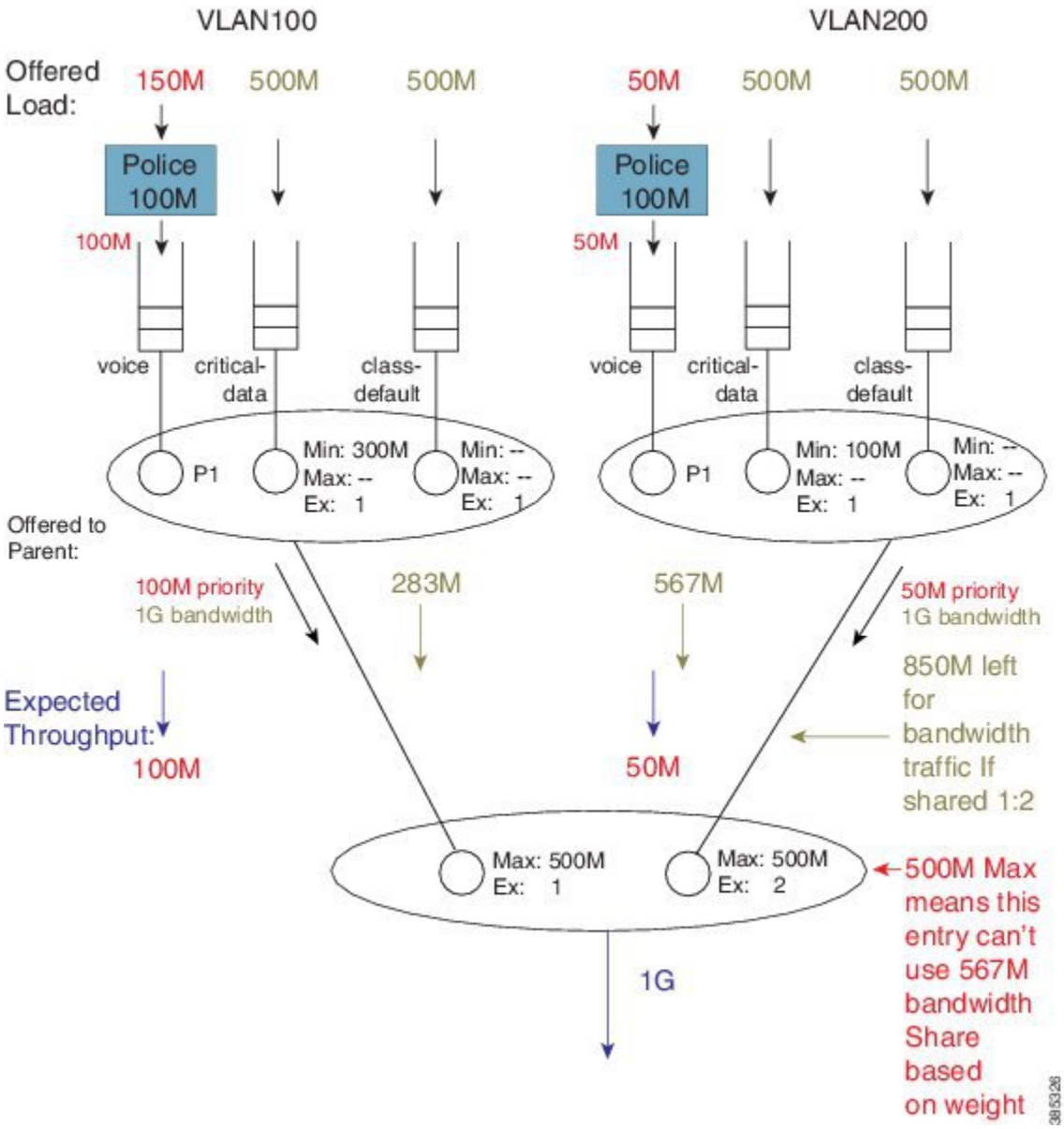


As with the previous example, we first examine the total load offered from the priority and bandwidth queues for each child to the parent.

The total priority load in this example is 150M. Each child is offering less than their Max rate (shape in parent policy) and the aggregate offered-priority load is less than the 1 Gbps total available bandwidth. (Recall the example in schedule operation where the total offered priority traffic exceeded the total available bandwidth.) This means the entire priority load offered from each child would be forwarded. With 150 Mbps scheduled from priority queues, we have 850 Mbps available for bandwidth queues.

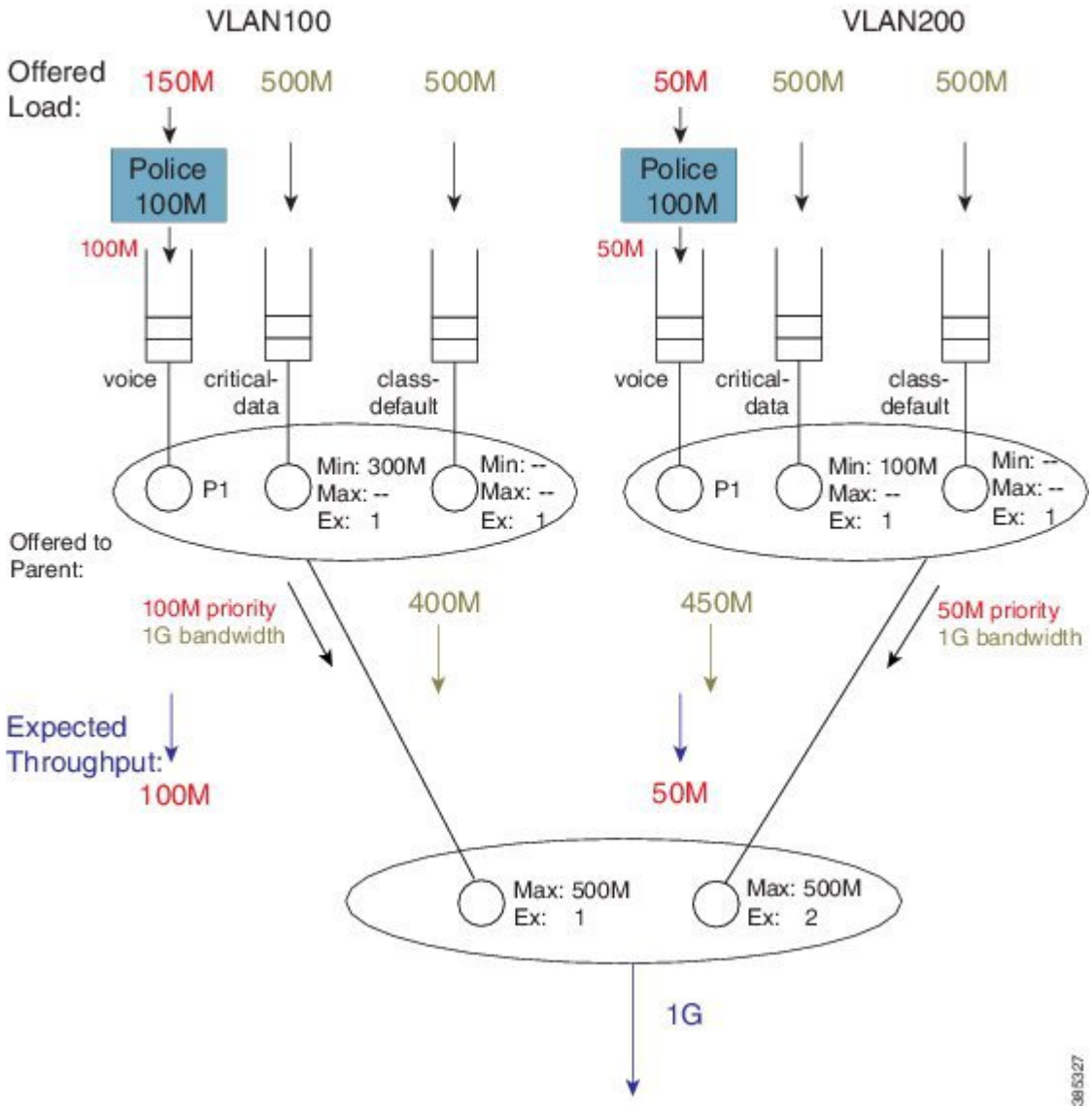
To calculate how to apportion the bandwidth between each child, let's first look at the excess weight configured in each schedule entry in the parent:

Figure 52: Apportioning Bandwidth Share Between Children



If we focus exclusively on the excess weight, VLAN200 would be apportioned 567 Mbps of the interface bandwidth (2/3 of 850 Mbps). However, we also need to factor in the Max value (500 Mbps) configured in the schedule entry, which includes the 50 Mbps of priority traffic from that child. This means that VLAN 200 will actually forward 450 Mbps of bandwidth traffic and VLAN100 will forward 400 Mbps of bandwidth traffic (850 Mbps - 450 Mbps for VLAN 200):

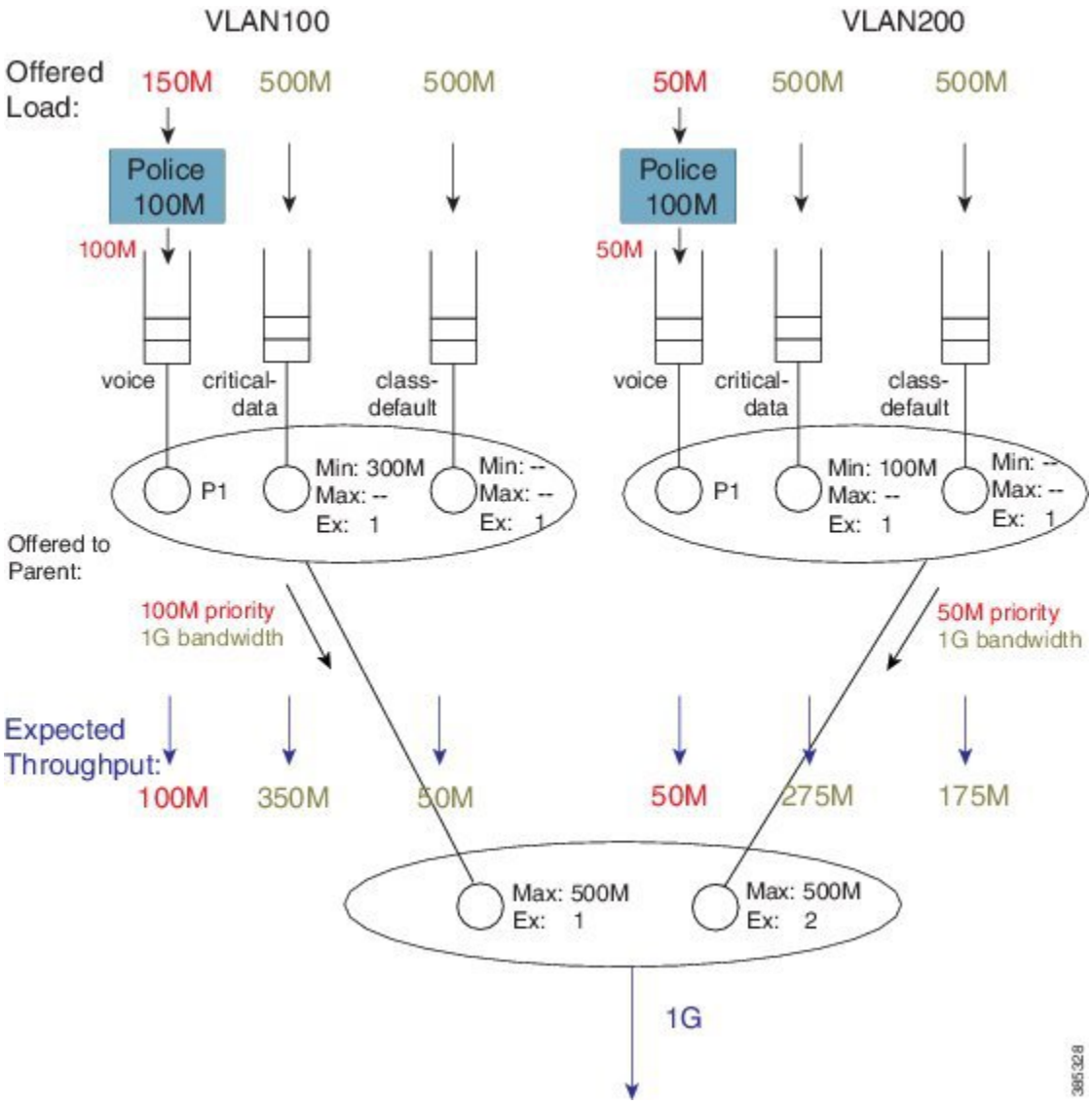
Figure 53: How the Max value of the Parent's Schedule Entry influences Bandwidth Sharing



As the sum of the Max values at the parent level is less than or equal to the available physical bandwidth, the Ex values in the parent policies do not add value – each child will receive a total throughput matching its shape rate (e.g., for VLAN100, 100M + 400M = 500M [the shape rate]). Observe that with such a configuration, any bandwidth unused by one child would not be available to another. Any child is always limited to the configured Max value.

With the total throughput for bandwidth classes in each child, we can now calculate the throughput each individual class in that child will receive. Recall from the schedule operation that Min bandwidth guarantees are always serviced first and any excess bandwidth is shared based on the Ex values, which always default to 1:

Figure 54: Factoring Total Throughput to Apportion Bandwidth within each Child Schedule



For example, the bandwidth apportioned to the class critical-data of VLAN200 would be 275M (100M (Min guarantee) + ½ (450M -100M), where we derive "½" from the Ex ratio of 1:1).

Bandwidth Command is Only Locally Significant

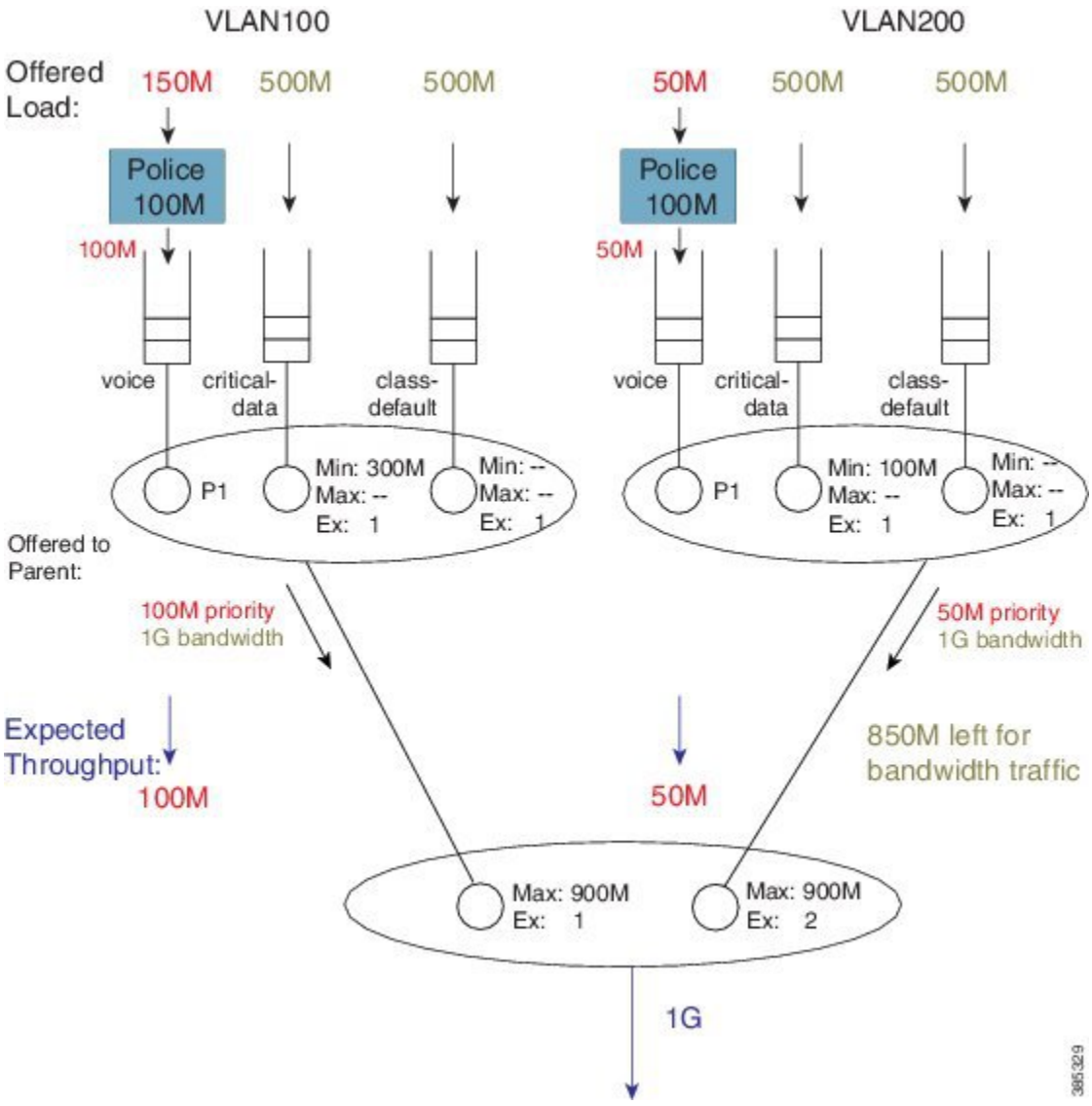
To highlight the risk of using the **bandwidth** command in hierarchical policies, we will modify the previous configuration example by increasing the parent shapers so that they are no longer the constraining factor. In the revised configuration, the sum of the parent shapers oversubscribes the physical bandwidth available. (Commands flagged with asterisks indicate how this configuration differs from that presented inBandwidth Command in Leaf Schedules, on page 422 .)

```
policy-map child100
  class voice
```

```
    priority
    police cir 100m
  class critical-data
    bandwidth 300000
!
policy-map child200
  class voice
    priority
    police cir 100m
  class critical-data
    bandwidth 100000
!
policy-map parent100
  class class-default
    shape average 900m                          ****
    service-policy child100
!
policy-map parent200
  class class-default
    shape average 900m                          ****
    bandwidth remaining ratio 2
    service-policy child200
!
int g1/0/4.100
  encaps dot1q 100
  service-policy out parent100
!
int g1/0/4.200
  encaps dot1q 200
  service-policy out parent200
```

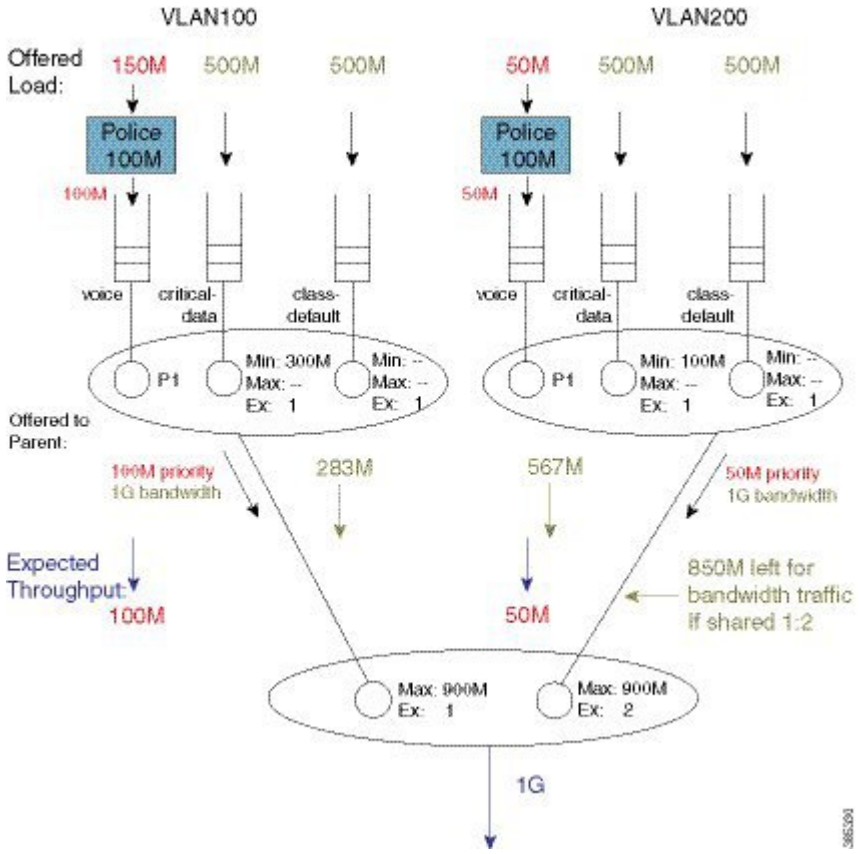
If we apply the offered load profile from Bandwidth Command in Leaf Schedules, on page 422 , the hierarchy and load profile will appear as follows:

Figure 55: Scheduling Hierarchy Example - Parent Shapers no Longer Constraining



Similar to the previous example, 850 Mbps are available (remaining) for bandwidth queues. (Inspecting the sum of priority load and bandwidth traffic share for each child, you notice that the Max value in each parent schedule would not be exceeded.) Based on the excess weights configured in the parent schedule, we calculate the bandwidth share each child would receive (from the parent schedule): 283 Mbps and 567 Mbps.

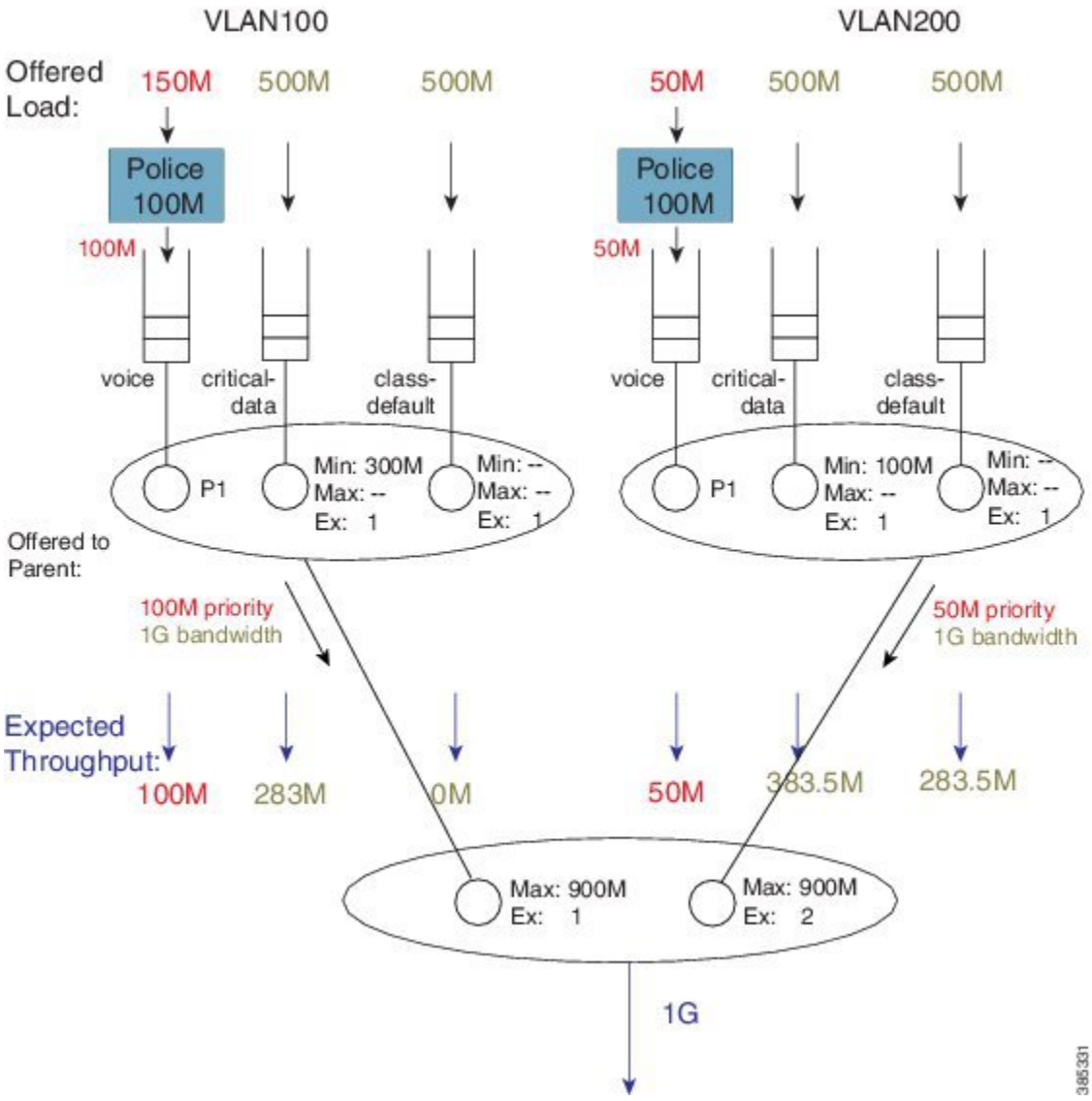
Figure 56: Calculating Bandwidth Share for Each Child based on Excess Weight Configured in Parent Schedule



Note In contrast to the previous example, because the shape values are no longer constraining, total throughput for each child does not match the shape rate.

Let's examine the entries in each child schedule to see how bandwidth would be apportioned to each class:

Figure 57: How Child Entries Dictate how Bandwidtth is Apportioned



Viewing the child schedule for VLAN100, you notice that the schedule entry for class critical-data has a Min value of 300 Mbps configured. The 283 Mbps bandwidth apportioned to this schedule is insufficient to satisfy this guarantee.

The key point of this discussion is that Min bandwidth guarantees are only locally relevant; Min bandwidth propagation does not exist. Traffic from one child schedule competes equally with excess traffic from another.

Also, please note that using Min in scheduling hierarchies could starve other classes of service (in this example, class-default in VLAN100). To avoid this, use only the **bandwidth remaining** command in child policies.

Tip

If you oversubscribe parent shapers in a hierarchical policy and want to avoid starving some classes of service, ensure that the sum of policers on your priority queues does not exceed the bandwidth available. Furthermore, consider using the **bandwidth remaining** over the **bandwidth** command.

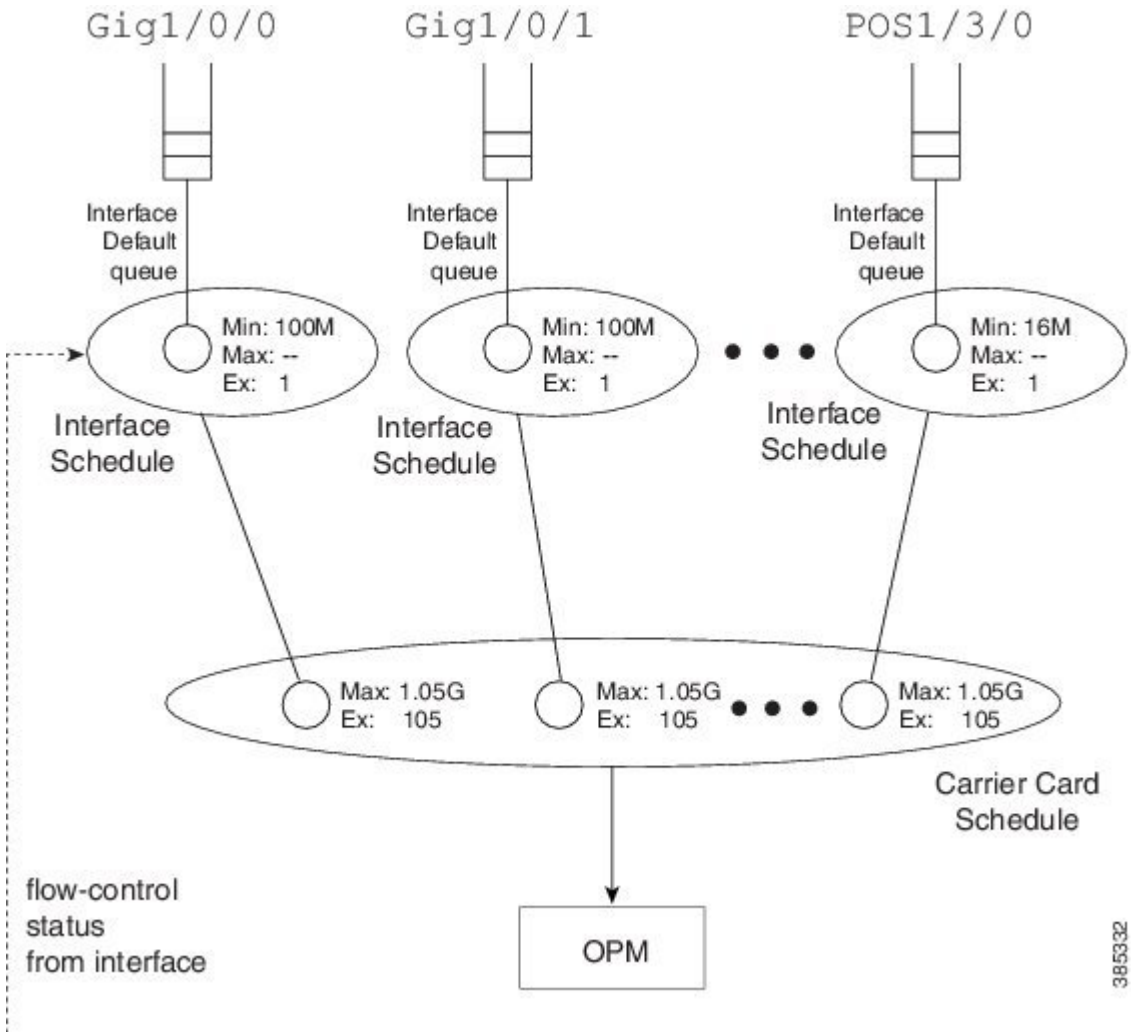
Policy-Maps Attached to Logical Interfaces

Earlier in this chapter, we delineated the two primary methods for creating scheduling hierarchies: QoS policies attached to logical interfaces and hierarchical policy-maps. In prior examples, we outlined polices attached to logical interfaces. Let's explore this scenario in more detail.

Interface Scheduling

Before looking at how a policy on a logical interface alters the hierarchy, we need to carefully examine the interface schedule and hierarchy that exist before any QoS policy is applied:

Figure 58: Interface Schedule and Hierarchy before Application of QoS Policy



The *OPM* (Output Packet Module) sits at the root of the scheduling hierarchy. Upon receiving a packet handle, it fetches the actual packet from memory and pushes it towards the physical interface.

Directly below the OPM layer (from a decision-making perspective) you will find the carrier card schedule. On modular platforms we find one such schedule per slot whereas on fixed systems we have one for the entire system.

Consider a modular chassis with one slot housing an SIP10 that has a 10 Gbps link over the backplane to the ESP (Embedded Services Processor – also termed the *forwarding processor*). The SIP10 can hold 4 SPAs (Shared Port Adapters) where each could have interface(s) totaling at most 10 Gbps capacity. If you combine SPAs in the SIP that exceed the backplane capacity, that link might be a congestion point. Should this occur, the carrier card schedule ensures fairness between interfaces; the excess weight for each interface is proportional to the interface speed.

To *condition traffic* within a platform, we set the Max value in the carrier card schedule for each interface to slightly exceed the interface's bandwidth. We want to send enough traffic towards a physical interface such that we never underrun (starve) that interface. Furthermore, we need to quit sending whenever the interface indicates that its egress buffers are filling, which could happen when an interface receives a pause frame from a downstream device, a serial interface expands its data by bit or byte stuffing, etc.

Here is the key: We push traffic towards a physical interface such that it always has data to send down the wire and we temporarily *pause sending* whenever the interface indicates that it has sufficient data buffered.

An interface directs us to stop sending traffic through a *flow-control message*. By design, a schedule (not a schedule entry) responds to this message - it stops sending. For this reason we must always have an interface schedule for every physical interface in the box. The *interface default queue* (the queue used in absence of QoS) is a child of this interface schedule.

Each interface can send distinct high and low priority flow control messages (to the interface schedule), maintaining distinct buffers and queues for priority and bandwidth traffic:

If the schedule receives a message that bandwidth traffic buffers are filling it will pause such traffic but continue to forward priority traffic.

If we receive a message that priority buffers are filling we will pause sending any packets until the congestion clears.

This scheme extends the concept of priority propagation to the physical interface (recall that this connotes whether a packet handle stems from a priority or bandwidth class) and minimizes jitter to industry leading levels for latency sensitive traffic.

Shape on Parent, or Queue on Child

Now let's look at a typical policy that might be attached to a logical interface (a construct referred to as *shape on parent* or *queue on child*):

```
policy-map child100
  class voice
    priority
    police cir 100m
  class critical-data
    bandwidth 300000
!
policy-map parent100
  class class-default
    shape average 900m
    service-policy child100
!
int g1/0/0.100
  encaps dot1q 100
```

```
service-policy out parent100
```

In this construct, you are required to configure a shaper in the parent policy(`shape average 900M`). The original intent of this construct was to apportion bandwidth to each logical interface. We consider the shape (Max) rate to be the bandwidth owned by that logical interface and allow the child policy to apportion bandwidth within that owned share.

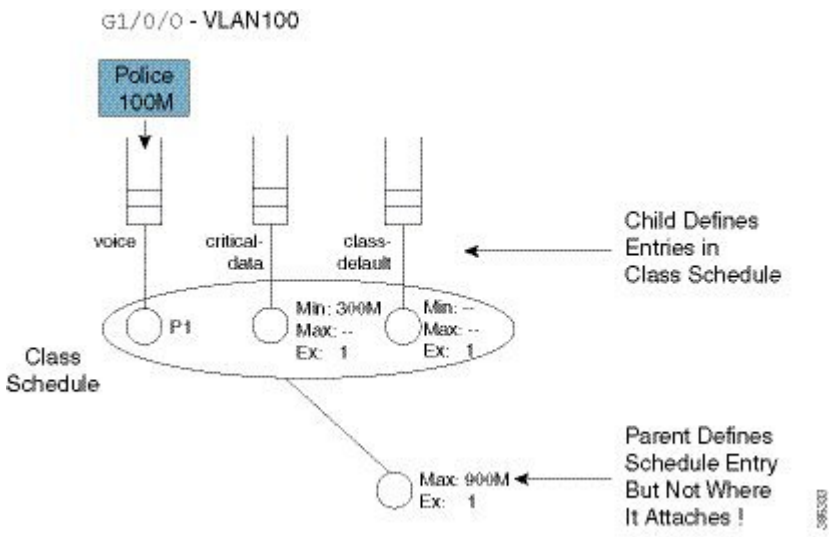
One useful application of this construct is to condition traffic for a remote site. For example, let's say that your corporate hub has a GigabitEthernet link but is sending traffic to a remote branch with a T1 connection. You want to send traffic at the rate the remote branch can receive it. To avoid potentially dropping packets in the provider device that offers service to that branch, you would configure the parent shaper at a T1 rate and queue packets on the hub. This maintains control of what is forwarded initially if that branch link were a congestion point.

Customers have asked to over-provision the shapers on logical interfaces (representing either individual subscribers or remote sites). The assumption is that all logical interfaces would not necessarily be active at all times. As we want to cap the throughput of an individual subscriber, we don’t want to waste bandwidth if an individual logical interface is not consuming its full allocated share.

So, do we oversubscribe? If yes, to provide fairness under congestion thru excess weight values, you should configure a `bandwidth remaining ratio` in the parent. Furthermore, be aware of what service any individual logical interface would receive under congestion.

Returning to the configuration, here is the resultant hierarchy:

Figure 59: Shape on Parent / Queue on Child Construct

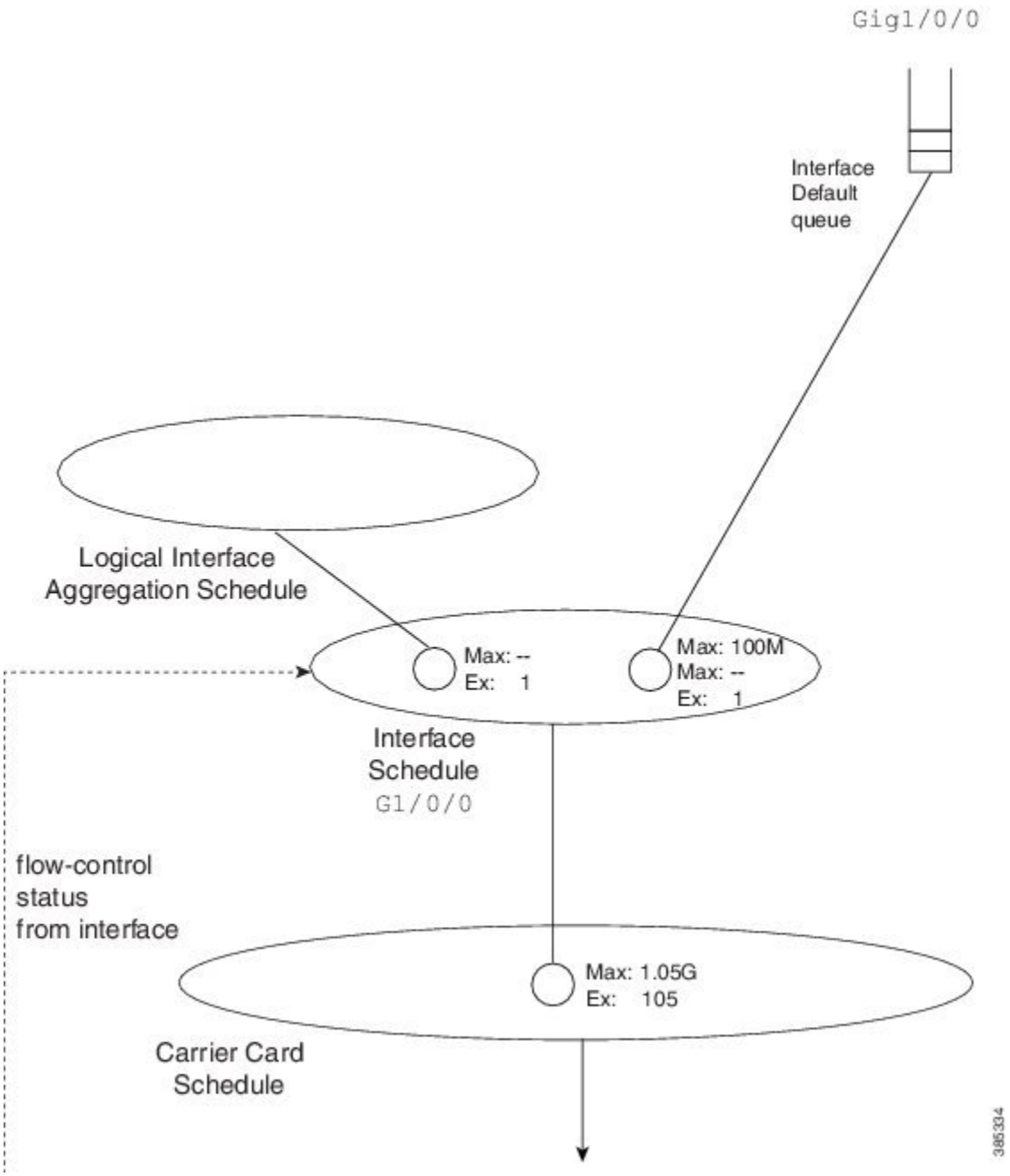


As stated, a child policy defines bandwidth sharing within the logical interface. We usually refer to the queues here (voice, etc.) as *class queues* (with treatment defined by classes within the policy-map) and the schedule at this layer as the *class layer schedule*.

In the parent policy we define a parent shaper (`Max: 900M`) and also the implicit bandwidth share of ‘1’ (`Ex: 1`). Observe that the QoS configuration does not explicitly specify where we should graft this logical interface to the existing interface hierarchy (note the un-attached schedule entry) and the router must know which physical interface a logical interface is associated with to determine where to build the hierarchy.

For a policy on a VLAN, it is evident which interface is involved - we attach the (logical interface) policy in the subinterface configuration. For other interface types (e.g., a tunnel interface), we may need to examine routing information to determine the egress physical interface for that particular logical interface.

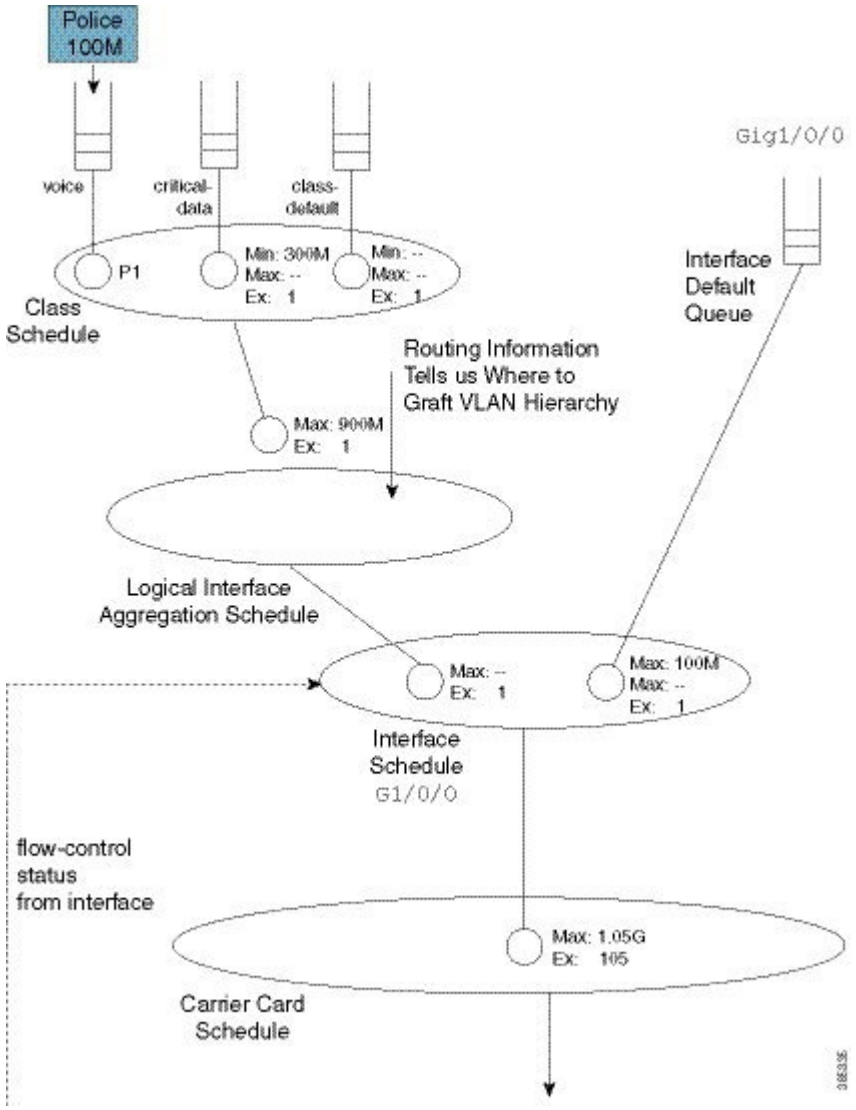
Figure 60: Existing Interface Hierarchy (The World Before the Graft)



After we know which interface is involved, we can modify the hierarchy for that interface. First we create a schedule (the logical interface aggregation) that will serve as a grafting spot for the logical interface hierarchy defined in the shape on parent (or queue on child) policy.

Initially, the interface schedule had a single child, the interface default queue. Now, we create a second child, the *logical interface aggregation schedule*. Observe how the excess weight for this schedule matches that of the interface default queue – it defaults to '1' as always.

Figure 61: Existing Interface Hierarchy (The World After the Graft)



Notice that in the shape on parent policy, we have only class-default with a child policy:

```
policy-map parent100
  class class-default
    shape average 900m
    service-policy child100
```

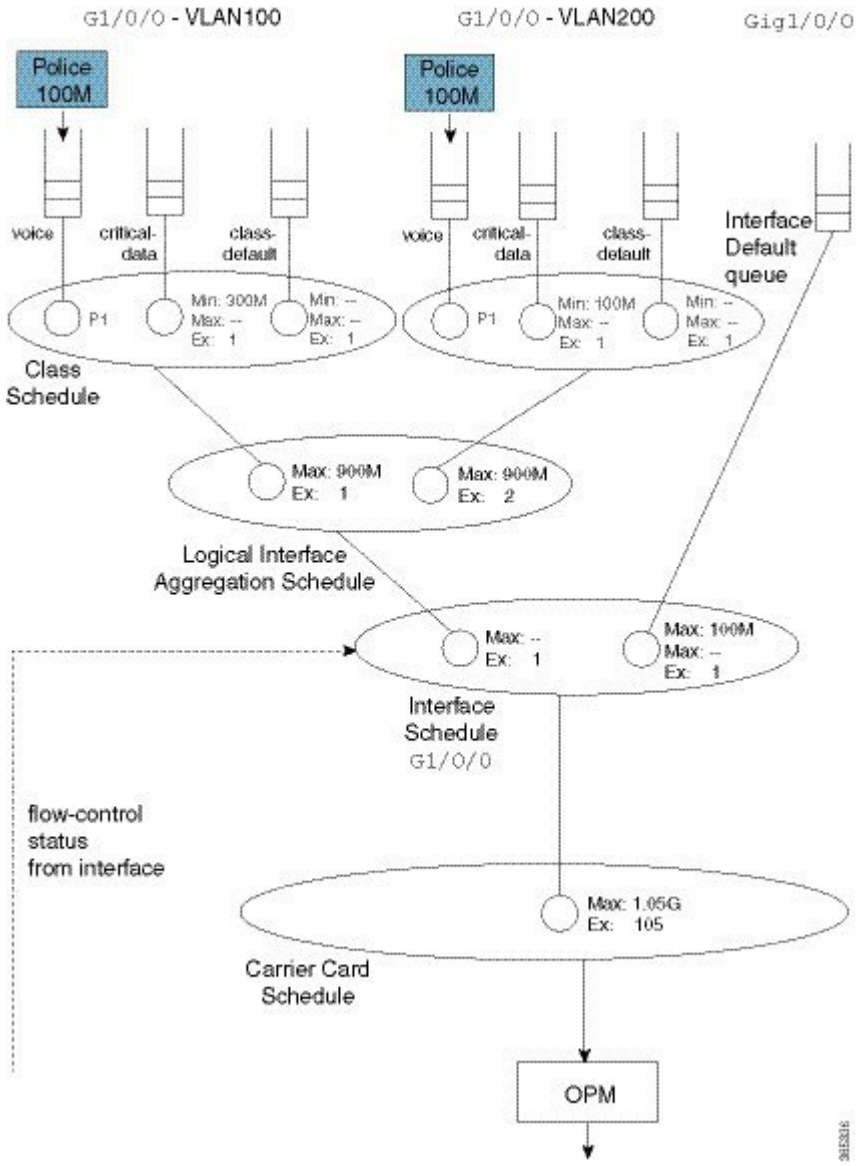
This is a special case where we just define a schedule entry rather than create a schedule for this policy. We refer to this entity as a *collapsed class-default*.

To grasp the significance of this concept, let's add a policy to another VLAN (VLAN200). (Relative to the `policy-map parent100` listed at the beginning of the topic, we have added asterisks):

```
policy-map child200
  class voice
    priority
    police cir 100m
  class critical-data
    bandwidth 100000
!
policy-map parent200
  class class-default
    shape average 900m                    ****
    bandwidth remaining ratio 2
    service-policy child200
!
int g1/0/0.200
  encaps dot1q 200
  service-policy out parent200
```

The complete scheduling hierarchy would now look as follows:

Figure 62: A Complete Hierarchical Scheduling Framework to Handle Congestion and avoid Wasting Bandwidth



Observe that in the second parent policy (the policy to VLAN200) we specified a bandwidth remaining ratio of 2, controlling fairness between VLANS. Recall from the Qos Scheduling chapter the existence of peers in the parent policy of flat policies, which enable us to use either the **bandwidth remaining ratio** or **bandwidth remaining percent** command to specify the excess weight. In the shape on parent policy construct no peers exist. When you configure a QoS policy-map, QoS cannot know what will materialize as peers in the logical interface aggregation schedule. So, neither the **bandwidth remaining ratio** nor the **bandwidth remaining percent** command is supported.

This complete scheduling hierarchy truly highlights the benefits of the Cisco Modular QoS CLI (MQC) and the Hierarchical Scheduling Framework (HQF). For any given interface, the hierarchy is deterministic; we know clearly which packet will be forwarded next. As we have schedules to handle all congestion points, no bandwidth is wasted regardless of where congestion may occur.

Advantages of Policies on Logical Interfaces

The ability to attach policy-maps to logical interfaces offers this significant advantage: management in scaled environments and ease of configuration. For each logical interface, you can reuse or create policy-maps. That is, you might attach a policy-map to each of 1000 VLANs configured on an Ethernet-type interface. To review the QoS statistics for an individual logical interface, you can issue the **show policy-map interface** *interface-name*.

Be aware that the advantages can also be perceived as dangers. If the physical bandwidth available exceeds the sum of your parent shapers, then examining a single logical interface in isolation suffices. However, if the sum of parent shapers exceeds the physical bandwidth available, you need to consider contention between logical interfaces and how much bandwidth an individual interface is truly guaranteed. Viewing an individual interface in isolation may be misleading.

Multiple Policies Definition and Restrictions

We use *Multiple Policies (MPOL)* to describe situations where a policy-map is attached to a logical interface while the policy-map is simultaneously attached to the physical interface to which that logical interface is bound (e.g., a VLAN subinterface and the physical Ethernet interface).

MPOL can also refer to instances where policy-maps are attached to different logical interface types that are bound to the same physical interface. For example, imagine a policy attached to both a VLAN subinterface and a tunnel interface, where both exit the same physical interface.

Currently, the ASR 1000 Series Aggregation Service Router supports a very limited implementation of MPOL. If you have a policy-map attached to a logical interface the only policy you can attach to the physical interface is flat with only class-default and a shaper configured, as in the example below. This topology supports scenarios where the service rate (from a provider) differs from the physical access rate. For example, consider a GigabitEthernet interface connection to your provider where you only pay for 200 Mbps of service. As the service provider will police traffic above that rate, you will want to shape everything (you send) to 200 Mbps and apportion that bandwidth locally.

Note You must attach the policy to the physical interface before you attach it to any logical interface. Furthermore, you can't attach policy-maps to more than one logical interface type bound to a single physical interface.

Returning to the previous example of policies attached to two VLAN subinterfaces, let's now add a 200 Mbps shaper to the physical interface. The complete configuration would look as follows. The asterisks indicate how this and the previous configuration differ.

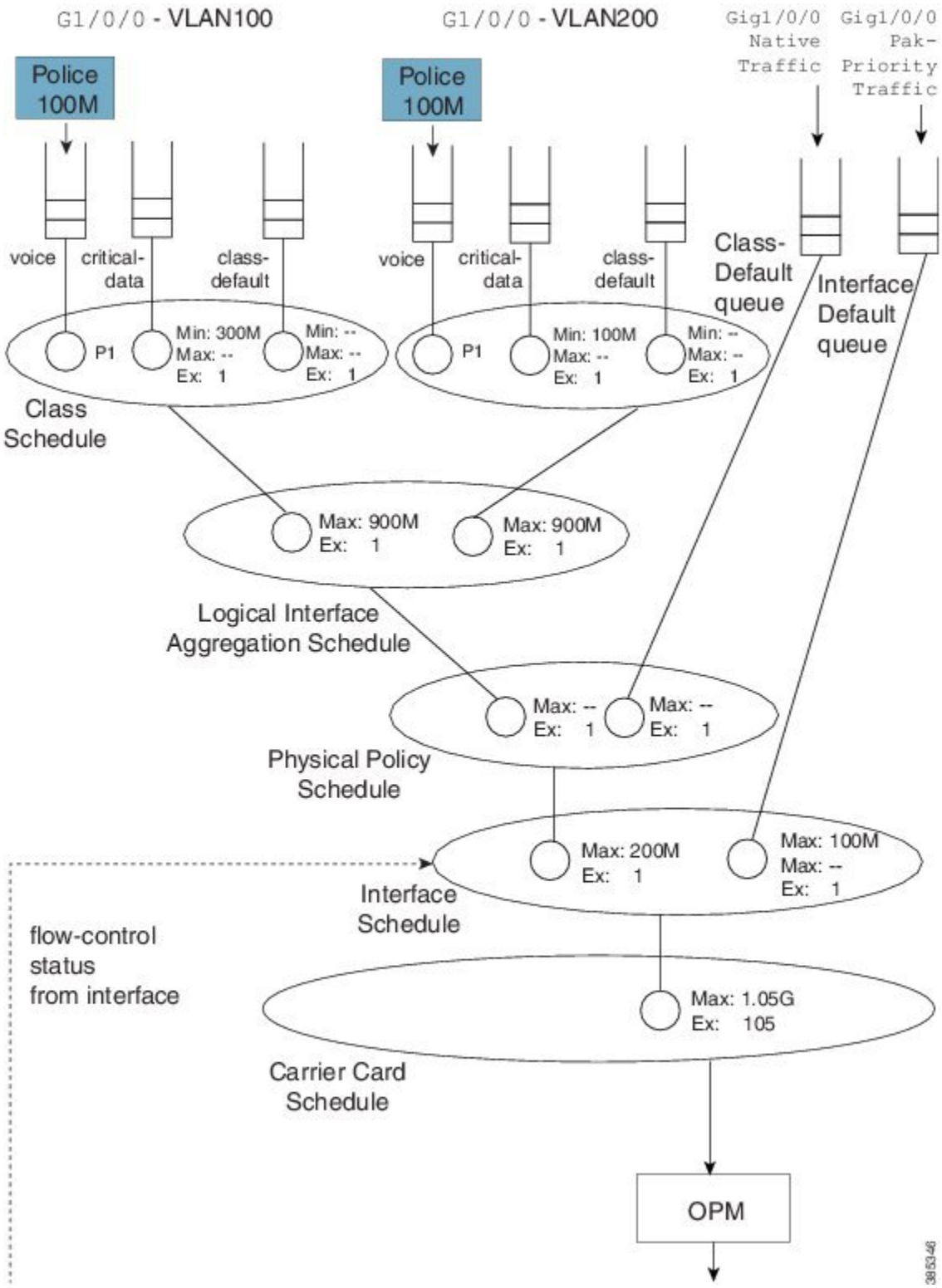
```
policy-map physical-shaper                          ****
  class class-default                               ****
    shape average 200m                              ****
 !
policy-map child100
  class voice
    priority
    police cir 100m
  class critical-data
    bandwidth 300000
!
policy-map child200
  class voice
    priority
```

```
    police cir 100m
  class critical-data
    bandwidth 100000
!
policy-map parent100
  class class-default
    shape average 900m
    service-policy child100
!
policy-map parent200
  class class-default
    shape average 900m
    bandwidth remaining ratio 2
    service-policy child200
!
! Note - must attach physical policy before logical policies
!
int g1/0/0                                          ****
  service-policy output physical-shaper             ****
!
int g1/0/0.100
  encaps dot1q 100
  service-policy out parent100
!
int g1/0/0.200
  encaps dot1q 200
  service-policy out parent200
```

Notice that we have introduced another schedule as well as a queue that will be used for any user traffic sent through the physical interface. The logical interface aggregation schedule has now been created as a child of the physical policy schedule rather than directly as a child of the interface schedule. The combination of traffic through the logical interfaces and user traffic through the physical interface is now shaped to 200 Mbps.

The complete scheduling hierarchy would appear as follows:

Figure 63: Creating a Logical Interface Aggregation as a Child of the Physical Policy Schedule



Hierarchical Policy-Maps

In the previous sections, we showed how hierarchies are constructed when policy-maps are attached to logical interfaces. A second approach is to use *hierarchical policy-maps* and explicitly construct the hierarchy you desire. Using this approach you gain some flexibility but loose some scale. (Recall that with policies on logical interfaces you gained management in scaled environments.) The ASR 1000 Series Aggregation Service Router supports up to 1,000 classes in a policy-map, which means that the largest number of logical interfaces you could represent is 1,000.

To belong to a class within a hierarchical policy-map, a packet must match the child and (any) parent classification rules. In an earlier VLAN example we showed how to use VLAN ID-based classification in a parent class and DSCP-based classification in a child class.

The following configuration shows how we might achieve similar behavior to that with a MPOL-physical shaper (see Multiple Policies Definition and Restrictions, on page 439). Here we use a three-level hierarchical policy-map (the maximum number of layers we support).

The parent policy has only class-default, which means that all traffic through the interface belongs to this class:

```
policy-map physicalshaper
  class class-default
    shape average 200m
    service-policy vlansharing
```

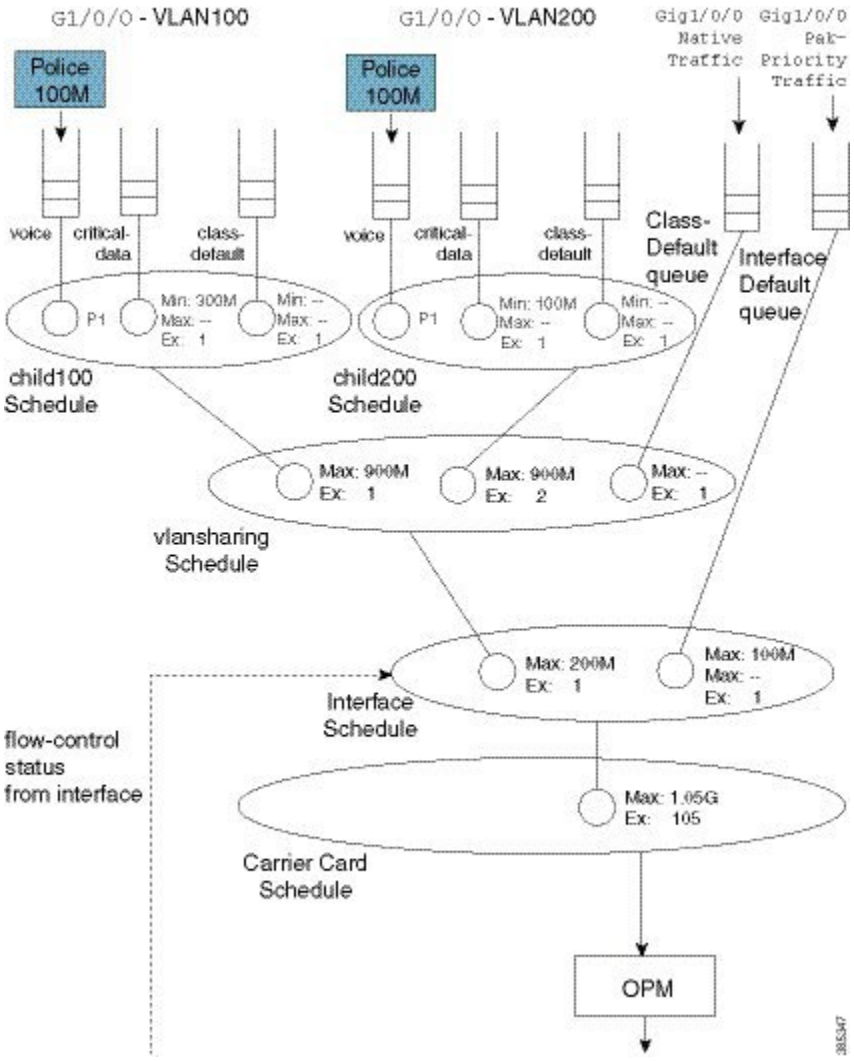
The child level has VLAN-based classification. Traffic belonging to VLAN 100 or VLAN200 will fall into one of the user-defined classes. (Additionally, we have an implicit class-default in this policy that will capture traffic from other VLANs or with no VLAN tag.) Each VLAN class has a policy to further classify traffic based on DSCP:

```
class-map vlan100
  match vlan 100
class-map vlan200
  match vlan 200
class-map voice
  match dscp ef
class-map critical-data
  match dscp af21
        !
policy-map child100
 class voice
   priority
   police cir 100m
 class critical-data
   bandwidth 300000
!
policy-map child200
 class voice
   priority
   police cir 100m
 class critical-data
   bandwidth 100000
!
policy-map vlansharing
 class vlan100
   shape average 900m
   bandwidth remaining ratio 1
   service-policy child100
```

```
 class vlan200
   shape average 900m
   bandwidth remaining ratio 2
   service-policy child200
!
policy-map physicalshaper
   class class-default
     shape average 200m
    service-policy vlansharing
 !
int g1/0/0
  service-policy output physicalshaper
```

A hierarchy constructed based on the above configuration will look as follows:

Figure 64: Hierarchical Policy-Maps to Explicitly-Construct a Hierarchy



If you compare this hierarchy to the previous MPOL example (Figure 25), you will notice some slight differences.

Firstly, native interface traffic (traffic in neither VLAN 100 nor 200) now shares a vlansharing schedule with the schedule entries for each VLAN. In the MPOL example, the native traffic received an equal share to that of all (both) VLANs (1/2 the available bandwidth). In this hierarchy, in contrast, it is guaranteed only 1/(1 + 2 + 1) of available bandwidth as it competes with the VLANs in the same schedule.

Secondly, with a single policy-map on the physical interface you no longer have the ability to look at statistics for a single VLAN only. Compare this code from the MPOL example:

```
int g1/0/0
  service-policy output physical-shaper
!
int g1/0/0.100
  encaps dot1q 100
  service-policy out parent100
!
int g1/0/0.200
  encaps dot1q 200
  service-policy out parent200
```

with this:

```
int g1/0/0
  service-policy output physicalshaper
```

The output of the **show policy-map interface GigabitEthernet1/0/0** command would reflect all levels of the hierarchical policy-map.

Hierarchical policy-maps can add flexibility that is unachievable with policy-maps on logical interfaces. The following examples illustrate this.

Example 1. Add Queues for Different Classes of Traffic

In the discussion of the MPOL example (and captured in the code snippet below), we noted that the physical interface policy could contain only class-default and a shaper in that class:

```
policy-map physical-shaper
  class class-default
    shape average 200m
```

That is, you cannot provide different treatment to unique classes of traffic that were forwarded over the native interface (traffic with no VLAN tag).

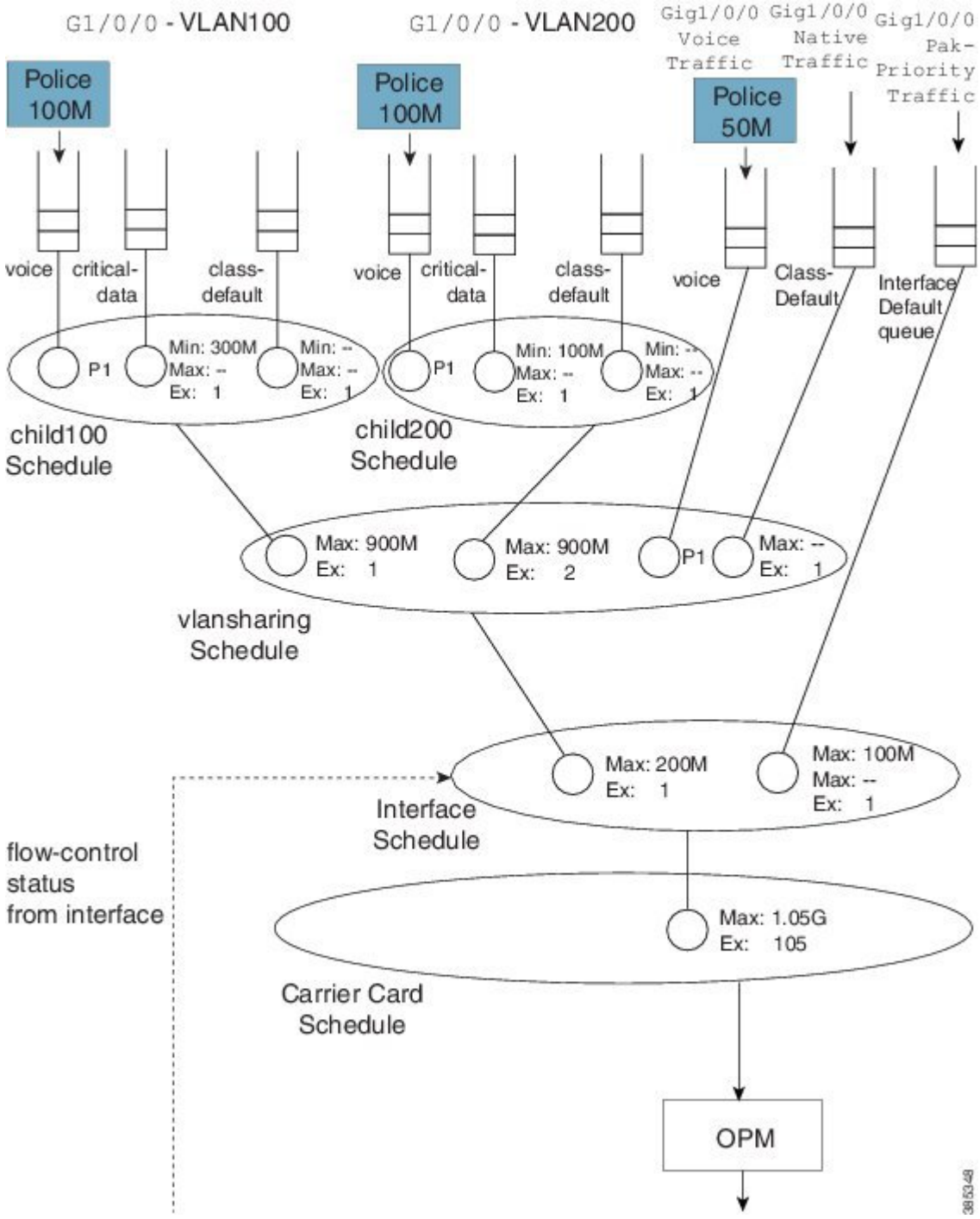
In contrast, with an hierarchical construct, we can add queues for different classes of traffic to forward (over the physical interface). For example, if we wanted to add a priority class for voice traffic over the physical interface, we could modify the vlansharing policy-map as follows (see the asterisks):

```
class-map vlan100
  match vlan 100
class-map vlan200
  match vlan 200
class-map voice
  match dscp ef
class-map critical-data
  match dscp af21
        !
policy-map child100
 class voice
   priority
   police cir 100m
```

```
 class critical-data
   bandwidth 300000
!
policy-map child200
 class voice
   priority
   police cir 100m
 class critical-data
   bandwidth 100000
!
policy-map vlansharing
  class vlan100
    shape average 900m
    bandwidth remaining ratio 1
    service-policy child100
  class vlan200
    shape average 900m
    bandwidth remaining ratio 2
    service-policy child200
  class voice                                ****
    priority                                 ****
    police cir 50m                           ****
!
policy-map physicalshaper
  class class-default
    shape average 200m
    service-policy vlansharing
 !

int g1/0/0
  service-policy output physicalshaper
```

The hierarchy for this configuration would look as follows:

Figure 65: Represent Queues for Different Traffic Classes with a Hierarchical Construct



Notice the new capture that captures any traffic marked with the DSCP codepoint of EF but not tagged with VLAN ID of 100 or 200.

Observe in this hierarchy that P1 traffic from a local queue (Gig1/0/0 Voice Traffic) competes with priority propagation traffic in the VLAN sharing schedule. In such a scenario a local entry configured with priority

is serviced before priority propagation traffic. That is, voice packets from a physical interface (Gig1/0/0) have a slightly higher priority than voice packets from VLAN 100 or 200. To avoid starvation of other classes, we use admission control on the priority queues.

Example 2. Attaching a Policy to Different Logical Interface Types

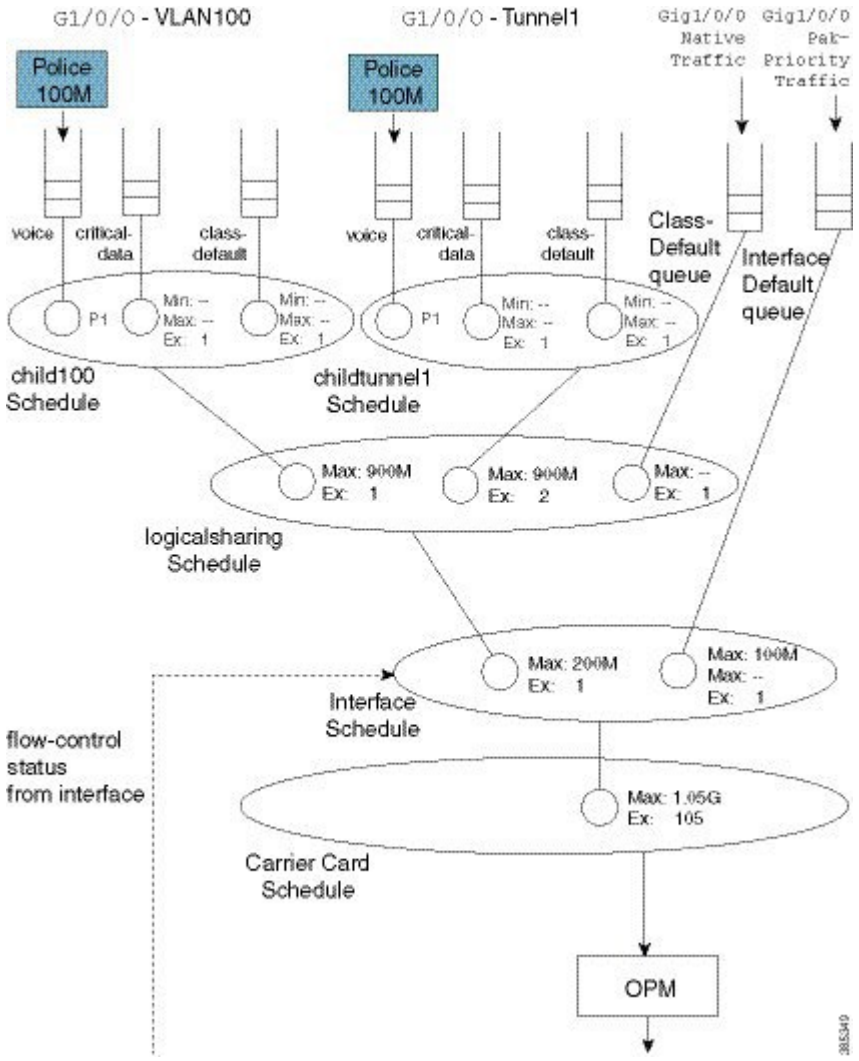
In the section Policy-Maps Attached to Logical Interfaces, on page 432 we indicated that you cannot attach a policy to different logical interface types on the same physical interface. This limitation does not apply to hierarchical class-maps.

Let's say that we want one child schedule for VLAN100 and one child for QoS on a tunnel where both exit the same physical interface. Within the same policy-map, we could classify tunnel traffic using an access list and VLAN traffic using the VLAN ID (see the asterisks):

```
ip access-list extended tunnel1traffic
  permit ip host 192.168.1.1 host 10.0.0.1
!
class-map vlan100
  match vlan 100
class-map tunnel1traffic
  match access-group name tunnel1traffic
!
class-map voice
  match dscp ef
class-map critical-data
  match dscp af21
        !
policy-map child
  class voice
    priority
    police cir 100m
  class critical-data
    bandwidth remaining ratio 1
!
policy-map logicalsharing                  ****
  class vlan100
    shape average 900m
    bandwidth remaining ratio 1
    service-policy child
  class tunnel1traffic
    shape average 900m
    bandwidth remaining ratio 2
    service-policy child
!
policy-map physicalshaper
  class class-default
    shape average 200m
    service-policy vlansharing
 !

int g1/0/0
  service-policy output physicalshaper
```

The hierarchy for this configuration would look as follows:

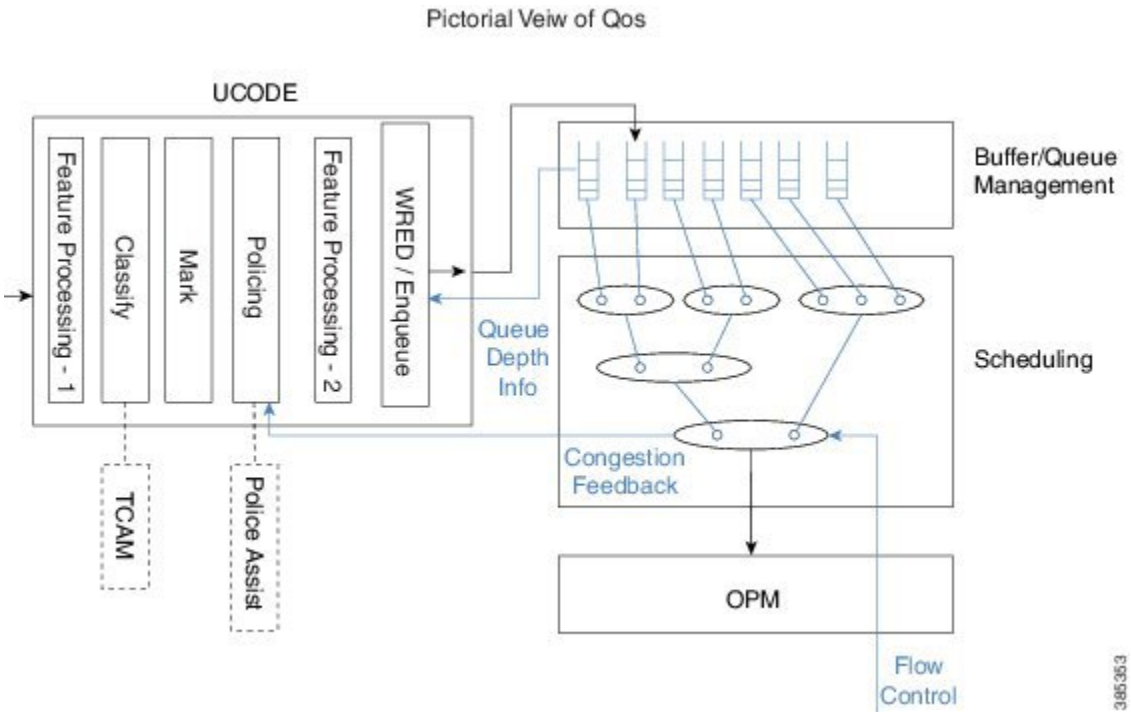
Figure 66: Attaching a Policy to Different Logical Interface Types



A Note on Overhead Accounting

In the policing chapter we introduced the concept of *policing length* (how we perceive a packet's length when a policer evaluates conformance to a configured police rate; see What's Included in the Policer-Rate Calculation (Overhead Accounting), on page 595). Similarly, in the scheduling chapter we introduced the concept of *scheduling length* (how we consider a packet's length when evaluating conformance to a configured scheduler rate; seeWhat's Included in Scheduling Rate Calculations (Overhead Accounting), on page 364 .) By convention, in both cases we include the Layer 2 header and datagram lengths and exclude CRC or interpacket overhead.

With an hierarchical scheduling construct, you might encounter instances where the policing and scheduling lengths differ. To understand this let's examine the execution order of features.



On the ASR 1000 Series Aggregation Services Router, queuing and scheduling is performed in hardware. After we enqueue a packet, hardware assumes control and no further processing is performed – the packet must have all headers and be prepared to traverse the wire. As expected, non-queuing features are performed in microcode on one of the processing elements (with hardware assists, in some instances).

Consider two scenarios.

Configuring a QoS-queuing policy on a GRE tunnel

When we classify an incoming IP packet (ultimately encapsulated in an outer IP/GRE header), we examine just the original IP packet. Consequently, classification statistics will exclude the outer IP/GRE headers as they are missing at the time. As the pictorial view indicates, we perform marking and evaluate policers at this time. Similar to the classification length, the policing length will include neither the outer IP/GRE headers nor any egress Layer 2 header, as we don't yet know which physical interface or encapsulation type the packet will egress. After QoS non-queuing features we continue processing the packet by adding the outer IP/GRE header and appropriate Layer 2 header for the final egress interface. When all processing concludes, we pass the packet to the WRED/Enqueue block. This action places the packet on the appropriate egress queue in hardware with all headers added; the scheduling length now includes the outer IP/GRE and Layer 2 headers.

Configuring the QoS policy on the physical egress interface

The results differ. When we examine features on the tunnel no QoS is configured and so we proceed to feature processing. Before reaching the QoS policy, we complete all tunnel processing and add egress headers. So, the classification statistics and policing length will now include the outer headers; policing and scheduling lengths will match.

Verification

In all QoS configuration work, the primary tool to verify hierarchical scheduling configurations is the **show policy-map interface** *interface-name* command. The output of this command is organized hierarchally, reflecting how we stratify the configuration.

For example, with a hierarchical policy attached to a physical interface you could use the **show policy-map interface** *interface-name***| include Class** | to display that hierarchy:

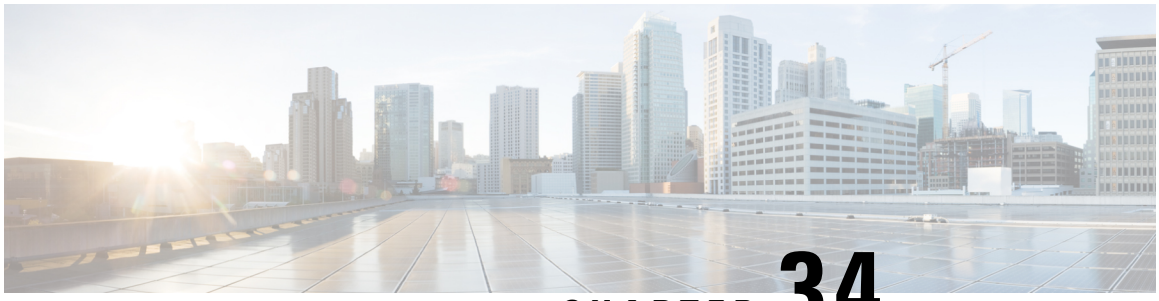
```
show policy-map int g1/0/0 | inc Class

    Class-map: class-default (match-any)
        Class-map: vlan100 (match-all)
            Class-map: voice (match-all)
            Class-map: critical-data (match-all)
            Class-map: class-default (match-any)
        Class-map: vlan200 (match-all)
            Class-map: voice (match-all)
            Class-map: critical-data (match-all)
            Class-map: class-default (match-any)
        Class-map: vlan300 (match-all)
            Class-map: voice (match-all)
            Class-map: class-default (match-any)
        Class-map: voice (match-all)
        Class-map: class-default (match-any)
```

In this example we have attached a 3-level hierarchical policy to interface GigabitEthernet1/0/0. Indentation in the class-map conveys that hierarchy. Within any class that includes a child policy, the `Service-policy: <policy-map name>` indicates that the next-indented section pertains to the child policy:

```
  Class-map: vlan100 (match-all)
          0 packets, 0 bytes
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match: vlan  100
          Queueing                                                  ****
          queue limit 3748 packets                                  ****
          (queue depth/total drops/no-buffer drops) 0/0/0           ****
          (pkts output/bytes output) 0/0
          shape (average) cir 900000000, bc 3600000, be 3600000
          target shape rate 900000000
          bandwidth remaining ratio 1

          Service-policy : child100

            queue stats for all priority classes:
              Queueing
              queue limit 512 packets
              (queue depth/total drops/no-buffer drops) 0/0/0
              (pkts output/bytes output) 0/0

            Class-map: voice (match-all)
              0 packets, 0 bytes
              5 minute offered rate 0000 bps, drop rate 0000 bps
              Match:  dscp ef (46)
              Priority: Strict, b/w exceed drops: 0

              police:
                  cir 100000000 bps, bc 3125000 bytes
                conformed 0 packets, 0 bytes; actions:
                  transmit
```

```
    exceeded 0 packets, 0 bytes; actions:
      drop
    conformed 0000 bps, exceeded 0000 bps

Class-map: critical-data (match-all)
  0 packets, 0 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match:  dscp af11 (10)
  Match:  dscp af21 (18)
  Queueing
  queue limit 1249 packets
  (queue depth/total drops/no-buffer drops) 0/0/0
  (pkts output/bytes output) 0/0
  bandwidth 300000 kbps

Class-map: class-default (match-any)
  0 packets, 0 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match: any

  queue limit 3748 packets
  (queue depth/total drops/no-buffer drops) 0/0/0
  (pkts output/bytes output) 0/0
```

Regarding **show** command output for a policy containing hierarchical scheduling, observe that any queue-related information in the parent class is meaningless (highlighted by asterisks in the example above). The output format for the **show policy-map interface** command was created at a time when IOS truly implemented a hierarchy of queues in software. The ASR 1000 Series Aggregation Services Router hardware implements a hierarchy of schedules and queues, which only exist at the leaf. Although the IOS control plane still calculates and displays a queue-limit, it never uses it. So tuning this value is fruitless.



CHAPTER 34

Legacy QoS Command Deprecation

The functionality provided by these hidden commands has been replaced by similar functionality provided via the modular QoS CLI (MQC). The MQC is a set of a platform-independent commands for configuring QoS on Cisco platforms. This means that you must now provision QoS by defining traffic classes, creating traffic policies containing those classes, and attaching those policies to the desired interfaces. This document lists the hidden commands and their replacement MQC commands.

Information About Legacy QoS Command Deprecation

QoS Features Applied Using the MQC

The MQC structure lets you define a traffic class (also called a class map), create a traffic policy (also called a policy-map), and attach the traffic policy to an interface. This comprises the following three high-level steps.

1. Define a traffic class by using the **class-map** command. A traffic class is used to classify traffic.
2. Create a traffic policy by using the **policy-map** command. A traffic policy contains a traffic class and one or more QoS features that will be applied to the traffic class. The QoS features in the traffic policy determine how to treat the classified traffic.
3. Attach the traffic policy to the interface by using the **service-policy** command.

Steps 1 and 3 do not involve legacy QoS hidden commands, which means that they are not within the scope of this document. For more information about these two steps, see the " Applying QoS Features Using the MQC " module in the *Quality of Service Solutions Configuration Guide* .

Legacy Commands Being Hidden

The table below lists the commands that have been hidden or removed. The table also lists their replacement commands (or sequence of commands).

Table 37: Map of Hidden, Removed or Unsupported Commands to Their Replacement Commands

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence
Configuring Weighted Random Early Detection or Distributed Weighted Random Early Detection Parameter Groups	

Commands
• random-detect-group
• random-detect (per VC)
Note This command is not supported in Cisco IOS Release 15.0(1)S.
Command Usage
`Router(config)# random-detect-group group-name [dscp-based|prec-based]`
`Router(config)# interface atm type number`
`Router(config-if)# pvc [name] vpi/vci`
`Router(config-if-atm-vc)# random-detect [attach group-name ]`
	Command Usage
	None (this functionality no longer exists).

Configuring Weighted Random Early Detection	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• random-detect
• random-detect dscp
• random-detect (dscp-based keyword)
• random-detect flow
• random-detect exponential-weighting-constant
• random-detect (prec-based keyword)
• random-detect precedence
Command Usage
`Router(config)# interface type number`
`Router(config-if)# random-detect [number]`
`Router(config-if)# random-detect exponential-weighting-constant exponent`
`Router(config-if)# random-detect flow`
`Router(config-if)# random-detect precedence {precedence|rsvp} min-threshold max-threshold max-probability-denominator`
`Router(config-if)# random-detect prec-based`
`Router(config-if)# random-detect dscp-based`
`Router(config-if)# random-detect dscp dscp-value min-threshold max-threshold[max-probability-denominator]`
	Command Usage
	`Router(config)# policy-map`
	`policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# random-detect dscp dscp-value min-threshold max-threshold[ mark-probability-denominator]`
	`Router(config-pmap-c)# random-detect clp clp-value min-threshold max-threshold[mark-probability-denominator]`
	`Router(config-pmap-c)# random-detect cos cos-value min-threshold max-threshold[mark-probability-denominator]`
	`Router(config-pmap-c)# random-detect discard-class discard-class-value min-threshold max-threshold[ mark-probability-denominator]`
	`Router(config-pmap-c)# random-detectprecedence ip-precedence min-threshold max-threshold[mark-probability-denominator]`
	`Router(config-pmap-c)# random-detect precedence-based`
	`Router(config-pmap-c)# random-detect ecn`
	`Router(config-pmap-c)# random-detect exponential-weighting-constant exponent`
	`Router(config-pmap-c)# random-detect cos-based`
	`Router(config-pmap-c)# random-detect dscp-based`

Commands
• random-detect flow
• random-detect flow average-depth-factor
• random-detect flow count
Command Usage
`Router(config)# interface type number`
`Router(config-if)# random-detect [number]`
`Router(config-if)# random-detect flow`
`Router(config-if)# random-detect flow count number`
`Router(config-if)# random-detect flow average-depth-factor scaling-factor`
	Command Usage
	None (this functionality no longer exists).

Configuring Bandwidth Allocation	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• max-reserved-bandwidth
Command Usage
`Router(config)# interface type number`
`Router(config-if)# max-reserved-bandwidth percentage`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# bandwidth{bandwidth-in-kbps |remaining percent percentage | percent percentage}`

Configuring Custom Queueing	

Commands
• custom-queue-list
Note This command is not supported in Cisco IOS Release 15.0(1)S.
Command Usage
`Router(config)# interface type number`
`Router(config-if)#`
`custom-queue-list[list-number]`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# bandwidth{ bandwidth-in-kbps |remaining percent percentage |percent percentage}`

Configuring Priority Queueing	

Commands
• ip rtp priority
• ip rtp reserve
Command Usage
`Router(config)# interface type number`
`Router(config-if)# ip rtp priority starting-port-number port-range bandwidth`
`Router(config)# interface type number`
`Router(config-if)# ip rtp reserve lowest-udp-port range-of-ports [maximum-bandwidth] 1000`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-name`
	`Router(config-pmap-c)# priority`

Configuring Weighted Fair Queueing	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• fair-queue (WFQ)
Command Usage (Cisco IOS Release 15.0(1)S)
`Router(config)# interface type number`
`Router(config-if)# fair-queue`
Command Usage (Cisco IOS Release 15.1(3)T)
`Router(config)# interfacetype number`
`Router(config-if)# fair-queue [congestive-discard-threshold [dynamic-queue-count [reserved-queue-count]]]`
	Command Usage (Cisco IOS Release 15.0(1)S)
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# fair-queue`
	Command Usage (Cisco IOS Release 15.1(3)T)
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# fair-queue[dynamic-queues ]`

Assigning a Priority Group to an Interface	

Commands
• priority-group
Note This command is not supported in Cisco IOS Release 15.0(1)S.
Command Usage
`Router(config)# interface type number`
`Router(config-if)# priority-group list-number`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# priority`
	`Router(config-pmap-c)# priority bandwidth-in-kbps [burst-in-bytes]`
	`Router(config-pmap-c)# priority percent percent [burst-in-bytes]`
	`Router(config-pmap-c)# priority level level`
	`Router(config-pmap-c)# priority level level [bandwidth-in-kbps [burst-in-bytes]]`
	`Router(config-pmap-c)# priority level level[percent percent [burst-in-bytes]]`

Configuring the Threshold for Discarding DE Packets from a Switched PVC Traffic Shaping Queue	

Commands
• frame-relay congestion threshold de
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay congestion threshold de percentage`
	Command Usage
	`Router(config)# policy-map policy-map-name1`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# random-detect discard-class-based`
	`Router(config-pmap-c)# random-detect discard-class discard-class min-threshold max-threshold`
	`Router(config-pmap-c)# exit`
	`Router(config-pmap)# exit`
	`Router(config)# policy-map shape`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`
	`Router(config-pmap-c)# service-policy policy-map-name1`
	`Router(config-pmap-c)# exit`
	`Router(config-pmap)# exit`
	`Router(config)# policy-map policy-map-name2`
	`Router(config-pmap)# class class-name`
	`Router(config-pmap-c)# set discard-classdiscard-class`

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence
Configuring Frame Relay Custom Queueing for Virtual Circuits	

Commands
• frame-relay custom-queue-list
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay custom-queue-list list-number`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# bandwidth{bandwidth-in-kbps | remaining percent percentage | percentpercentage}`

Configuring Frame Relay ECN Bits Threshold	

Commands
• frame-relay congestion threshold ecn
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay congestion threshold ecn percentage`
	Command Usage
	None (this functionality no longer exists).
	The closest equivalent is MQC traffic shaping (not based on ECN).
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`

Configuring Frame Relay Weighted Fair Queueing	

Commands
• frame-relay fair-queue
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay fair-queue [discard-threshold [dynamic-queue-count[reserved-queue-count [buffer-limit]]]]`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# fair-queue`
	`Router(config-pmap-c)# fair-queue queue-limit packets`
	Note The **queue-limit** *packets* keyword and argument pair is not supported in Cisco IOS Release 15.1(3)T.

Configuring Frame Relay Priority Queueing on a PVC	

Commands
• frame-relay ip rtp priority
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay ip rtp priority starting-port-number port-range bandwidth`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-name`
	`Router(config-pmap-c)# priority bandwidth-in-kbps [burst-in-bytes]`

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence
Assigning a Priority Queue to Virtual Circuits Associated with a Map Class	

Commands
• frame-relay priority-group
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay priority-group group-number`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# priority`
	`Router(config-pmap-c)# priority bandwidth-in-kbps [burst-in-bytes]`
	`Router(config-pmap-c)# priority percent percentage [burst-in-bytes]`
	`Router(config-pmap-c)# priority level level [percent percentage [burst-in-bytes]]`
	Note The **priority level**command is not supported in Cisco IOS Release 15.1(3)T.

Configuring the Frame Relay Rate Adjustment to BECN	

Commands
• frame-relay adaptive-shaping (becn keyword)
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay adaptive-shaping becn`
	Command Usage
	None (this functionality no longer exists). The closest equivalent is MQC traffic shaping (not based on BECN).
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape adaptive rate`

Configuring the Frame Relay Rate Adjustment to ForeSight Messages	

Commands
• frame-relay adaptive-shaping (foresight keyword)
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config)# frame-relay adaptive-shaping foresight`
	Command Usage
	None (this functionality no longer exists).

Enabling Frame Relay Traffic-Shaping FECNs as BECNs	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• frame-relay fecn-adapt
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)#frame-relay fecn-adapt`
	Command Usage
	None (this functionality no longer exists). The closest equivalent is MQC traffic shaping (not based on FECN/BECN).
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`

Configuring the Frame Relay Enhanced Local Management Interface	

Commands
• frame-relay qos-autosense
Note This command has not been hidden in Cisco IOS Release 15.0(1)S.
Command Usage
`Router(config)# interface type number`
`Router(config-if)#no ip address`
`Router(config-if)# encapsulation frame-relay`
`Router(config-if)# frame-relay lmi-typeansi`
`Router(config-if)# frame-relay traffic-shaping`
`Router(config-if)# frame-relay qos-autosense`
	Command Usage
	None (this functionality no longer exists).

Configuring Frame Relay Minimum Committed Information Rate (MINCIR)	
Commands • frame-relay mincir **Command Usage** `Router(config)# frame-relay mincir {in | out} bps`	**Command Usage** None (this functionality no longer exists).
Configuring Frame Relay Priority to a permanent virtual circuit (PVC)	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• frame-relay interface-queue
Command Usage
`Router(config)# interface type number`
`Router(config-if)#no ip address`
`Router(config-if)# frame-relay interface-queue priority 10 20 30 40`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# priority`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# priority`

Configuring Frame Relay Traffic Shaping	

Commands
• frame-relay bc
• frame-relay be
• frame-relay cir
Note In Cisco IOS Release 15.1(3)T, these commands are not hidden, but they are valid only for SVCs (not PVCs).
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# frame-relay bc {in | out} committed-burst-size-in-bits`
`Router(config-map-class)# frame-relay be {in | out} excess-burst-size-in-bits`
`Router(config-map-class)# frame-relay cir {in | out} bits-per-second`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`

Configuring Frame Relay Traffic Shaping on a VC	

Commands
• frame-relay traffic-rate
Command Usage
`Router(config)# map-class frame-relay map-class-name`
`Router(config-map-class)# traffic-rate average [peak]`
	Command Usage
	`Router(config)# policy-map policy-map-name`
	`Router(config-pmap)# class class-default`
	`Router(config-pmap-c)# shape average rate`
	`Router(config-pmap-c)# service-policy output traffic-rate` service-policy output traffic-rate

Displaying the Contents of Packets Inside a Queue for an Interface or VC	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence
Commands • show queue **Command Usage** `Router# show queue interface`	**Command Usage** `Router# show policy-map interface`
Displaying Queueing Strategies	
Commands • show queueing **Command Usage** `Router# show queueing`	**Command Usage** `Router# show policy-map interface`
Displaying Weighted Random Early Detection (WRED) Information	
Commands • show interfaces random-detect **Command Usage** `Router# show interfaces [type number] random-detect`	**Command Usage** `Router# show policy-map interface`
Displaying WRED Parameter Groups	
Commands • show random-detect-group **Command Usage** `Router# show random-detect-group`	**Command Usage** `Router# show policy-map interface`
Displaying the Traffic-Shaping Configuration, Queueing, and Statistics	

Hidden, Removed or Unsupported Commands	Replacement MQC Command Sequence

Commands
• show traffic-shape
• show traffic-shape queue
• show traffic-shape statistics
Command Usage
`Router# show traffic-shape [interface-type interface-number]`
`Router# show traffic-shape queue [interface-number [dlci dlci-number]]`
`Router# show traffic-shape statistics [interface-type interface-number]`
	Command Usage
	`Router# show policy-map interface`

Displaying Weighted Fair Queueing Information	
Commands • show interfaces fair-queue **Command Usage** `Router# show interfaces [interface-type interface-number] fair-queue`	**Command Usage** `Router# show policy-map interface`

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Defining traffic classes; attaching traffic policies to interfaces	" Applying QoS Features Using the MQC " module in the *Quality of Service Solutions Configuration Guide*
Reference pages for QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*
Reference pages for wide-area networking commands	*Cisco IOS Wide-Area Networking Command Reference*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Legacy QoS Command Deprecation

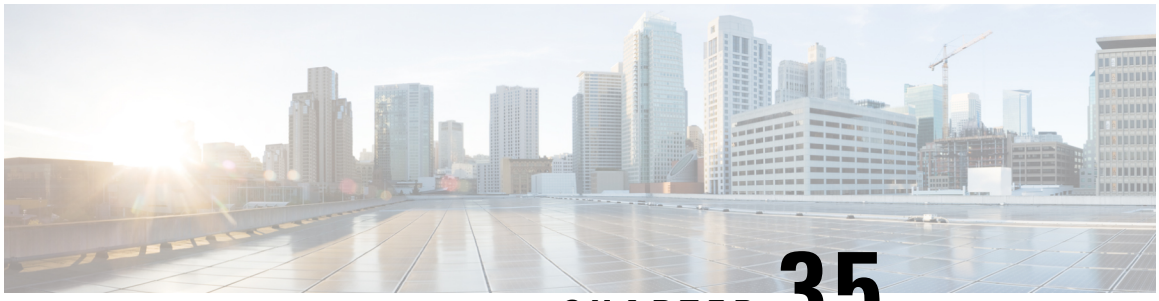
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 38: Feature Information for Legacy QoS Command Deprecation

Feature Name	Releases	Feature Information
Legacy QoS Command Deprecation: Hidden Commands	15.0(1)S 15.1(3)T	To streamline Cisco IOS QoS, certain commands have been hidden, which means that if you try to view a hidden command by entering a question mark (**?**) at the command line, the command does not appear. However, if you know the command syntax, you can enter it. These commands will be removed in a future release. The functionality provided by these hidden commands is replaced by similar functionality from the modular QoS CLI (MQC), which is a set of a platform-independent commands for configuring QoS. The following commands were modified: **custom-queue-list**, **fair-queue (WFQ)**, **frame-relay adaptive-shaping** (**becn** keyword), **frame-relay adaptive-shaping** (**foresight** keyword), **frame-relay bc**, **frame-relay be**, **frame-relay cir**, **frame-relay congestion threshold de**, **frame-relay congestion threshold ecn**, **frame-relay custom-queue-list**, **frame-relay fair-queue**, **frame-relay fecn-adapt**, **frame-relay ip rtp priority**, **frame-relay priority-group**, **frame-relay qos-autosense**, **ip rtp priority**, **max-reserved-bandwidth**, **priority-group**, **random-detect**, **random-detect dscp**, **random-detect**(**dscp-based** keyword), **random-detect exponential-weighting-constant**, **random-detect flow**, **random-detect flow average-depth-factor**, **random-detect flow count**, **random-detect**(**prec-based** keyword), **random-detect precedence**, **random-detect-group**, **show interfaces fair-queue**, **show interfaces random-detect**, **show queue**, **show queueing**, **show random-detect-group**, **show traffic-shape**, **show traffic-shape queue**, **show traffic-shape statistics**.

Feature Name	Releases	Feature Information
Legacy QoS Command Deprecation: Hidden Commands	Cisco IOS XE Release 2.6	To streamline Cisco IOS XE QoS, certain commands have been hidden, which means that if you try to view a hidden command by entering a question mark (**?**) at the command line, the command does not appear. However, if you know the command syntax, you can enter it. These commands will be removed in a future release. The functionality provided by these hidden commands is replaced by similar functionality from the modular QoS CLI (MQC), which is a set of a platform-independent commands for configuring QoS. The following commands were modified: **custom-queue-list**, **fair-queue (WFQ)**, **frame-relay adaptive-shaping** (**becn** keyword), **frame-relay adaptive-shaping** (**foresight** keyword), **frame-relay bc**, **frame-relay be**, **frame-relay cir**, **frame-relay congestion threshold de**, **frame-relay congestion threshold ecn**, **frame-relay custom-queue-list**, **frame-relay fair-queue**, **frame-relay fecn-adapt**, **frame-relay ip rtp priority**, **frame-relay priority-group**, **frame-relay qos-autosense**, **ip rtp priority**, **max-reserved-bandwidth**, **show interfaces fair-queue**, **show interfaces random-detect**, **show queue**, **show queueing**, **show traffic-shape**, **show traffic-shape queue**, **show traffic-shape statistics**.
Legacy QoS Command Deprecation: Removed Commands	Cisco IOS XE Release 3.2S	The legacy QoS commands were removed. This means that you must use the appropriate replacement MQC commands. The following commands were removed: **custom-queue-list**, **fair-queue (WFQ)**, **frame-relay adaptive-shaping** (**becn** keyword), **frame-relay adaptive-shaping** (**foresight** keyword), **frame-relay bc**, **frame-relay be**, **frame-relay cir**, **frame-relay congestion threshold de**, **frame-relay congestion threshold ecn**, **frame-relay custom-queue-list**, **frame-relay fair-queue**, **frame-relay fecn-adapt**, **frame-relay ip rtp priority**, **frame-relay priority-group**, **frame-relay qos-autosense**, **ip rtp priority**, **max-reserved-bandwidth**, **show interfaces fair-queue**, **show interfaces random-detect**, **show queue**, **show queueing**, **show traffic-shape**, **show traffic-shape queue**, **show traffic-shape statistics**.



CHAPTER 35

QoS Packet Marking

QoS Packet Marking refers to changing a field within a packet either at Layer 2 (802.1Q/p CoS, MPLS EXP) or Layer 3 (IP Precedence, DSCP and/or IP ECN). It also refers to preserving any classification decision that was reached previously.

About

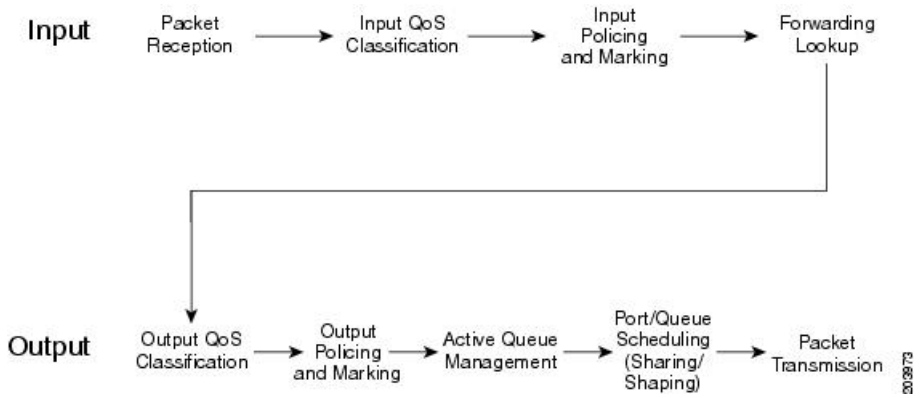
Marking Definition

Marking is similar conceptually to "service class" designation on an airplane ticket: first, business, or economy. This value reflects the level (quality) of service you should receive. Similarly, we mark a value in a packet to indicate the service class (henceforth termed service-class) for that packet as it traverses the network. By examining the marked value, network elements can decide how to treat your packet.

People in business-class may have used a variety of means to achieve that designation. They may have paid extra, used airmiles, or been lucky and booked at the normal rate when no other seat was available. Elsewhere, someone performed the complex task of classification - determining eligibility for a particular service-class then marked the ticket with a mere designation: first-class, business-class, or economy-class. Flight-attendants are unconcerned with how eligibility was determined; they simply look at the class marked on the ticket and provide that level of service.

This dynamic plays out in the networking world. One device may perform complex classification on the data in a flow, determining an appropriate service-class. Other network elements "trust" the value marked in packets they receive and provide service appropriate for that designation.

Figure 67: QoS Packet Processing



Within the context of QoS packet processing, marking occurs after classification and before queuing and is applicable on ingress or egress.

Typically, you would create a *trust boundary* at the edge of the network, then classify and mark packets on the edge device. Then, you would use that marked field for classification and determination of per-hop treatment throughout the network.

Note A trust boundary enables you to apply network-controlled marking on all packets as they enter the network and to remove or modify any non-default markings you did not apply.

Imagine that your system recognizes router ports with attached VoIP devices. You could mark the differentiated services code point (DSCP) value of voice packets as EF (at the edge of the network) and employ DSCP-based classification throughout the network to determine those packets that warrant low latency treatment.

Why Mark Packets

Reasons for marking packets include the following:

- Indicate the treatment you would like a packet to receive as it traverses the network.
- Perform complex classification once. By marking the service class, you can use simpler, less cpu-intensive classification elsewhere in the network.
- Perform classification at a point in the network where you have greater visibility into the flow. For example, if data is encrypted, you cannot perform complex classification such as determining the application carried within that flow. Instead, you could classify prior to encryption and mark a value in the unencrypted header that is visible to network elements along the path.

  As a packet traverses networks managed by different autonomous entities (e.g., the service provider network between two enterprise offices), you may need to re-mark if the markings to service-level designations are inconsistent across those networks.

  As a packet traverses different networking technologies the fields available to indicate service-class may differ. For example, you might carry service-class designation in the DSCP field of an IP packet but if this packet traverses an the multiprotocol label switching (MPLS) network only the MPLS experimental (EXP) field may be usable by network elements to determine service-class. As you enter that portion of the network, you may need to determine the appropriate marking of the MPLS EXP bits.

As a network operator you may contract to accept data from a user at a certain rate. Rather than dropping packets that exceed that rate, you can mark them as a lesser service-class.

Approaches to Marking Packets

You have two main approaches to marking packets: the **set** command and a policer marking action.

Note We only briefly touch upon "policing" actions within this chapter.

set Command

The simplest approach to marking packets on a router is to use the **set** command in a *policy-map* definition. (A policy-map is where you specify a QoS action for each class of traffic that you have defined).

You may decide to classify all RTP ports into a traffic class and mark each packet with AF41. If so, the policy-map may look something like this:

```
policy-map mark-rtp
  class rtp-traffic
    set dscp af41
```

Policer Marking Action

Recall that you can use a policer to drop packets within a traffic class above a defined rate. Alternatively, you could mark packets above that rate and allow them to receive a different per-hop treatment than packets below that rate.

For example, let's say that video traffic arrives at your router marked AF41. You may decide to consider user traffic up to 2 Mbps top *assured forwarding behavior* and to demote any traffic exceeding 2Mbps to AF42 (and considered *out of contract* - non-conforming).

The policy-map might appear as follows:

```
class-map video-traffic
  match dscp af41
!
policy-map enforce-contract
  class video-traffic
    police cir 2m conform-action transmit exceed-action set-dscp-transmit AF42
```

Scope of Marking Action

Similar to classification, marking cannot access every field within a data packet. For example, if an IP packet is encapsulated in multiprotocol label switching (MPLS), it cannot mark the DSCP within the IP header as that would require first de-capsulating from MPLS. However, you could mark the MPLS experimental (EXP) bits.

Note Only Layer 2 and outer Layer 3 headers are available for marking.

Multiple Set Statements

You can configure multiple marking rules within a single class (or policer action). This allows you to mark both Layer 2 and Layer 3 fields within the same packet, or if multiple traffic types are present in the same class, define marking values for each type.

For example consider the following egress policy attached to an Ethernet subinterface:

```
policy-map mark-rtp
  class rtp-traffic
    set cos 4
    set mpls exp topmost 4
    set dscp af41
```

If an MPLS packet were forwarded through this subinterface, the Layer 2 COS field and the EXP bits in the MPLS header would be marked. If an IP datagram were encapsulated in that packet, its DSCP value would remain unchanged. However, if an IP packet were forwarded through the subinterface, its Layer 2 COS value and Layer 3 DSCP values would be marked.

For details, refer to the command pages for set cos, on page 486, set mpls experimental topmost, on page 490, and set dscp, on page 487.

Marking Internal Designators

Cisco routers allow you to mark two internal values (qos-group and discard-class) that travel with the packet within the router but do not modify the packet's contents.

Typically, you mark these in an ingress policy and use them to classify to a traffic-class or WRED drop profile in an egress policy. For example, you may want to base your egress classification on a user's IP address but realize that encryption is configured and the user's IP address is invisible on an egress interface. You could classify their traffic on ingress (before encryption) and set an appropriate qos-group value. On egress, you could now classify based on the qos-group and choose the action accordingly.

Ingress vs. Egress Marking Actions

Certain marking values are only relevant to ingress or egress policies. For example, marking the ATM CLP bit or Frame Relay DE bit in an ingress policy is meaningless as they are discarded when the packet is decapsulated. Similarly, marking qos-group or discard-class in an egress policy is unproductive as these leave the packet unchanged and are discarded when we enqueue the packet for forwarding to the next hop.

Imposition Marking

Under special circumstances, you can mark a header field that has not yet been added to a packet (we term this behavior *imposition marking*).

The most common example of imposition marking is the application ofthe **set mpls experimental imposition** command - you can use it on an ingress interface where a packet may arrive containing an IP datagram and no multiprotocol label switching (MPLS) header. When and if the router encapsulates the datagram with a MPLS header, the EXP bits will be marked accordingly as specified by this command.

Application of the **set dscp tunnel** and **set precedence tunnel** commands (for IPv4 only) represent another example of imposition marking. If an egress policy is applied on a tunnel interface, no tunnel header exists when the policy executes. This means that any marking would apply to the original (eventually inner) IP

header. Using either command, you can mark the tunnel (outer) IP header and leave the original header unchanged.

The following table lists the tunnel types and encapsulation variants that support these commands:

When a new header is added (encapsulated), any QoS marking in the inner header is copied to the outer header. For example, when an IP datagram is encapsulated with an MPLS header, the default behavior is to copy the IP Precedence bits from the IP header to the MPLS EXP bits in the newly-imposed header.

Regarding header disposition, we typically do not copy any outer marking(s) to the inner header. For example, at the endpoint for a GRE tunnel, let's say that we receive a packet with different DSCP values in the outer and inner IP headers. When we remove the outer header we do not copy its DSCP value to the inner header.

Configuration Examples

Example 1: Configuring Ingress Marking

You can set up a trust boundary at the edge of a network (where marking is used) to indicate service-class for some traffic and to bleach all other traffic (see *** below). Enforcing a trust boundary at all ingress ports to the network allows you to maintain control of which applications are mapped to each service-class within the network:

```
policy-map ingress-marking
  class voice
    set dscp ef
  class video
    set dscp af41
  class scavenger
    set dscp cs1
  class class-default                                    ***
    set dscp 0                                           ***
!
interface gigabitethernet1/0/0
  Service-policy in ingress-marking
```

For details, refer to the page set dscp, on page 487.

Example 2: Configuring Egress Marking

If a different administrator controls a portion of a network path and uses a different DSCP to service-class mapping, egress marking may be necessary (e.g., within your enterprise, you classify 12 distinct classes of traffic as described in RFC4594). However, your service provider only provides a three-class model.

You may also need egress marking to indicate treatment for certain classes in a Layer 2 network (like Ethernet, frame-relay, or ATM switched networks):

```
policy-map egress-marking
  class scavenger
    set atm-clp
```

Example 3: Configuring MPLS EXP Imposition

With MPLS, a provider edge (PE) router encapsulates datagrams or frames with MPLS headers. Switching decisions within the core are based on the MPLS headers without visibility into the encapsulated data.

Consider a Layer 3 MPLS network where IPv4 datagrams are encapsulated in MPLS headers. On the customer edge (CE) facing interface we have visibility into the IPv4 header of the packet. On the core-facing interface, we have encapsulated datagrams with MPLS headers and we cannot see beyond those headers.

By default, we copy the IP precedence to the MPLS EXP bits. What if we want to override this behavior? We can't parse the IPv4 type of service byte on the core-facing interface. We can, however, parse the IP header on ingress and store the EXP value we plan to set when MPLS headers are added. Although MPLS headers are absent when we execute the command, the router retrieves the instruction and marks the EXP bits on the egress interface:

```
policy-map mpls-exp-remark
  class voice
    set mpls experimental imposition 5
  class video
    set mpls experimental imposition 4
  class scavenger
    set mpls experimental imposition 0
!
interface gigabitethernet1/0/0
  policy-map input mpls-exp-remark
```

For command details, refer to the page set mpls experimental imposition, on page 489.

Example 4: Configuring Tunnel Imposition Marking

Conceptually, tunnel and MPLS EXP imposition marking are similar. We want to mark a value in a header that has not yet been added to the packet and with a Layer 3 tunneling technology like GRE or IPinIP, a Layer 3 datagram may be encapsulated with an outer IP header. (Refer to Imposition Marking, on page 470.)

Let's say that we have a DMVPN network where a branch location encrypts data and encapsulates it with a GRE header before sending it over a public IP network. An administrator may attach a policy-map to the tunnel interface to prioritize applications within that tunnel and may also need to mark the DSCP of the outer IP header to indicate service-class within the provider's network. When the policy is executed, the outer header has not yet been added and commands like **set dscp** or **set precedence** would mark the inner IP header.

To solve the problem, we use the **set dscp tunnel** and **set precedence tunnel** commands, as they allow you to set the value in an outer header that has not yet been added.

In the following example, voice and video traffic are classified and queued separately within the enterprise network. The service provider has a smaller number of service-classes and we have decided to put both voice and video into the priority class within the provider's network.

By marking the DSCP in the outer tunnel header we achieve this yet preserve original markings in the inner header:

```
policy-map mark-outer-gre-header
  class voice
    priority level1 percent 20
    set dscp tunnel ef
  class video
    priority level 2 percent 20
    set dscp tunnel ef
```

```
!
interface tunnel100
  service-policy out mark-outer-gre-header
```

For command details, refer to the page set dscp tunnel, on page 488.

Example 5: Configuring QoS-Group Marking

Occasionally, you may want to base egress queuing on ingress classification. For example, let's say you want more than 8 egress queues on a MPLS-enabled interface. Using egress classification, you are limited to MPLS EXP bits and therefore 8 classes. As a solution, you could perform classification on the ingress interface and set a QoS group for packets that match that classification. QoS group has relevance only within the current router; it doesn't alter anything in the packet header. Instead, it's a value associated with the packet as it passes through the router.

In the following example we use Network Based Application Recognition (NBAR) classification on ingress and mark both telepresence and jabber video with qos-group 4. In the egress policy we classify based on the qos-group we marked on ingress (see "***"):

```
class-map telepresence-video
  match protocol telepresence-media
class-map jabber-video
  match protocol cisco-jabber-video
class-map egress-video-traffic                          ***
  match qos-group 4                                     ***
!
policy-map mark-qos-group
  class telepresence-video
    set qos-group 4
  class jabber-video
    set qos-group 4
!
policy-map egress-queuing
  class egress-video-traffic
    bandwidth remaining percent 50
!
interface gig 1/0/0
  service-policy in mark-qos-group
!
interface serial1/1/0
  service-policy out egress-queuing
```

For command details, refer to the page set qos-group command page.

Example 6: Configuring Discard-Class Marking

In Example 5: Configuring QoS-Group Marking, on page 473, we marked both telepresence video and jabber video with qos-group 4 and placed both of these applications into the same egress queue.

What if we want to run Weighted Random Early Detection (WRED) on the egress queue and drop the jabber video first during congestion. Typically, WRED examines the precedence or DSCP value to determine drop thresholds for a flow. However, as indicated in Example 3: Configuring MPLS EXP Imposition, on page 472, we do not have visibility into the IP header. A solution is to mark a second internal value named discard-class. Then, we could use the qos-group to select the egress class (and queue) and the discard-class to select the WRED drop profile within that class.

```
class-map telepresence-video
  match protocol telepresence-media
class-map jabber-video
  match protocol cisco-jabber-video
class-map egress-video-traffic
  match qos-group 4
!
policy-map mark-qos-group
  class telepresence-video
    set qos-group 4
    set discard-class 1
  class jabber-video
    set qos-group 4
    set discard-class 2
!
policy-map egress-queuing
  class egress-video-traffic
    bandwidth remaining percent 50
    random-detect discard-class-based
    random-detect discard-class 1 24 40
    random-detect discard-class 2 22 30
!
interface gig 1/0/0
  service-policy in mark-qos-group
!
interface serial1/1/0
  service-policy out egress-queuing
```

For command details, refer to the page set discard-class, on page 487.

Verifying QoS Packet Marking

The **show policy-map interface** command is the primary means of verifying any QoS behavior on IOS XE platforms. Although the packet forwarding path (dataplane) is separated from the IOS instance (control plane), statistics are still reported through this well-known IOS command. This functionality is enabled by default.

This table describes the fields we employ in the following sections.

Table 39: show policy-map interface Field Descriptions (those useful for verifying marking)

Field	Description
Service-policy input	Denotes the name of the input service policy applied to the specified interface or VC
Class-map	Specifies the class of traffic being displayed. Output is displayed for each configured class in the policy. The choice for implementing class matches (e.g., match-all or match-any) can also appear adjacent to the traffic class
packets, bytes	Specifies the number of packets (shown in bytes) identified as belonging to the class of traffic being displayed
offered rate	Specifies the rate in bits per second of the packets entering the class
Match	Specifies the match criteria for the traffic class
QoS Set	Details the QoS marking actions configured for the particular class

Field	Description
Packets marked	If enabled, denotes the total number of packets marked for the particular class. If not enabled, you see "Marker statistics: Disabled."

Verifying with the show policy-map interface Command

The **show policy-map interface** command is the primary means of verifying any QoS behavior on IOS XE platforms. Ordinarily, knowing how many packets match a particular class ("class match statistics, " which is enabled by default) and what (if any) marking action is configured suffices to know how many packets were marked by that action.

Note You should understand how *class match statistics* (enabled by default) and *marking statistics* (disabled by default) differ. Typically, the former is sufficient. When a packet "hits" a class, you can assume it is marked. However, if you configure multiple, mutually exclusive marking values, and need to know how many packets were marked with each **set** command, you can enable marking statistics with all its caveats.

Here is an example of ingress marking with a policy attached to a physical interface. In this example, let's say that jabber-video is configured on ports 2000-3000:

```
class-map match-all jabber-video
  match ip rtp 2000 3000
!
policy-map mark-traffic
  class jabber-video
    set dscp af41

show policy-map int g1/0/0
GigabitEthernet1/0/0

  Service-policy input: mark-traffic

    Class-map: jabber-video (match-all)
      850 packets, 51000 bytes                                    note 1
      5 minute offered rate 2000 bps, drop rate 0000 bps
      Match: ip rtp 2000 3000
      QoS Set                                                     note 2
        dscp af41
          Marker statistics: Disabled

    Class-map: class-default (match-any)                          note 3
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Footnotes

note 1	Statistics for the class match
note 2	Packet matching section

note 3	Class-Default statistics section

Observe "`Marker statistics: Disabled`" in the output of ingress marking. If you are invoking multiple statistics and find the information provided in the previous output insufficient, you can enable "Packet Marker Statistics."

Verifying with QoS Packet Marking Statistics

Before you begin

Either

- Remove all policy-maps, issue the command, and re-attach all policy-maps.
- Issue the command, save the configuration, and reload the router.

Note Enabling QoS: Packet Marking Statistics may increase CPU utilization on a scaled configuration. Weigh the benefits of displaying statistics information against the increased CPU utilization for your system.

Enabling QoS Packet Marking Statistics

To enable Packet Marking Statistics, issue the **platform qos marker-statistics** command in configuration mode.

Displaying QoS Packet Marking Statistics

To display the packet statistics of all classes that are configured for all service policies either on the specified interface (or subinterface) or on a specific Permanent Virtual Circuit (PVC), use the **show policy-map interface** command.

When we singularly-configure marking in a policy-map, the output from an ASR 1000 Series Aggregation Services Router would appear as follows:

```
policy-map remark-af41
  class af41-traffic
    set dscp tunnel ef
```

Let's place this map on a tunnel interface with traffic marked af41 in the user's IP header and DSCP marked EF in the GRE IP header. The output of the **show policy-map interface** will appear as follows:

```
show policy-map interface tunnel1

  Service-policy output: remark-af41

    Class-map: af41-traffic (match-all)
      978 packets, 68460 bytes                                  note 1
      5 minute offered rate 2000 bps, drop rate 0000 bps
      Match:  dscp af41 (34)
      QoS Set                                                   note 2
        dscp tunnel ef
          Marker statistics: Disabled                           note 3
```

```
    Class-map: class-default (match-any)
      365 packets, 25550 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Footnotes

note 1	Displays the class match statistics (assume all "observed" packets are marked AF41).
note 2	Marking is the only action configured.
note 3	Per-set action statistics are disabled by default.

Now, if we enable marking statistics, output from the **show policy-map interface** command would appears as follows:

```
show policy-map interface tunnel1

  Service-policy output: remark-af41

    Class-map: af41-traffic (match-all)
      575 packets, 40250 bytes
      5 minute offered rate 1000 bps, drop rate 0000 bps
      Match:  dscp af41 (34)
      QoS Set
        dscp tunnel ef
          Packets marked 575                  note

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Footnote

note	We have now enabled marking statistics but in this example the information is redundant.

For command details, refer to the page set dscp tunnel, on page 488.

Validating the Dataplane Configuration

To verify that the dataplane configuration reflects the IOS control plane configuration, use the **show platform hardware qfp active feature qos interface [input|output]** command, which engages only if issued before you attach any policy-map to an interface. So, you must do one of the following:

- Remove all policy-maps, issue the command and re-attach all policy-maps.
- Issue the command, save the configuration and reload the router.

In the following output, notice that we have configured the actions and set the values on the dataplane:

```
show platform hardware qfp active feature qos interface g1/0/0 input
```

```
Interface: GigabitEthernet1/0/0, QFP interface: 12
  Direction: Input
    Hierarchy level: 0
    Policy name: mark-traffic
      Class name: jabber-video, Policy name: mark-traffic
        QOS Set:
            dscp 34                                                    note
      Class name: class-default, Policy name: mark-traffic
```

Footnote

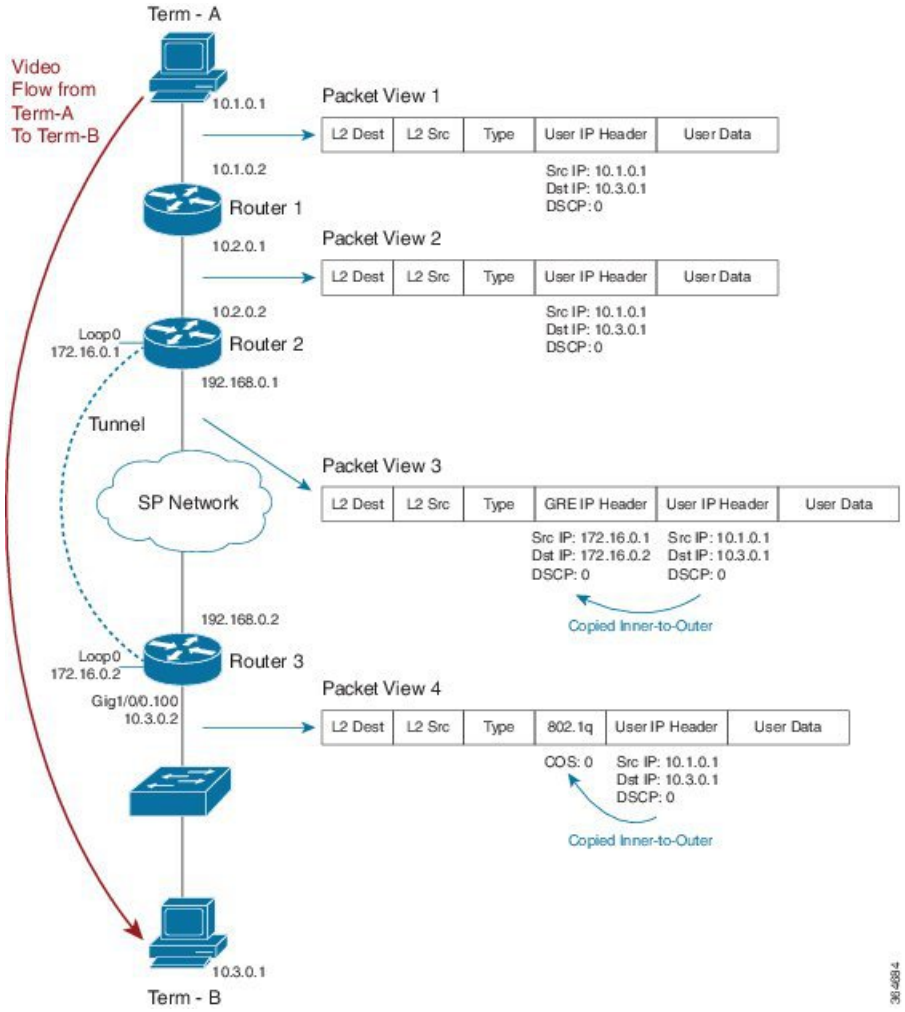
note	Dataplane is programmed to mark.

Network-Level Configuration Examples

In the scenarios that follow, a video-flow transits from Terminal-A to Terminal-B.

Example 1: Propagating Service-Class Information Throughout the Network

Figure 68: Propagating Service-Class Information Throughout the Network

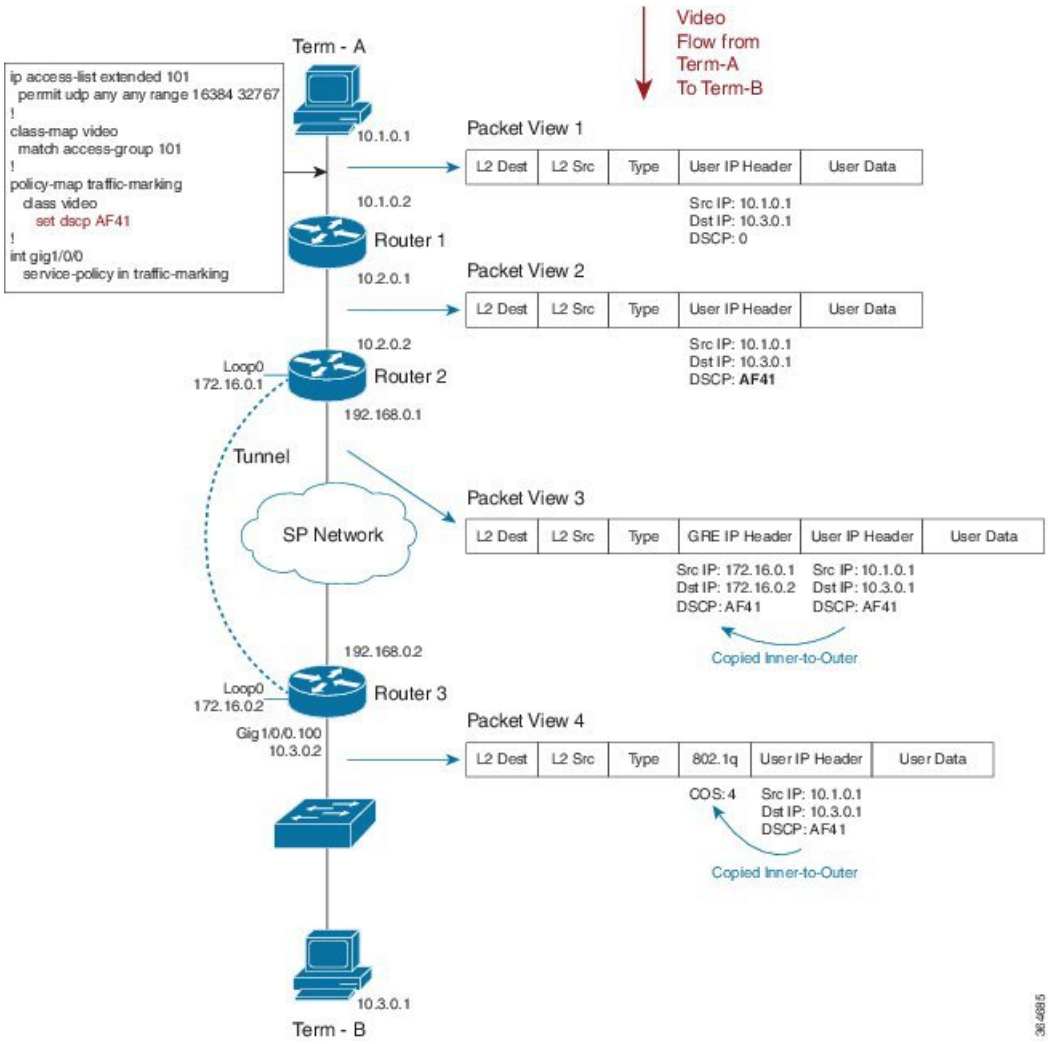


Imagine that an application marks the video stream with DSCP codepoint 0 (see Packet View 1). To cross the provider's network, we send the stream through a GRE tunnel (possibly encrypted). Packet View 3 shows that we have encapsulated the users' IP datagram in a GRE packet. Notice how the DSCP codepoint is copied by default to the imposed GRE header.

With the last hop at the final destination, Router 3 sends a VLAN tagged packet to a switch (see Packet View 4). Observe that the GRE header was stripped and a Dot1Q header was added due to the VLAN configuration. The precedence portion of the user's DSCP 0 (000 000) is copied by default to the COS bits of the VLAN header. The COS value set is 0 (000).

Example 2: Indicating Service-Class by Marking at the Network's Edge

Figure 69: Indicating Service-Class by Marking at the Network's Edge



In this example, we modify the default behavior by remarking the DSCP of users' traffic in an ingress policy as it enters Router 1. The following code shows how we do this:

```
ip access-list extended 101
 permit udp any any range 16384 32767
!
class-map video
 match access-group 101
!
policy-map traffic-marking
  class video
    set dscp AF41
!
int gig1/0/0
  service-policy in traffic-marking
```

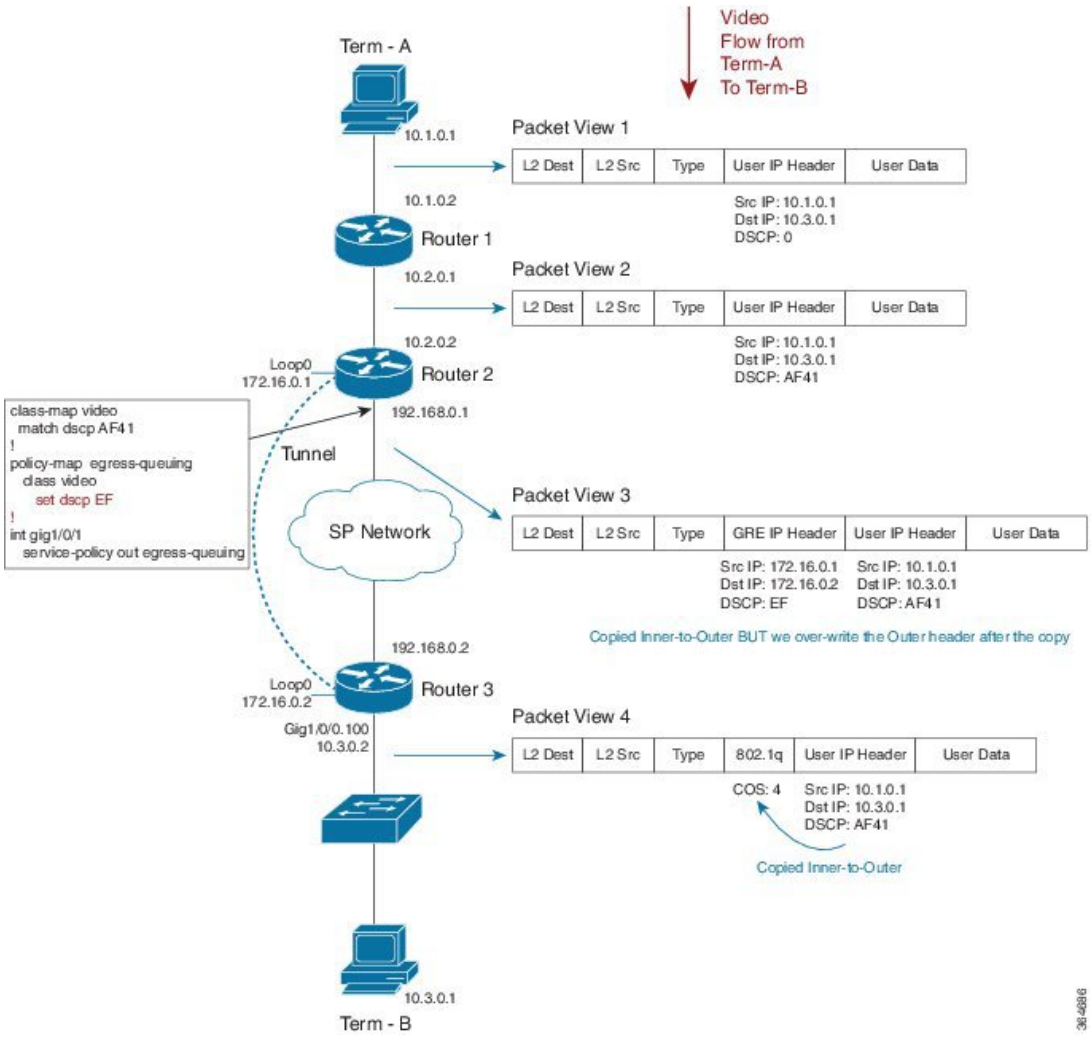
Let's say that we designate video traffic as DSCP AF41 throughout the network. When the packet reaches the GRE interface on egress, its DSCP value has already been changed to AF41 and its behavior matches that in Example 1. We send the stream through a GRE tunnel (possibly encrypted) as it traverses the providers network. Notice how the newly-marked DSCP codepoint (AF41) is copied by default to the imposed GRE header.

When we arrive at our destination, the router sends a VLAN-tagged packet to the last hop (a switch). The precedence portion of the users' DSCP value is copied by default into the COS bits of the VLAN header. As our DSCP is now AF41 (100 010), the COS value will be 4 (100).

For command details, refer to the command page set dscp, on page 487.

Example 3: Remarking Traffic to Match Service Provider Requirements

Figure 70: Remarking Traffic to Match Service Provider Requirements



In this example, we mark the DCSP value within the network while the service provider anticipates a different marking. The following code shows how we handle this:

```
class-map video
  match dscp AF41
!
policy-map  egress-queuing
  class video
    set dscp EF
!
int gig1/0/1
  service-policy out egress-queuing
```

We mark DCSP as AF41 for video within out network while the service provider expects video packets to be marked EF. On the egress Gig interface of Router 2, we add a policy that contains queuing commands (recall that we are only focusing on the marking portion of the configuration in this example).

When the packet reaches the egress physical interface it already has the GRE header imposed and we copy the DSCP value of AF41 from the inner encapsulated datagram. The policy on the physical interface changes the DSCP value in the outer GRE header only.

Note Notice how the inner-user datagram IP header is unchanged.

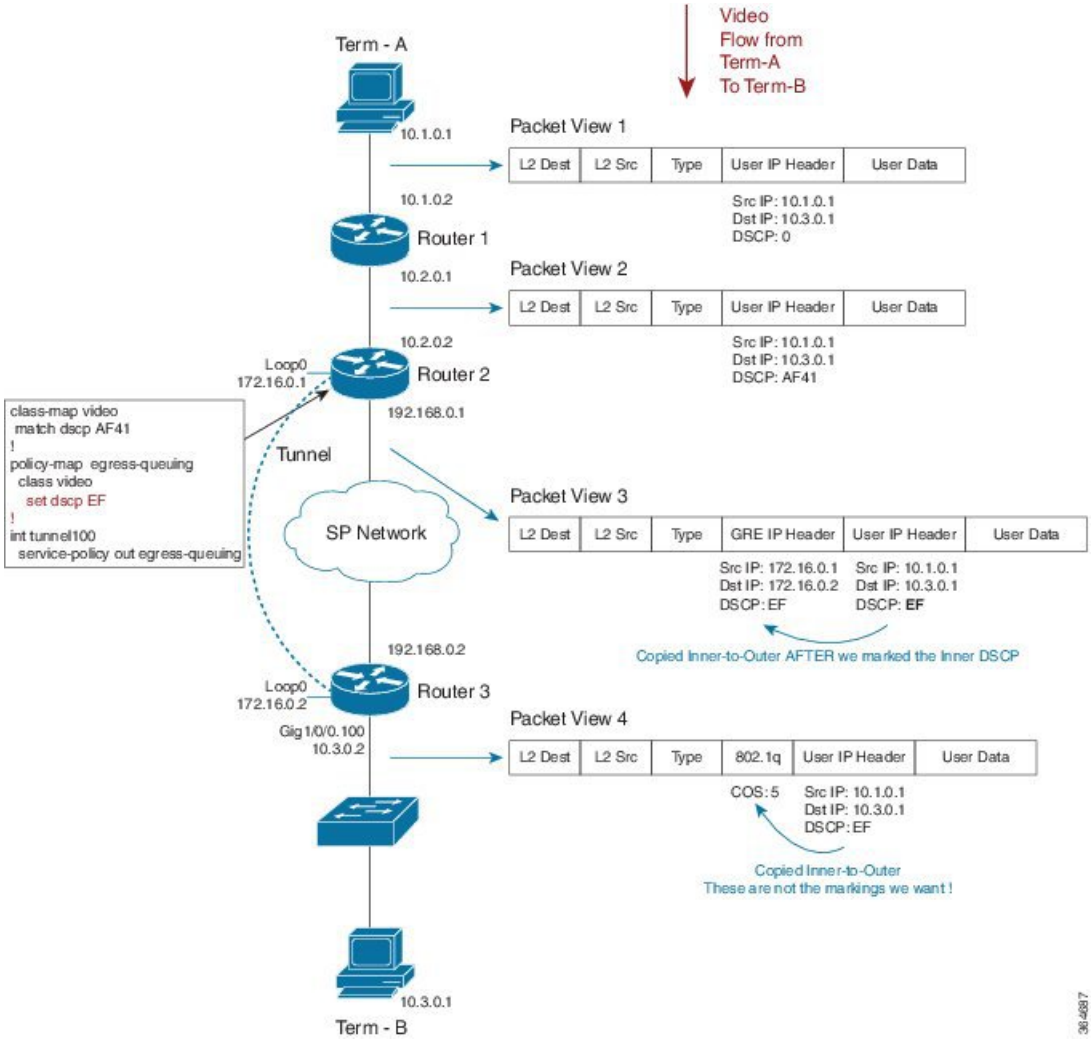
When we reach Router 3 and exit the tunnel, the tunnel GRE header is stripped. Henceforth, only the user datagram IP header is visible, still preserving the AF41 value we marked on ingress to the network.

As in previous examples, the router sends a VLAN-tagged packet to the last hop (a switch). By default, the precedence portion of the User IP Header's DSCP value is copied into the COS bits of the VLAN header (802.1q). As the DSCP value is currently af41 (100 010), the COS value will be 4 (100).

For command details, refer to the page set dscp, on page 487.

Example 4: Remarking on a Tunnel Interface for an SP Network - Potential Gotcha

Figure 71: Remarking on a Tunnel Interface for an SP Network - Potential Gotcha



In this example, we place the QoS policy on the tunnel interface of Router 1 rather than on the physical interface. (There are many advantages to configuring queuing per tunnel rather than as an aggregate policy on the physical interface.) The following code shows how we do this:

```
class-map video
  match dscp AF41
!
policy-map  egress-queuing
  class video
    set dscp EF
!
int tunnel100
```

```
service-policy out egress-queuing
```

We focus solely on the marking portion of the policy. The key point is that marking on the tunnel interface is performed before the tunnel headers are added.

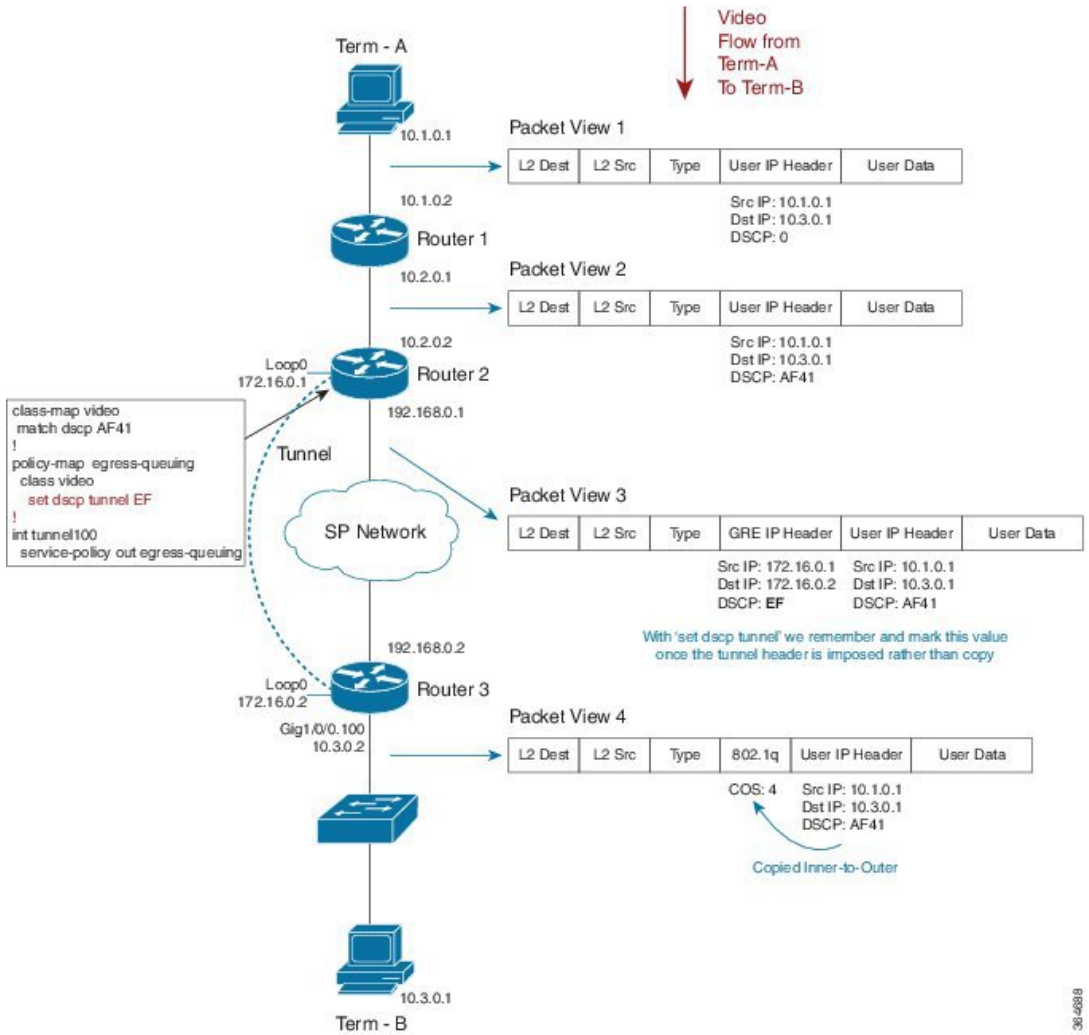
Notice how our policy has over-written the DSCP in the user datagram IP header. Because this happened before GRE encapsulation, we copy the newly-marked value to the outer header.

When we reach Router 3 and exit the tunnel the tunnel GRE header is stripped. Because we marked the user datagram header, the new value propagates through the rest of the network. This is not the behavior we wanted.

For command details, refer to the page for set dscp, on page 487.

Example 5: Using Tunnel Imposition Marking to Remark for an SP Network

Figure 72: Using Tunnel Imposition Marking to Remark for an SP Network



In this example, we use the **set dscp tunnel** *dscp-value* command to alter only the tunnel IP Header:

```
class-map video
  match dscp AF41
!
policy-map  egress-queuing
  class video
    set dscp tunnel EF
!
int tunnel100
  service-policy out egress-queuing
```

We have a QoS policy on the tunnel interface of Router 2 and we have used the **set dscp tunnel** command rather than **set dscp** command.

We have yet to impose the GRE header. The **set dscp tunnel** command dictates that we remember the DSCP value; during encapsulation we use this value instead of copying "inner to outer." Observe that the DSCP value in the users IP datagram header is unchanged. The **set dscp tunnel** command will alter only the tunnel IP header.

For command details, refer to the page for set dscp tunnel, on page 488.

Command Reference

platform qos marker-statistics

To enable individual statistics collection for each marking action in every policy configured on the router, use the **platform qos marker-statistics** command in global configuration mode. To disable packet marking statistics, use the **no** form of this command.

[no] platform qos marker-statistics

Syntax Description

This command has no arguments or keywords.

Command Default

Disabled (no packet marking statistics are displayed). The network operator relies on class match statistics.

Command Modes

policy-map (config-pmap)

Usage Guidelines

This command executes only if issued before any policy-map is attached to an interface. So, you must do one of the following:

- Remove all policy-maps, issue the command and re-attach all policy-maps.
- Issue the command, save the configuration and reload the router.

Note Enabling packet marking statistics may increase CPU utilization on a scaled configuration. So, weigh the benefits of the statistics information against the increased CPU utilization for your system.

set atm-clp

To set the ATM cell loss priority (CLP) bit, use the **set atm-clp** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set atm-clp

Syntax Description

This command has no arguments or keywords.

Command Default

The ATM CLP bit is not set.

Command Modes

policy-map (config-pmap)

Usage Guidelines

On ATM interfaces, you can use the **set atm-clp** command in an outbound policy to set the ATM-CLP bit in ATM cell headers to 1.

This command is supported for ATM, PPPoA, PPPoEoA and L2TPv3 encapsulations. It is not supported if the policy is attached to a tunnel rather than directly to the VC.

You cannot attach a policy-map containing ATM set cell loss priority (CLP) bit QoS to PPP over X (PPPoX) sessions. The map is accepted only if you do not specify the **set atm-clp** command.

For an example using the **set atm-clp** command to configure egress marking, please refer to Example 2: Configuring Egress Marking, on page 471.

set cos

To set the Layer 2 class of service (CoS) value of an outgoing packet, use the **set cos** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set cos *cos-value*

Syntax Description

cos-value	Specifies the IEEE 802.1Q CoS value of an outgoing packet ranging from 0 to 7

Command Default

Either IP Precedence or MPLS EXP bits are copied from the encapsulated datagram.

Command Modes

policy-map (config-pmap)

Usage Guidelines

You can use the **set cos** command to propagate service-class information to a Layer 2 switched network. Although a Layer 2 switch may not be able to parse embedded Layer 3 information (such as DSCP), it might be able to provide differentiated service based on CoS value. Switches can leverage Layer 2 header information, including the marking of a CoS value.

Traditionally the **set cos** command had meaning only in service policies that are attached in the egress direction of an interface because routers discard Layer 2 information from received frames. With the introduction of features like EoMPLS and EVC, the setting of CoS on ingress has meaning, such that you can preserve Layer 2 information throughout the routed network.

set cos-inner

To set the Layer 2 CoS value in the inner VLAN tag of a QinQ packet, use the **set cos-inner** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set cos-inner *cos-value*

Syntax Description

cos-value	Specifies a IEEE 802.1q CoS value ranging from 0-7

Command Default

Either IP Precedence or MPLS EXP bits are copied from the encapsulated datagram.

Command Modes

policy-map (config-pmap)

Usage Guidelines

Traditionally, because routers discard Layer 2 information from received frames, the **set cos-inner** command had meaning only in service policies that are attached in the egress direction of an interface. With the introduction of features like EoMPLS and EVC, the setting of CoS on ingress has significance as you can preserve Layer 2 information throughout the routed network.

set discard-class

To set the QoS discard class for a packet, use the **set discard-class** command in policy-map configuration mode. To disable this setting, use the **no** form of this command.

[no] set discard-class *discard-class-value*

Syntax Description

discard-class-value	Specifies a Discard Class value ranging from 0 to 7

Command Default

The discard-class value associated with a packet is set to 0.

Command Modes

policy-map (config-pmap)

Usage Guidelines

The **set discard-class** command allows you to associate a discard class value with a packet while processed by the router. Setting this value leaves the packet unchanged.

You can use the discard class and discard-class based WRED in egress policies to control which packets are dropped during congestion.

set dscp

To set the DSCP value in the IP header, use the **set dscp** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set dscp *dscp-value*

Syntax Description

dscp-value	Sets the DSCP value in an IP header ranging from 0 to 63. You can specify the value numerically or by using it's well known DiffServe name (e.g., EF)

Command Default

Retain the existing DSCP value in the received packet.

Command Modes

policy-map (config-pmap)

Usage Guidelines

The command may be used in ingress or egress policies.

You can use the DSCP value to indicate the QoS treatment a packet should receive as it traverses a network.

Note The differentiated services architecture using DSCP supersedes use of precedence.

This command marks packets where the outermost Layer 3 header is either IPv4 or IPv6.

If issued in an egress policy-map, this command will not alter the class or queue selection but might influence the WRED drop profile selection.

The **set dscp** and **set ip dscp** commands behave identically, marking both IPv4 and IPv6 packets.

Note This differs from the process of classification wherein the **match ip dscp** command classifies only IPv4 packets while the **match dscp** command classifies both IPv4 and IPv6 packets.

set dscp tunnel

To set the DSCP value in a tunnel header that has not yet been added to a packet, use the **set dscp tunnel** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set dscp tunnel *dscp-value*

Syntax Description

dscp-value	Specifies the DSCP value in a tunnel header ranging from 0 to 63. You can either specify the value numerically or use its well known DiffServe name (e.g. EF).

Command Default

DSCP value from an encapsulated datagram is copied to the newly-imposed tunnel header.

Command Modes

policy-map (config-pmap)

Usage Guidelines

This command only makes sense before a tunnel header is added.

Note You can use this command in either an ingress or egress policy that is attached to a tunnel interface. However, if the latter is attached, the command has no meaning because all headers would be added when the policy is evaluated.

On the Cisco ASR Series Aggregation Services Router, the **set dscp tunnel** command is supported for IPv4 only. See Imposition Marking, on page 470 for a table that lists the supported DSCP tunnel marking configurations.

For an example using this command to encapsulate a Layer 3 datagram with an outer IP header, please refer toExample 4: Configuring Tunnel Imposition Marking, on page 472.

set fr-de

To set the frame-relay (FR) discard eligible (DE) bit, use the **set fr-de** command in policy-map class configuration mode. To disable the setting, use the **no** form of this command.

[no] set fr-de

Syntax Description This command has no arguments or keywords.

Command Default The DE bit is not set when datagrams are encapsulated with frame relay.

Usage Guidelines On serial interfaces configured with Frame Relay encapsulation, you can use the **set fr-de** command in an outbound policy to set the Discard Eligible bit in the Frame Relay header to 1.

set ip dscp

To preserve backwards compatibility, we support two command variants that perform identical functions: **set ip dscp** and **set dscp**. You can use either to mark the DSCP value in the IP header.

set ip dscp tunnel

To preserve backwards compatibility, we support two command variants that perform identical functions: **set ip dscp tunnel** and **set dscp tunnel**.

set ip precedence

To preserve backwards compatibility, we support two command variants that perform identical functions: **set ip precedence** and **set precedence**. You can use either to mark the precedence value in the IP header. Please refer to the **set precedence** command page (set precedence, on page 490) for more information.

set ip precedence tunnel

To preserve backwards compatibility, we support two command variants that perform identical functions: **set ip precedence tunnel** and **set precedence tunnel**. Please refer to the **set precedence tunnel** command page (set precedence tunnel, on page 491) for more information.

set mpls experimental imposition

To set the value of the MPLS EXP field on all imposed label entries, use the **set mpls experimental imposition** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set mpls experimental imposition *mpls-exp-value*

Syntax Description

mpls-exp-value	Specifies the MPLS EXP value, which ranges from 0 to 7

Command Default

MPLS value is copied from the appropriate field (usually precedence) in the encapsulated packet.

Command Modes

policy-map (config-pmap)

Usage Guidelines

The **set mpls experimental imposition** command is supported only on input interfaces. Use this command during label imposition to set the MPLS EXP field on all imposed label entries.

For an example of using this command to set the EXP bits in an MPLS header that we use to encapsulate the datagram or frame, please refer to Example 3: Configuring MPLS EXP Imposition, on page 472.

set mpls experimental topmost

To set the MPLS EXP field value in the topmost label, use the **set mpls experimental topmost** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no]set mpls experimental topmost *mpls-exp-value*

Syntax Description

mpls-exp-value	Specifies the MPLS EXP value ranging from 0 to 7

Command Default

The MPLS EXP value is either copied from the innermost header on encapsulation or remains unchanged.

Command Modes

policy-map (config-pmap)

Usage Guidelines

This command marks packets provided the outermost Layer 3 header is an MPLS label when the command is evaluated.

This command sets the MPLS EXP value in the topmost label only. If multiple labels exist in a stack, the MPLS EXP value in labels other than the topmost remain unchanged.

set precedence

To set the IP Precedence value in the packet header, use the **set precedence** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set precedence *precedence-value*

Syntax Description

precedence-value	Sets the precedence bit in the packet header, which ranges from 0 to 7

Command Default

Retain the precedence value in the received packet.

Command Modes

policy-map (config-pmap)

Usage Guidelines

The command may be used in ingress or egress policies. However, if you issue the command in an egress policy-map, it will not alter the class or queue selection but it may influence the WRED drop profile selection.

By setting a precedence value, you indicate the QoS treatment a packet should receive as it traverses a network.

Note The differentiated services architecture using DSCP largely supersedes the use of precedence.

The **set precedence** and **set ip precedence** commands behave identically, marking packets where the outermost Layer 3 header is IPv4 or IPv6. In contrast, the **match ip precedence** command classifies only IPv4 packets while the **match precedence** command classifies both IPv4 and IPv6.

set precedence tunnel

To set the IP precedence value in a tunnel header that has not yet been added to a packet, use the **set precedence tunnel** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set precedence tunnel *precedence-value*

Syntax Description

precedence-value	Sets the precedence bit in the tunnel header ranging from 0 to 7

Command Default

DSCP (and the precedence portion) are copied from the encapsulated to the newly-imposed header.

Command Modes

policy-map (config-pmap)

Usage Guidelines

On the Cisco ASR Series Aggregation Services Router, the **set precedence tunnel** command is supported for IPv4 only. See Imposition Marking, on page 470 for a table that lists the supported DSCP tunnel marking configurations.

set qos-group

To set the QoS group identifier (ID) for a packet, use the **set qos-group** command in policy-map class configuration mode. To disable this setting, use the **no** form of this command.

[no] set qos-group *group-id*

Syntax Description

group-id	Specifies a QoS group ID ranging from 0 to 99

Command Default

QoS group-id defaults to 0.

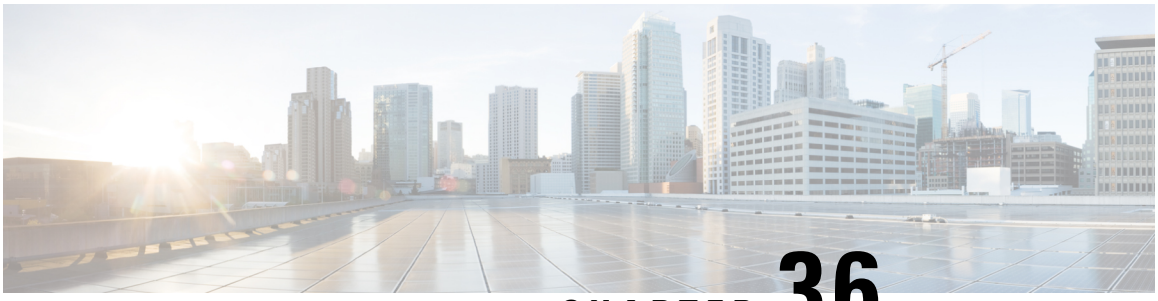
Command Modes

policy-map (config-pmap)

Usage Guidelines

The **set qos-group** command allows you to associate a group ID with a packet as it is processed by the router.

You can use the group ID in egress policies to classify packets to service-classes. Historically, this action had no meaning because we chose the service-class before egress marking occured. With color-aware policing, however, setting the QoS group ID in an egress policy can have meaning.



CHAPTER 36

QoS Packet-Matching Statistics Configuration

The QoS Packet-Matching Statistics feature comprises the following subfeatures:

- The QoS Packet-Matching Statistics: Per Filter feature allows users to count and display the number of packets and bytes matching individual filters (match statements) within a QoS class-map.
- The QoS Packet-Matching Statistics: Per ACE feature allows users to count and display the number of packets and bytes matching the individual access control entries (ACEs) in the filter.

Prerequisites for QoS Packet-Matching Statistics Feature

You cannot enable or disable the QoS Packet-Matching Statistics: Per Filter feature if a policy-map is associated with any interface on the system.

The QoS Packet-Matching Statistics: Per ACE feature is dependent on the QoS Packet Matching Statistics feature. Therefore, the following prerequisites apply:

- If the QoS Packet-Matching Statistics: Per Filter is not enabled and a user tries to enable the QoS Packet-Matching Statistics: Per ACE feature, the command to enable this feature will be rejected by the CLI. An informational message will be displayed to let the user know why the command was rejected.
- If the QoS Packet-Matching Statistics: Per ACE feature is enabled and a user tries to disable this feature, the command to disable this feature will be rejected by the CLI. An informational message will be displayed to let the user know why the command was rejected.

Restrictions for QoS Packet-Matching Statistics Feature

Enabling the Qos: Packet Matching Statistics feature may increase CPU utilization on a scaled configuration. Before enabling the Qos: Packet Matching Statistics feature, weigh the benefits of the statistics information against the increased CPU utilization for your system.

This section provides information about the restrictions pertaining to the QoS Packet-Matching Statistics: Per Filter feature and the QoS Packet-Matching Statistics: Per ACE feature.

The followings are the restrictions for the QoS Packet Matching Statistics feature:

- Enabling the QoS Packet-Matching Statistics: Per Filter feature may increase CPU utilization on a scaled configuration. Before enabling the QoS Packet-Matching Statistics: Per Filter feature, weigh the benefits of the statistics information against the increase in CPU utilization for your system.
- QoS Packet-Matching Statistics: Per Filter is not supported for the match-all class-maps. However, QoS Packet-Matching Statistics: Per ACE is supported for the match-all class-maps.

The following table provides information about the QoS Packet-Matching Statistics: Per ACE scaling limitations:

Table 40: QoS Packet-Matching Statistics: Per ACE Scaling Limitations

Platform	ACEs (IPv4 or IPv6)
ASR1000-ESP5, ASR1001, ASR1002-F, ASR1002-X	25,000
ASR1000-ESP10	30,000
ASR1000-ESP20/ESP40/ESP100	30,000
ISR4400	20,000
CSR1000V	1,000

Information About QoS Packet-Matching Statistics

This section provides an overview of the QoS Packet-Matching Statistics: Per Filter feature and the QoS Packet-Matching Statistics: Per ACE feature.

QoS Packet-Matching Statistics: Per Filter Feature Overview

The QoS Packet-Matching Statistics: Per Filter feature allows you to count and display the number of packets and bytes matching a filter.

To define a filter, use the **class-map** command with the **match-any** keyword, for example:

```
class-map match-any my_class
   match ip precedence 4 <-------- User-defined filter
   match qos-group 10 <-------- User-defined filter
```

Using this information, you can perform the following tasks:

- Compare the amount of voice traffic with the amount of data traffic on a segment of your network
- Adjust bandwidth availability
- Accurately determine billing
- Troubleshoot service problems

The system collects packet matching statistics in 10-second cycles. If there are many interfaces or sessions, the system collects statistics for about 8000 of them during each cycle. In a scaled configuration, several 10-second cycles may be required to gather all the statistics.

QoS Packet-Matching Statistics: Per ACE Feature Overview

The QoS Packet-Matching Statistics: Per ACE feature allows you to track and display the number of packets and bytes matching individual ACEs that are used in QoS policies (access groups used in class maps).

This feature provides hit counters for ACEs used in QoS policies. When this feature is enabled, it will add QoS hit counters for the ACEs used in a QoS policy to the existing security access list counters for that particular ACE. The access list counters can be seen in the following command output:

```
Router# show ip access-lists

Extended IP access list A1
    10 permit ip 32.1.1.0 0.0.0.255 any (129580275 matches)
Extended IP access list A6and7
    10 permit ip 32.1.6.0 0.0.0.255 any (341426749 matches)
    20 permit ip 32.1.7.0 0.0.0.255 any (398245767 matches)
Extended IP access list source
    10 permit ip any host 16.1.1.5 (16147976 matches)
```

The QoS hit counters (for the ACEs used in QoS policies) will be added to the access list counters. We recommend that you pay attention to the following points when you enable this feature:

- Access list counts are not interface specific, as can be seen in the output of the **show ip access-lists** command (there is no mention of interface). They are aggregate counters of all the hits, for all the features that use the ACEs and support the counts, across all interfaces and directions.
- Interface-specific counts are provided in the existing QoS command(**show policy-map interface**) if the QoS Packet-Matching Statistics: Per Filter feature is enabled. However, the command specified previously shows only the counts per filter (ACL or access group), not per ACE, as can be seen in the following sample output:

```
Router# show access-lists

Extended IP access list A1
    10 permit ip 32.1.1.0 0.0.0.255 any (2000 matches)

Router# show policy-map interface GigabitEthernet0/0/2

  Service-policy input: simple

    Class-map: A1-class (match-all)
      1000 packets, 124000 bytes
      5 minute offered rate 4000 bps
      Match: access-group name A1

    Class-map: class-default (match-any)
```

```
    0 packets, 0 bytes
    5 minute offered rate 261000 bps, drop rate 0 bps
    Match: any
```

- If an ACE is present in a QoS filter (match statement within a class map), but the packet does *not* match the statement, the ACE counter will *not* be incremented for that packet. This can happen if:
  - The ACE is used in a deny statement.
  - Other matching criteria in a match-all class map definition (such as match ip prec 1) prevent the packet from matching the class.
  - Other matching criteria in a match-any class map definition (such as match ip prec 1) match the packet and keep it from matching the ACE match criteria. (This filter precedes the ACE filter and the packet matches both the statements).
- Access list counts are an aggregate (for a particular ACE) of the hit counts for all the features using that ACE, and support the per ACE counts. (In Cisco IOS XE3.10, only Security and QoS ACLs support per ACE counts, but that may change in future releases). Therefore, it is possible that a single packet will hit (and be counted by) multiple features using the same ACE and hence result in multiple counts for the same packet (as it traverses each feature). The following is an example of this:

```
ip access-list extended A1
  permit ip 32.1.1.0 0.0.0.255 any
class-map match-all A1-class
  match access-group name A1

interface GigabitEthernet0/0/2
 ip address 32.0.0.1 240.0.0.0
 ip access-group A1 in
 duplex auto
 speed auto
 media-type rj45
 no negotiation auto
 service-policy input simple


Router# show access-lists

Extended IP access list A1
    10 permit ip 32.1.1.0 0.0.0.255 any (2000 matches)

Router# show policy-map interface GigabitEthernet0/0/2

  Service-policy input: simple

    Class-map: A1-class (match-all)
      1000 packets, 124000 bytes
      5 minute offered rate 4000 bps
      Match: access-group name A1

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 261000 bps, drop rate 0 bps
      Match: any
```

How to Configure QoS Packet-Matching Statistics

This section provides information about how to configure QoS Packet-Matching Statistics.

Configuring QoS Packet-Matching Statistics: Per Filter

Before you begin

- Before enabling the QoS Packet-Matching Statistics: Per Filter feature, ensure that no policy-maps are associated with the interfaces on the system. If they are, the system returns the following message:

```
Either a) A system RELOAD or
       b) Remove all service-policies, re-apply the change
            to the statistics, re-apply all service-policies
   is required before this command will be activated.
```

- Before enabling the QoS Packet-Matching Statistics: Per Filter feature, ensure that you have defined a filter that is using the **class-map** command with the **match-any** keyword.

Note Enabling the QoS Packet-Matching Statistics: Per Filter feature may increase CPU utilization on a scaled configuration. Before enabling the QoS Packet-Matching Statistics: Per Filter feature, weigh the benefits of the statistics information against an increase in CPU utilization for your system.

To configure the QoS Packet-Matching Statistics: Per Filter feature, perform the following procedure:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos match-statistics per-filter**
4. **interface** *interface -name*
5. **service-policy** {**input** | **output**} *policy-map-name*
6. **end**
7. **show policy-map interface** *interface-name*
8. **configure terminal**
9. **interface** *interface-name*
10. **no service-policy** {**input** | **output**} *policy-map-name*
11. **exit**
12. **no platform qos match-statistics per-filter**
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables the privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.
Step 3	**platform qos match-statistics per-filter** **Example:** `Router(config)# platform qos match-statistics per-filter`	Enables the QoS Packet-Matching Statistics: Per Filter feature.
Step 4	**interface** *interface -name* **Example:** `Router(config)# interface GigabitEthernet0/0/0`	Specifies the interface for attaching the policy-map.
Step 5	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# service-policy input pol1`	Attaches a QoS policy-map to the interface. The QoS Packet Matching Statistics feature should be enabled before attaching any QoS policies.
Step 6	**end** **Example:** `Router# end`	Exits the configuration mode.
Step 7	**show policy-map interface** *interface-name* **Example:** `Router# show policy-map interface serial4/0/0`	Displays the packet statistics of all the classes that are configured for all the service policies that are present on the specified interface, subinterface, or a specific permanent virtual circuit (PVC) on the interface.
Step 8	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.
Step 9	**interface** *interface-name* **Example:** `Router(config)# interface GigabitEthernet0/0/0`	Specifies the interface for removing the policy-map.

	Command or Action	Purpose
Step 10	**no service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# no service-policy input pol1`	Removes a QoS policy-map from an interface. All the QoS policies should be removed from the interfaces before the QoS Packet Matching Statistics feature can be disabled.
Step 11	**exit** **Example:** `Router(config-if)# exit`	Exits the interface configuration mode.
Step 12	**no platform qos match-statistics per-filter** **Example:** `Router(config)# no platform qos match-statistics per-filter`	Disables the QoS Packet-Matching Statistics: Per Filter feature.
Step 13	**end** **Example:** `Router# end`	Exits the configuration mode.

Examples

Use the **show policy-map interface** command to display the packet statistics of all the classes that are configured for all the service policies that are present on the specified interface, subinterface, or a specific PVC on the interface:

```
Router# show policy-map interface gig1/1/0

 GigabitEthernet1/1/0
  Service-policy input: pol1     ! target = gig1/1/0,input
    Class-map: class1 (match-any)
      1000 packets, 40000 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: ip precedence 1 <-------- User-defined filter
        800 packets, 32000 bytes <-------- Filter matching results
      Match: ip precedence 2 <-------- User-defined filter
        200 packets, 8000 bytes <-------- Filter matching results
      QoS Set
        ip precedence 7
          No packet marking statistics available
    Class-map: class-default (match-any)
      500 packets, 20000 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any <-------- User-defined filter
        500 packets, 20000 bytes <-------- Filter matching results
```

Configuring QoS Packet-Matching Statistics: Per ACE

Before you begin

Before enabling the QoS Packet-Matching Statistics: Per ACE feature, ensure that the QoS Packet-Matching Statistics: Per Filter feature has been enabled.

The following example shows how to check the feature status by using the **show platform hardware qfp active feature qos configuration global** command:

```
Router# show platform hardware qfp active feature qos configuration global
Marker statistics are: disabled
Match per-filter statistics are: enabled <<<<<<<
Match per-ace statistics are: enabled <<<<<<
Performance-Monitor statistics are: disabled
```

To configure the QoS Packet-Matching Statistics: Per ACE feature, perform the following procedure:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos match-statistics per-filter**
4. **platform qos match-statistics per-ace**
5. **interface** *interface-name*
6. **service-policy**{**input**|**output**}*policy-map-name*
7. **end**
8. **show policy-map interface** *interface-name*
9. **show access-lists**
10. **configure terminal**
11. **interface** *interface-name*
12. **no service-policy**{**input**|**output**}*policy-map-name*
13. **exit**
14. **no platform qos match-stat per-ace**
15. **no platform qos match-statistics per-filter**
16. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables the privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.

	Command or Action	Purpose
Step 3	**platform qos match-statistics per-filter** **Example:** `Router(config)# platform qos match-statistics per-filter`	Enables the QoS Packet-Matching Statistics: Per Filter feature.
Step 4	**platform qos match-statistics per-ace** **Example:** `Router(config)# platform qos match-statistics per-ace`	Enables the QoS Packet-Matching Statistics: Per ACE feature.
Step 5	**interface** *interface-name* **Example:** `Router(config)# interface GigabitEthernet0/0/0`	Specifies the interface for attaching the policy-map.
Step 6	**service-policy** {**input**|**output**} *policy-map-name* **Example:** `Router(config-if)# service-policy input pol1`	Attaches a QoS policy-map to an interface. The QoS Matching Statistics feature should be enabled before attaching QoS policies.
Step 7	**end** **Example:** `Router# end`	Exits the configuration mode.
Step 8	**show policy-map interface** *interface-name* **Example:** `Router# show policy-map interface serial4/0/0`	Displays the packet statistics pertaining to all the classes that are configured for all the service policies either on the specified interface, subinterface, or on a specific PVC on the interface.
Step 9	**show access-lists** **Example:** `Router# show access-lists`	Displays the contents of current access lists, including the QoS Packet-Matching Statistics: Per ACE.
Step 10	**configure terminal** **Example:** `Router# configure terminal`	Enters the global configuration mode.
Step 11	**interface** *interface-name* **Example:** `Router(config)# interface GigabitEthernet0/0/0`	Specifies the interface for removing the policy-map.
Step 12	**no service-policy** {**input**|**output**} *policy-map-name* **Example:** `Router(config-if)# no service-policy input pol1`	Removes a QoS policy-map from an interface. All the QoS policies should be removed from the interfaces before the QoS Matching Statistics feature can be disabled.

	Command or Action	Purpose
Step 13	**exit** **Example:** `Router(config-if)# exit`	Exits the interface configuration mode.
Step 14	**no platform qos match-stat per-ace** **Example:** `Router(config)# no platform qos match-stat per-ace`	Disables the QoS Packet-Matching Statistics: Per ACE feature.
Step 15	**no platform qos match-statistics per-filter** **Example:** `Router(config)# no platform qos match-statistics per-filter`	Disables the QoS Packet-Matching Statistics: Per Filter feature.
Step 16	**end** **Example:** `Router# end`	Exits the configuration mode.

Example

Use the **show policy-map interface** command to display the per-filter statistics of all the classes that are configured for all the service policies on the specified interface, subinterface, or on a specific PVC on the interface:

```
Router# show policy-map interface GigabitEthernet0/0/2


  Service-policy input: test-match-types

    Class-map: A1orA2-class (match-any)
      482103366 packets, 59780817384 bytes
      5 minute offered rate 6702000 bps
      Match: access-group name A1
        62125633 packets, 7703578368 bytes
        5 minute rate 837000 bps
      Match: access-group name A2
        419977732 packets, 52077238892 bytes
        5 minute rate 5865000 bps

    Class-map: A3andprec1-class (match-all)
      5673520 packets, 703516480 bytes
      5 minute offered rate 837000 bps
      Match: access-group name A3
      Match: ip precedence 1

    Class-map: A5-class (match-all)
      227101820 packets, 28160625680 bytes
      5 minute offered rate 3351000 bps
      Match: access-group name A5
```

```
    Class-map: A6and7-class (match-all)
      627615840 packets, 77824340228 bytes
      5 minute offered rate 9215000 bps
      Match: access-group name A6and7

    Class-map: A3-class (match-all)
      111548288 packets, 13831987712 bytes
      5 minute offered rate 1675000 bps
      Match: access-group name A3

    Class-map: A4andsource (match-all)
      16115590 packets, 1998333160 bytes
      5 minute offered rate 2513000 bps
      Match: access-group name A4
      Match: access-group name source

    Class-map: class-default (match-any)
      164881212 packets, 20445270288 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any
```

Use the **show ip access-lists** command to display the contents of current access lists (which includes the QoS Packet-Matching Statistics: Per ACE):

```
Router# show ip access-lists

Extended IP access list A1
    10 permit ip 32.1.1.0 0.0.0.255 any (129580275 matches)
Extended IP access list A2
    10 permit ip 32.1.2.0 0.0.0.255 any (486342300 matches)
Extended IP access list A3
    10 permit ip 32.1.3.0 0.0.0.255 any (306738457 matches)
Extended IP access list A4
    10 permit ip 32.1.4.0 0.0.0.255 any (16147975 matches)
Extended IP access list A5
    10 permit ip 32.1.5.0 0.0.0.255 any (294357455 matches)
Extended IP access list A6and7
    10 permit ip 32.1.6.0 0.0.0.255 any (341426749 matches)
    20 permit ip 32.1.7.0 0.0.0.255 any (398245767 matches)
Extended IP access list source
    10 permit ip any host 16.1.1.5 (16147976 matches)
```

Troubleshooting Tips

To confirm that the QoS: Packet Matching Statistics feature is enabled, use the **show platform hardware qfp active feature qos config global** command. If the feature is disabled, you should see a message similar to the following:

Router# **show platform hardware qfp active feature qos config global**

```
Marker statistics are: enabled
Match per filter statistics are: enabled
```

Example: Configuring a QoS Packet-Matching Statistics: Per Filter

The following example shows how to configure a QoS Packet-Matching Statistics: Per Filter, perform the following tasks:

- Define a QoS packet matching filter
- Display the **show policy-map interface** command output

```
Router# show policy-map interface Tunnel1

  Service-policy output: DATA-OUT-PARENT
    Class-map: class-default (match-any)
      4469 packets, 4495814 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any <-------- User-defined filter
      Queueing
      queue limit 416 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 4469/4558380
      shape (average) cir 100000000, bc 400000, be 400000
      target shape rate 100000000
      Service-policy : DATA-OUT
        queue stats for all priority classes:
          Queueing
          queue limit 200 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 4469/4558380
        Class-map: ATM-VTI-RIP-SPK1-DATA (match-any)
          4469 packets, 4495814 bytes <-------- Filter matching results
          5 minute offered rate 0000 bps, drop rate 0000 bps
          Match: access-group 121 <-------- User-defined filter
            4469 packets, 4495814 bytes <-------- Filter matching results
            5 minute rate 0 bps
          QoS Set
            ip precedence 3
              Packets marked 4469
          Priority: 100 kbps, burst bytes 2500, b/w exceed drops: 0
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Quality of service commands	*Cisco IOS Quality of Service Command Reference*

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
CISCO-CLASS-BASED-QOS-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

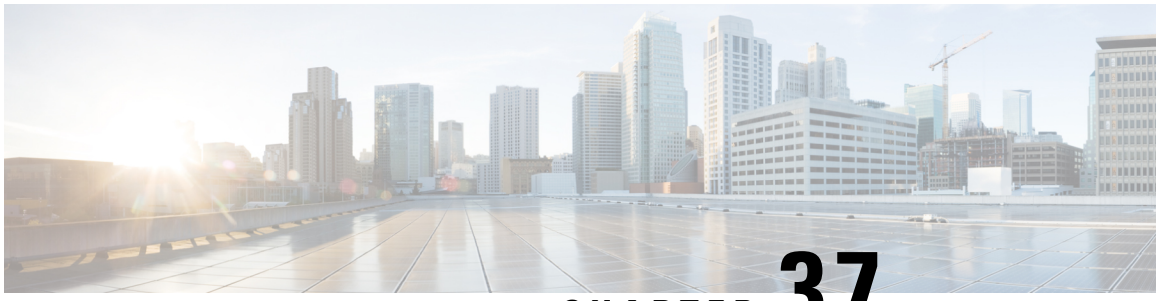
Feature Information for QoS Packet-Matching Statistics

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 41: Feature Information for QoS Packet-Matching Statistics

Feature Name	Releases	Feature Information
QoS Packet-Matching Statistics: Per Filter	Cisco IOS XE Release 3.3S	The QoS Packet-Matching Statistics: Per Filter feature allows you to count and display the number of packets matching individual filters (match statements) used in class-maps within QoS service policies that have. The following commands were introduced or modified: • **platform qos match-statistics per-filter** • **no platform qos match-statistics per-filter** • **show platform hardware qfp active feature qos config global**
QoS Packet-Matching Statistics: Per ACE	Cisco IOS XE Release 3.10S	The QoS Packet-Matching Statistics: Per ACE feature allows you to track and display the number of packets and bytes matching individual ACEs that are used in QoS policies (access groups used in class maps). The following command was introduced: **platform qos match-statistics per-ace**



CHAPTER 37

Set ATM CLP Bit Using Policer

The Set ATM CLP Bit Using Policer feature allows you to police and then mark outbound PPP over ATM (PPPoA) traffic. You can set the ATM cell loss priority (CLP) bit using either of the following methods:

- A policed threshold
- Matching a class

Prerequisites for Set ATM CLP Bit Using Policer

If you are setting the ATM CLP bit by a policed threshold, ensure that a policy-map includes the **set-clp-transmit**action. The new policer action conditionally marks PPPoA traffic in the matched class for a higher drop probability in the ATM network when traffic exceeds a given rate.

If you are setting the ATM CLP bit strictly by matching a class, ensure that a policy-map includes the **set atm-clp** action. The set directive marks all traffic in the matched class for higher drop probability in the ATM network.

You can attach policy-maps with the **set-clp-transmit**or **set atm-clp**actions to a virtual template. This template is cloned when PPPoA sessions are created or by dynamic assignment.

Information About Set ATM CLP Bit Using Policer

ATM CLP Bit

The ATM CLP bit shows the drop priority of the ATM cell. During ATM network congestion, the router discards ATM cells with the CLP bit set to 1 before discarding cells with a CLP bit setting of 0.

Using the Set ATM CLP Bit Using Policer feature, you can configure the **police** command to enable the ATM CLP bit in cell headers. The ATM CLP bit can be explicitly marked by a set directive.

The Set ATM CLP Bit Using Policer feature supports the **set-clp-transmit** policing action in the following types of policies:

- Single-rate policing
- Dual-rate policing
- Hierarchical

How to Set the ATM CLP Bit Using Policer

Configuring PPPoA Broadband Traffic Policing

Before you begin

Before configuring the policy-map, ensure that you have defined any class maps used to classify traffic.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name*| **class-default**]
5. **police** [**cir** *cir*] [**conform-action** *action*] [**exceed-action** *action*]
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map parent-policy`	Enters policy-map configuration mode and creates a policy-map.
Step 4	**class** {*class-name*| **class-default**]	Enters policy-map class configuration mode.

	Command or Action	Purpose
	Example: `Device(config-pmap)# class class-default`	Specifies the name of the class whose policy you want to create or change or specifies the default class (commonly known as the class-default class) before you configure its policy. Repeat this command as many times as necessary to specify the child or parent classes that you are creating or modifying: • *class name* --Name of the class to be configured or whose policy is to be modified. The class name is used for both the class map and to configure a policy for the class in the policy-map. • **class-default** --Specifies the default class so that you can configure or modify its policy.

Step 5	**police** [**cir** *cir*] [**conform-action** *action*] [**exceed-action** *action*]
	Example:
	`Device(config-pmap-c)# police 1000000`
	Example:
	`Router(config-pmap-c-police)# conform-action`
	Example:
	`transmit`
	Example:
	`Device(config-pmap-c-police)# exceed-action`
	Example:
	`set-clp-transmit`
		Configures traffic policing and specifies multiple actions applied to packets marked as conforming to, exceeding, or violating a specific rate.
		• Enters policy-map class police configuration mode. Use one line per action that you want to specify:
		• **cir**--(Optional) Committed information rate. Indicates that the CIR will be used for policing traffic.
		• **conform-action**--(Optional) Action to take on packets when the rate is less than the conform burst.
		• **exceed-action**--(Optional) Action to take on packets whose rate is within the conform and conform plus exceed burst.

Step 6	**end** **Example:** `Device(config-pmap-c)# end`	(Optional) Returns to privileged EXEC mode.

Example

The following example shows you how to set the ATM CLP using a policer:

```
policy-map egress_atm_clp_policer
 class prec0
  police cir 5000000
 class prec1
  police cir 3000000 conform-action transmit exceed-action set-clp-transmit
```

```
class class-default
 police cir 1000000 conform-action transmit exceed-action set-clp-transmit
```

Marking the ATM CLP Bit

Before you begin

Before configuring the policy-map, ensure that you have defined any class maps used to classify traffic.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map policy-map-name**
4. **class** {*class-name*| **class-default**]
5. **set atm-clp**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map policy-map-name** **Example:** `Router(config)# policy-map parent-policy`	Enters policy-map configuration mode and creates a policy-map.
Step 4	**class** {*class-name*| **class-default**] **Example:** `Router(config-pmap)# class class-default`	Enters policy-map class configuration mode. Specifies the name of the class whose policy you want to create or change or specifies the default class (commonly known as the class-default class) before you configure its policy. Repeat this command as many times as necessary to specify the child or parent classes that you are creating or modifying: • *class name* --Name of the class to be configured or whose policy is to be modified. The class name is used for both the class map and to configure a policy for the class in the policy-map.

	Command or Action	Purpose
		• **class-default** --Specifies the default class so that you can configure or modify its policy.
Step 5	**set atm-clp** **Example:** `Router(config-pmap-c)# set atm-clp`	Configures marking of the ATM CLP bit for all traffic matching this class.
Step 6	**end** **Example:** `Router(config-pmap-c)# end`	(Optional) Returns to privileged EXEC mode.

Example

The following example shows you how to set the ATM CLP using explicit marking:

```
policy-map egress_atm_clp_policer
 class prec0
  police cir 5000000
 class class-default
  set atm-clp
```

Configuration Examples for Set ATM CLP Bit Using Policer

Example Marking the ATM CLP by Policer Action Matching a Class

This example shows how to do the following:

- Define traffic classes.
- Configure a two-layer policy-map.
- Apply the policy-map to PPPoA sessions.

This policy conditionally marks the ATM CLP bit on the traffic in the matching low_interest class once traffic on the class exceeds a given rate.

```
class-map voice
  match precedence 4
!
class-map web
  match precedence 3
!
class low_interest
  match precedence 1 0
!
policy-map child
```

```
 child class voice
  police cir 256000
  priority level 1
 class web
  bandwidth remaining ratio 10
 class low_interest
  police cir 1000000 conform-action transmit exceed-action set-clp-transmit
 class class-default
  bandwidth remaining ratio 1
!
policy-map parent
 class class-default
  shape average 15000000
  service-policy child
```

Policy-maps attached to virtual templates are cloned and used to create a virtual access interface for each PPPoA session:

```
interface Virtual-Template1
 ip unnumbered Loopback1
 load-interval 30
 peer default ip address pool POOL1
 ppp authentication chap ppp
 ipcp address required
 service-policy output parent
```

Example Marking the ATM CLP by Policer Action Policed Threshold

This example shows how to do the following:

- Define traffic classes.
- Configure a two-layer policy-map.
- Apply the policy-map to PPPoA sessions.

This policy marks all non-essential traffic with the ATM CLP bit so that it is eligible for dropping if the ATM network becomes congested.

```
class-map video
  match precedence 5
!
class-map voice
  match precedence 4
!
class-map web
  match precedence 3
!
policy-map child
 child class voice
  police cir 256000
  priority level 1
 class video
  police cir 4000000
  priority level 2
 class web
  set atm-clp
  bandwidth remaining ratio 10
 class class-default
  bandwidth remaining ratio 1
```

```
  set atm-clp
!
interface Virtual-Template1
 ip unnumbered Loopback1
 load-interval 30
 peer default ip address pool POOL1
 ppp authentication chap ppp
 ipcp address required
 service-policy output parent
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Quality of Service commands	*Cisco IOS Quality of Service Command Reference*

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	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 Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

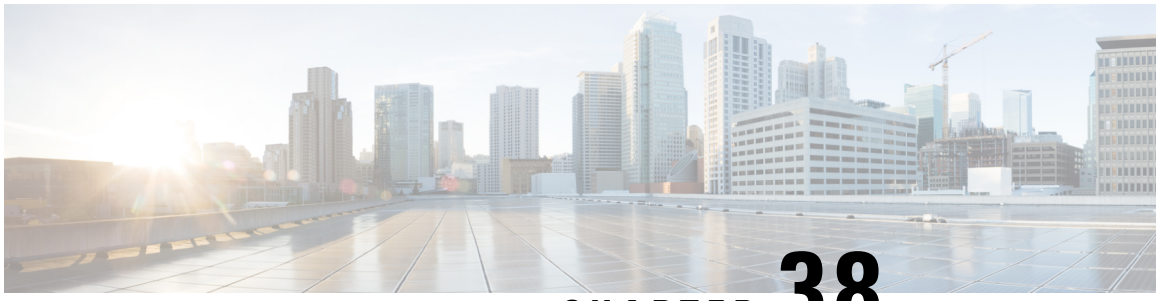
Feature Information for Set ATM CLP Bit Using Policer

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 42: Feature Information for Set ATM CLP Bit Using Policer

Feature Name	Releases	Feature Information
Set ATM CLP Bit Using Policer	Cisco IOS Release XE 3.3S Cisco IOS Release XE 3.14S	The Set ATM CLP Bit Using Policer feature allows you to police and then mark outbound PPPoA traffic. In Cisco IOS Release XE 3.14S, support for this feature was added on the Cisco 4451-X Integrated Services Router. The following commands were introduced or modified: **set atm-clp**and **police**.



CHAPTER 38

EVC Quality of Service

This document contains information about how to enable quality of service (QoS) features (such as traffic classification and traffic policing) for use on an Ethernet virtual circuit (EVC).

An EVC as defined by the Metro Ethernet Forum is a port-level point-to-point or multipoint-to-multipoint circuit. It is an end-to-end representation of a single instance of a service being offered by a provider to a customer. It embodies the different parameters on which the service is being offered.

Information About Quality of Service on an EVC

EVC Quality of Service and the MQC

QoS functionality is typically applied using traffic classes, class maps, and policy-maps. For example, you can specify that traffic belonging to a particular class be grouped into specific categories, and receive a specific QoS treatment (such as classification or policing). The QoS treatment the traffic is to receive is specified in a policy-map and the policy-map is attached to an interface. The mechanism used for applying QoS in this manner is the modular QoS CLI (MQC.)

The policy-map can be attached to an interface in either the incoming (ingress) or outgoing (egress) direction with the **service-policy** command.

The MQC structure allows you to define a traffic class, create a traffic policy, and attach the traffic policy to an interface (in this case, an EVC).

The MQC structure consists of the following three high-level steps.

1. Define a traffic class by using the **class-map**command. A traffic class is used to classify traffic.
2. Create a traffic policy by using the **policy-map** command. (The terms *traffic policy* and *policy-map* are often synonymous.) A traffic policy (policy-map) contains a traffic class and one or more QoS features that will be applied to the traffic class. The QoS features in the traffic policy determine how to treat the classified traffic.

3. Attach the traffic policy (policy-map) to the interface by using the **service-policy** command.

Note For more information about the MQC, including information about hierarchical policy-maps and class maps, see the "Applying QoS Features Using the MQC" module.

QoS-Aware Ethernet Flow Point (EFP)

As described in the EVC Quality of Service and the MQC, on page 515 , the MQC is used to apply one or more QoS features to network traffic. The last step in using the MQC is to attach the traffic policy (policy-map) to an interface (in this case, an EVC) by using the **service-policy** command.

With the EVC Quality of Service feature, the **service-policy** command can be used to attach the policy-map to an Ethernet Flow Point (EFP) in either the incoming (ingress) *or* outgoing (egress) direction of an EVC. This way, the EFP is considered to be "QoS-aware."

QoS Functionality and EVCs

The specific QoS functionality includes the following:

- Packet classification (for example, based on differentiated services code point (DSCP) value and QoS group identifier)
- Packet marking (for example, based on Class of Service (CoS) value)
- Traffic policing (two- and three-color and multiple actions)
- Bandwidth sharing
- Priority queueing (in the outbound direction on the EVC only)
- Weighted Random Early Detection (WRED)

The QoS functionality is enabled by using the appropriate commands listed in the following sections.

match Commands Supported by EVC QoS for Classifying Traffic

The table below lists *some* of the available **match** commands that can be used when classifying traffic on an EVC. The available **match** commands vary by Cisco IOS XE release. For more information about the commands and command syntax, see the Cisco IOS Quality of Service Solutions Command Reference.

Table 43: match Commands That Can Be Used with the MQC

Command	Purpose
match access-group	Configures the match criteria for a class map on the basis of the specified access control list (ACL).
match any	Configures the match criteria for all packets.
match cos	Matches a packet based on a Layer 2 CoS marking.

Command	Purpose
match cos inner	Matches the inner CoS of QinQ packets on a Layer 2 CoS marking.
match [**ip**] **dscp**	Identifies a specific IP DSCP value as a match criterion. Up to eight DSCP values can be included in one match statement.
match not	Specifies the single match criterion value to use as an unsuccessful match criterion. **Note** The **match not** command, rather than identifying the specific match parameter to use as a match criterion, is used to specify a match criterion that prevents a packet from being classified as a member of the class. For instance, if the **match not qos-group 6**command is issued while you configure the traffic class, QoS group 6 becomes the only QoS group value that is not considered a successful match criterion. All other QoS group values would be successful match criteria.
match [**ip**] **precedence**	Identifies IP precedence values as match criteria.
match qos-group	Identifies a specific QoS group value as a match criterion.
match source-address mac	Uses the source MAC address as a match criterion. **Note** Classifying traffic using the **match source-address mac**command is supported in the input direction only.
match vlan (QoS)	Matches and classifies traffic on the basis of the VLAN identification number.
match vlan inner	Configures a class map to match the innermost VLAN ID in an 802.1q tagged frame.

Multiple match Commands in One Traffic Class

If the traffic class contains more than one **match** command, you need to specify how to evaluate the **match** commands. You specify this by using either the **match-any** or **match-all**keyword of the **class-map** command. Note the following points about the **match-any** and **match-all** keywords:

- If you specify the **match-any**keyword, the traffic being evaluated by the traffic class must match *one* of the specified criteria.
- If you specify the **match-all** keyword, the traffic being evaluated by the traffic class must match *all* of the specified criteria.
- If you do not specify either keyword, the traffic being evaluated by the traffic class must match *all* of the specified criteria (that is, the behavior of the **match-all** keyword is used).

Commands Used to Enable QoS Features on the EVC

The commands used to enable QoS features vary by Cisco IOS XE release. The table below lists *some* of the available commands and the QoS features that they enable. For complete command syntax, see the Cisco IOS Quality of Service Solutions Command Reference.

For more information about a specific QoS feature that you want to enable, see the appropriate module of the Cisco IOS Quality of Service Solutions Configuration Guide.

Table 44: Commands Used to Enable QoS Features

Command	Purpose
bandwidth	Configures a minimum bandwidth guarantee for a class.
bandwidth remaining	Configures an excess weight for a class.
drop	Discards the packets in the specified traffic class.
fair-queue	Enables the flow-based queueing feature within a traffic class.
police	Configures traffic policing. Allows specifying of multiple policing actions.
police (percent)	Configures traffic policing on the basis of a percentage of bandwidth available on an interface.
police (two rates)	Configures traffic policing using two rates, the committed information rate (CIR) and the peak information rate (PIR).
priority	Gives priority to a class of traffic belonging to a policy-map.
queue-limit	Specifies or modifies the maximum number of packets the queue can hold for a class configured in a policy-map.
random-detect	Enables Weighted Random Early Detection (WRED).
random-detect cos-based	Enables Weighted random early detection (WRED) on the basis of the class of service (CoS) value of a packet.
random-detect dscp-based	Specifies that Weighted random early detection (WRED) is to use the differentiated services code point (DSCP) value when it calculates the drop probability for a packet.
random-detect discard-class	Configures the WRED parameters for a discard-class value for a class in a policy-map.
random-detect discard-class-based	Configures WRED on the basis of the discard class value of a packet.
random-detect exponential-weighting-constant	Configures the exponential weight factor for the average queue size calculation for the queue reserved for a class.
random-detect precedence	Configure the WRED parameters for a particular IP Precedence for a class policy in a policy-map.
service-policy	Specifies the name of a traffic policy used as a matching criterion (for nesting traffic policies [hierarchical traffic policies] within one another).
set cos	Sets the Layer 2 CoS value of an outgoing packet.

Command	Purpose
set cos-inner	Marks the inner class of service field in a bridged frame.
set discard-class	Marks a packet with a discard-class value.
set [**ip**] **dscp**	Marks a packet by setting the DSCP value in the type of service (ToS) byte.
set mpls experimental	Designates the value to which the Multiprotocol Label Switching (MPLS) bits are set if the packets match the specified policy-map.
set precedence	Sets the precedence value in the packet header.
set qos-group	Sets a QoS group identifier (ID) that can be used later to classify packets.
shape	Shapes traffic to the indicated bit rate according to the algorithm specified.

input and output Keywords of the service-policy Command

As a general rule, the QoS features configured in the traffic policy can be applied to packets entering the interface or to packets leaving the interface. Therefore, when you use the **service-policy** command, you need to specify the direction of the traffic policy by using the **input** or **output** keyword.

For instance, the **service-policy output policy-map1** command would apply the QoS features in the traffic policy to the interface in the output direction. All packets leaving the interface (output) are evaluated according to the criteria specified in the traffic policy named policy-map1.

Note For Cisco releases, queueing mechanisms are not supported in the input direction. Nonqueueing mechanisms (such as traffic policing and traffic marking) are supported in the input direction. Also, classifying traffic on the basis of the source MAC address (using the **match source-address mac** command) is supported in the input direction only.

How to Configure a Quality of Service Feature on an EVC

Creating a Traffic Class for Use on the EVC

To create a traffic class, use the **class-map** command to specify the traffic class name. Then use one or more **match** commands to specify the appropriate match criteria. Packets matching the criteria that you specify are placed in the traffic class.

To create the traffic class for use on the EVC, complete the following steps.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **class-map** [**match-all** | **match-any**] *class-name*
4. **match cos** *cos-number*
5. Enter additional **match** commands, if applicable; otherwise, proceed with the next step.
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** [**match-all**	**match-any**] *class-name* **Example:** `Router(config)# class-map match-any class1`	Creates a class map and enters class-map configuration mode. • The class map is used for matching packets to the specified class. **Note** The **match-all** keyword specifies that all match criteria must be met. The **match-any** keyword specifies that one of the match criteria must be met. Use these keywords only if you will be specifying more than one **match** command.
Step 4	**match cos** *cos-number* **Example:** `Router(config-cmap)# match cos 2`	Matches a packet on the basis of a Layer 2 CoS number. **Note** The **match cos** command is an example of a **match** command you can use.
Step 5	Enter additional **match** commands, if applicable; otherwise, proceed with the next step.	--
Step 6	**end** **Example:** `Router(config-cmap)# end`	(Optional) Exits class map configuration mode and returns to privileged EXEC mode.

Creating a Policy-Map for Use on the EVC

To create a traffic policy (or policy-map) for use on the EVC, complete the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name*| **class-default**}
5. **police** *bps* [*burst-normal*] [*burst-max*] [**conform-action** *action*] [**exceed-action** *action*] [**violate-action** *action*]
6. Enter the commands for any additional QoS feature that you want to enable, if applicable; otherwise, proceed to the next step.
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**policy-map** *policy-map-name*
	Example:
	`Router(config)#`
	`policy-map policy1`
		Creates or specifies the name of the traffic policy and enters QoS policy-map configuration mode.

Step 4	**class** {*class-name*| **class-default**}
	Example:
	`Router(config-pmap)#`
	`class class1`
		Specifies the name of a class and enters QoS policy-map class configuration mode.
		Note This step associates the traffic class with the traffic policy.

Step 5	**police** *bps* [*burst-normal*] [*burst-max*] [**conform-action** *action*] [**exceed-action** *action*] [**violate-action** *action*] **Example:** `Router(config-pmap-c)# police 3000`	(Optional) Configures traffic policing. **Note** The **police**command is an example of a command that you can use in a policy-map to enable a QoS feature.
Step 6	Enter the commands for any additional QoS feature that you want to enable, if applicable; otherwise, proceed to the next step.	--

	Command or Action	Purpose
Step 7	**end** **Example:** `Router(config-pmap-c)# end`	(Optional) Exits QoS policy-map class configuration mode and returns to privileged EXEC mode.

Configuring the EVC and Attaching a Traffic Policy to the EVC

The traffic policy (policy-map) applies the enabled QoS feature to the traffic class once you attach the policy-map to the EVC.

To configure the EVC and attach a traffic policy to the EVC, complete the following steps.

Note One of the commands used to attach the traffic policy to the EVC is the **service-policy** command. When you use this command, you must specify either the **input** or **output** keyword along with the policy-map name. The policy-map contains the QoS feature you want to use. Certain QoS features can only be used in either the input or output direction. For more information about these keywords and the QoS features supported, see the input and output Keywords of the service-policy Command, on page 298. Also, if you attach a traffic policy to an interface containing multiple EVCs, the traffic policy will be attached to *all* of the EVCs on the interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-type interface-number*
4. **service instance** *id* **ethernet** [*evc-name*]
5. **encapsulation dot1q** *vlan-id* [,*vlan-id*[-*vlan-id*]] [**native**]
6. **rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric**
7. **bridge domain** *domain-number*
8. **service-policy** {**input** | **output**} *policy-map-name*
9. **end**
10. **show policy-map interface** *type number* **service instance** *service-instance-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**interface** *interface-type interface-number* **Example:** `Router(config)# interface gigabitethernet 0/0/1`	Configures an interface type and enters interface configuration mode. • Enter the interface type and interface number.
Step 4	**service instance** *id* **ethernet** [*evc-name*] **Example:** `Router(config-if)# service instance 333 ethernet evc1`	Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode. • Enter the service instance identification number and, if applicable, the EVC name (optional).
Step 5	**encapsulation dot1q** *vlan-id* [,*vlan-id*[-*vlan-id*]] [**native**] **Example:** `Router(config-if-srv)# encapsulation dot1q 10`	Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance.
Step 6	**rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric** **Example:** `Router(config-if-srv)# rewrite ingress tag translate 1-to-1 dot1q 300 symmetric`	Specifies the encapsulation adjustment to be performed on a frame ingressing a service instance.
Step 7	**bridge domain** *domain-number* **Example:** `Router(config-if-srv)# bridge domain 1`	Configures a bridge domain. • Enter the bridge domain number.

Step 8	**service-policy** {**input** | **output**} *policy-map-name*
	Example:
	`Router(config-if-srv)#`
	`service-policy input policy1`
		Attaches a policy-map to an interface.
		• Enter either the **input** or **output** keyword and the policy-map name.

Step 9	**end** **Example:** `Router(config-if-srv)# end`	(Optional) Returns to privileged EXEC mode.
Step 10	**show policy-map interface** *type number* **service instance** *service-instance-number* **Example:** `Router# show policy-map interface gigabitethernet 1/0/0 service instance 30`	(Optional) Displays the statistics and the configurations of the input and output policies that are attached to an interface. • Enter the interface type, interface number, and service instance number.

Configuration Examples for EVC Quality of Service

Example Creating a Traffic Class for Use on the EVC

In this example, traffic with a CoS value of 2 is placed in the traffic class called class1:

```
Router> enable

Router# configure terminal

Router(config)# class-map match-any class1

Router(config-cmap)# match cos 2

Router(config-cmap)# end
```

Example Creating a Policy-Map for Use on the EVC

In this example, traffic policing has been configured in the policy-map called policy1. Traffic policing is the QoS feature applied to the traffic in class1:

```
Router> enable

Router# configure terminal

Router(config)#
 policy-map policy1

Router(config-pmap)#
 class class1

Router(config-pmap-c)# police 3000

Router(config-pmap-c)# end
```

Example Configuring the EVC and Attaching a Traffic Policy to the EVC

In this example, an EVC has been configured and a traffic policy called policy1 has been attached to the EVC:

```
Router> enable
```

```
Router# configure terminal
Router(config)# interface gigabitethernet 0/0/1
Router(config-if)# service instance 333 ethernet evc1
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# rewrite ingress tag translate 1-to-1 dot1q 300 symmetric
Router(config-if-srv)# bridge domain 1
Router(config-if-srv)# service-policy input policy1
Router(config-if-srv)# end
```

Example Verifying the Traffic Class and Traffic Policy Information for the EVC

The following is sample output of the **show policy-map interface service instance**command. It displays the QoS features configured for and attached to the EFP on the GigabitEthernet interface 1/1/7.

```
Router# show policy-map interface gigabitethernet 1/1/7 service instance 10
 GigabitEthernet1/1/7: EFP 10
  Service-policy input: multiaction
    Class-map: c1 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: ip precedence 3
      police:
          cir 300000 bps, bc 2000 bytes
        conformed 0 packets, 0 bytes; actions:
          set-prec-transmit 7
          set-qos-transmit 10
        exceeded 0 packets, 0 bytes; actions:
          drop
        conformed 0000 bps, exceed 0000 bps
    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Packet classification	"Classifying Network Traffic" module
Selective Packet Discard	“IPv6 Selective Packet Discard” module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

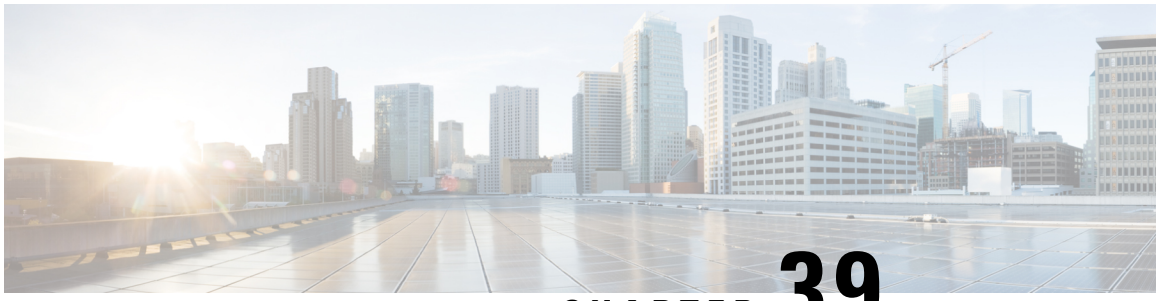
Feature Information for Configuring EVC Quality of Service

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 45: Feature Information for EVC Quality of Service

Feature Name	Releases	Feature Information
EVC Quality of Service	Cisco IOS XE Release 3.3 Cisco IOS Release 15.5(2)T	This document contains information about how to enable quality of service (QoS) features (such as traffic classification and traffic policing) for use on an Ethernet virtual circuit (EVC). The EVC Quality of Service feature was introduced on the Cisco ASR 1000 Series Aggregation Services Router. The following commands were introduced or modified: **service-policy**, **show policy-map interface service instance**.



CHAPTER 39

Quality of Service for Etherchannel Interfaces

Quality of Service (QoS) is supported on Ethernet Channel (Etherchannel) interfaces on Cisco ASR 1000 Series Routers. The QoS functionality has evolved over several Cisco IOS XE releases and has different capabilities based on software level, Etherchannel configuration, and configured Modular QoS CLI (MQC) features.

Etherchannel with QoS Feature Evolution

An Etherchannel is a port-channel architecture that allows grouping of several physical links to create one logical Ethernet link for the purpose of providing fault tolerance, and high-speed links between switches, routers, and servers. An Etherchannel can be created from between two and eight active Fast, Gigabit, or 10-Gigabit Ethernet ports, with an additional one to eight inactive (failover) ports, which become active as the other active ports fail.

QoS for Etherchannel interfaces has evolved over several Cisco IOS XE releases. It is important to understand what level of support is allowed for your current level of Cisco IOS XE software and underlying Etherchannel configuration. Various combinations of QoS are supported based on how Etherchannel is configured. There are three different modes in which Etherchannel can be configured:

- Etherchannel VLAN-based load balancing via port-channel subinterface encapsulation CLI
- Etherchannel Active/Standby with LACP (no Etherchannel load balancing)
- Etherchannel with LACP with load balancing

Each of these models has specific restrictions regarding which levels of Cisco IOS XE software include support and the possible QoS configurations with each.

The following summarizes the various Etherchannel and QoS configuration combinations that are supported. Example configurations will be provided later in this document. Unless specifically mentioned together, the combination of service policies in different logical and physical interfaces for a given Etherchannel configuration is not supported.

Etherchannel VLAN-Based Load Balancing via Port-Channel Subinterface Encapsulation CLI

Supported in Cisco IOS XE Release 2.1 or later:

- Egress MQC Queuing Configuration on Port-Channel Subinterface
- Egress MQC Queuing Configuration on Port-Channel Member Link
- QoS Policies Aggregation—Egress MQC Queuing at Subinterface
- Ingress Policing and Marking on Port-Channel Subinterface
- Egress Policing and Marking on Port-Channel Member Link

Supported in Cisco IOS XE Release 2.6 or later:

- QoS Policies Aggregation—MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface

Etherchannel Active/Standby with LACP (No Etherchannel Load Balancing)

Supported in Cisco IOS XE 2.4 or later:

- Egress MQC Queuing on Port-Channel Member Link—No Etherchannel Load Balancing

Etherchannel with LACP and Load Balancing

Supported in Cisco IOS XE 2.5 or later:

- Egress MQC Queuing Configuration on Port-Channel Member Link—Etherchannel Load Balancing

Supported in Cisco IOS XE 3.12 or later:

- General MQC QoS support on Port-channel main-interface

  We recommend that as a best practice for QoS, that you use port-channel aggregation—see the "Aggregate EtherChannel Quality of Service" chapter.

Supported in Cisco IOS XE 3.16.3 or later and in Cisco IOS XE Fuji 16.3 or later:

- General MQC QoS support on Port-channel sub-interface

  We recommend that as a best practice for QoS, that you use port-channel aggregation—see the "Aggregate EtherChannel Quality of Service" chapter.

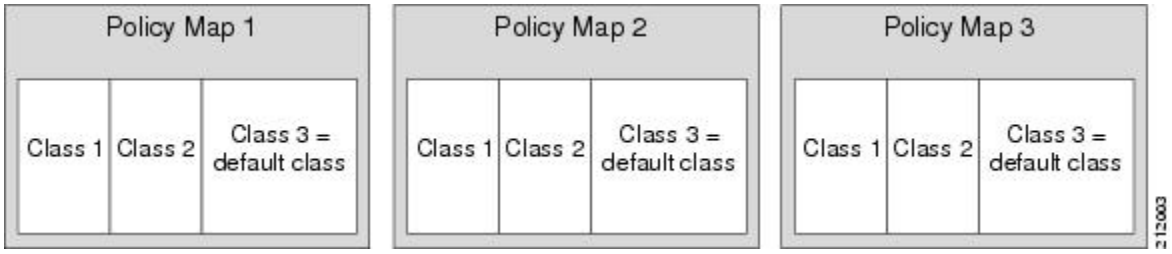
Understanding Fragments in Class Definition Statements

The QoS Policies Aggregation feature introduces the idea of fragments in class definition statements. A default traffic class definition statement can be marked as a fragment within a policy-map. Other policy-maps on the

same interface can also define their default traffic class statements as fragments, if desired. A separate policy-map can then be created with a service fragment class definition statement that will be used to apply QoS to all of the fragments as a single group.

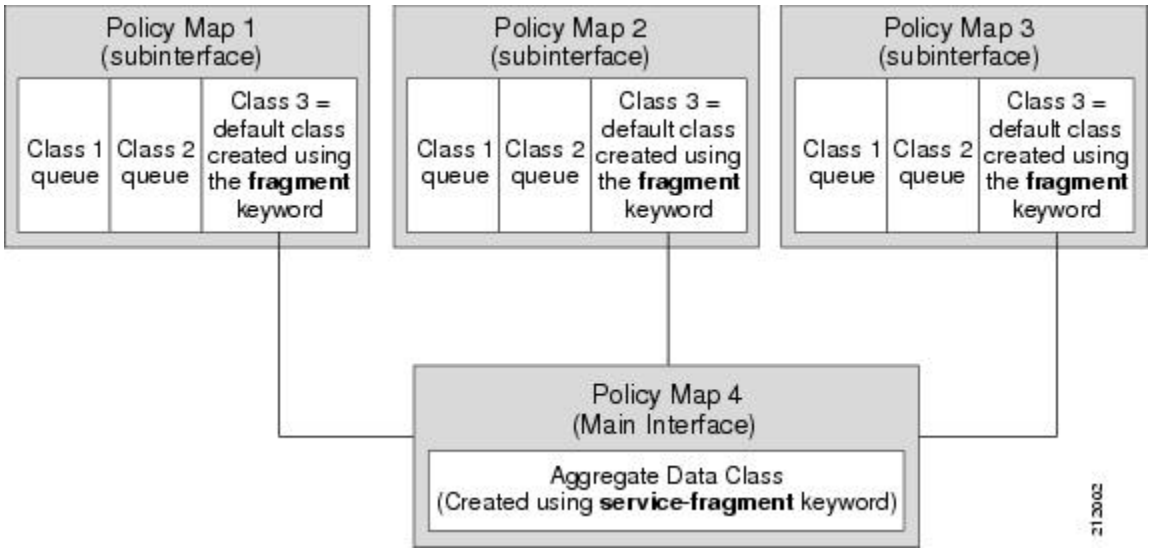
The figure below provides an example of one physical interface with three attached policy-maps that is not using fragments. Note that each policy-map has a default traffic class that can classify traffic only for the default traffic within its own policy-map.

Figure 73: Physical Interface with Policy-Maps—Not Using Fragments



The figure below shows the same configuration configured with fragments, and adds a fourth policy-map with a class definition statement that classifies the fragments collectively. The default traffic classes are now classified as one service fragment group rather than three separate default traffic classes within the individual policy-maps.

Figure 74: Physical Interface with Policy-Maps—Using Fragments



Fragments for Gigabit Etherchannel Bundles

When fragments are configured for Gigabit Etherchannel bundles, the policy-maps that have a default traffic class configured using the **fragment** keyword are attached to the member subinterface links, and the policy-maps that have a traffic class configured with the **service-fragment** keyword to collectively classify the fragments is attached to the physical interface.

All port-channel subinterfaces configured with fragments that are currently active on a given port-channel member link will use the aggregate service fragment class on that member link. If a member link goes down,

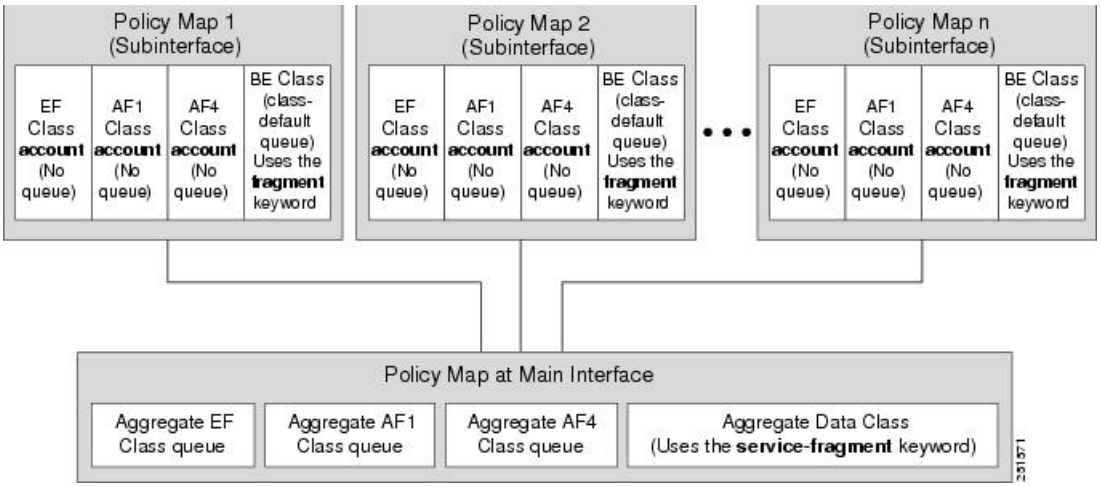
the port-channel subinterfaces that must switch to the secondary member link will then use the aggregate service fragment on the new interface.

QoS: Policies Aggregation MQC

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature extends the previous support of aggregation of class-default traffic using the **fragment** and **service-fragment** configurations, to other user-defined traffic classes in a subinterface policy-map, such as DSCP-based traffic classes, that are aggregated at the main-interface policy-map as shown in the figure below.

When no queueing is configured on a traffic class in the subinterface policy-map, the **account** command can be used to track queueing drops that occur at the aggregate level for these classes, and can be displayed using the **show policy-map interface** command.

Figure 75: Policy-Map Overview for the MQC Support for Multiple Queue Aggregation at Main Interface Feature



Differences Between the Original Feature and the MQC Support for Multiple Queue AggregationDifferences Between Policy Aggregation—Egress MQC Queuing at Subinterface and the MQC Support for Multiple Queue Aggregation at Main Interface

Although some of the configuration between the “Policy Aggregation – Egress MQC Queuing at Subinterface” scenario and the “MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface” scenario appear similar, there are some important differences in the queuing behavior and the internal data handling. See the figure in the “Understanding the QoS: Policies Aggregation MQC” section.

For example, both configurations share and require the use of the **fragment** keyword for the **class class-default** command in the subscriber policy-map, as well as configuration of the **service-fragment** keyword for a user-defined class in the main-interface policy-map to achieve common policy treatment for aggregate traffic. However, the use of this configuration results in different behavior between the original and enhanced QoS policies aggregation implementation:

- In the original implementation using the fragment and service-fragment architecture, all default class traffic and any traffic for classes without defined queueing features at the subinterface goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy-map. Subinterface traffic aggregation (for example, from multiple subscribers on the same physical interface) ultimately occurs only for a single class, which is the default class.
- In the enhanced implementation of the MQC Support for Multiple Queue Aggregation at Main Interface feature also using the fragment and service-fragment architecture, all default class traffic also goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy-map. However, other classes, such as DSCP-based subscriber traffic classes, are also supported for an aggregate policy. These traffic classes do not support any queues or queueing features other than **account** at the subscriber policy-map. The use of the fragment and service-fragment architecture enables these other subscriber traffic classes (from multiple subscribers on the same physical interface) to achieve common policy treatment for aggregate traffic that is defined for those same classes at the main policy-map.

How to Configure QoS for Etherchannels

Configuring Egress MQC Queuing on Port-Channel Subinterface

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The port-channel subinterface should have been previously configured with the appropriate encapsulation subcommand to match the select primary and secondary physical interfaces on the Etherchannel. Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command, or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-number* **.** *subinterface-number*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**interface port-channel** *port-channel-number* **.** *subinterface-number* **Example:** `Device(config)# interface port-channel 1.200`	Specifies the port-channel subinterface that receives the service policy configuration.

Step 4	**service-policy output** *policy-map-name*
	Example:
	`Device(config-subif)#`
	`service-policy output WAN-GEC-sub-Out`
		Specifies the name of the service policy that is applied to output traffic.

Step 5	**end** **Example:** `Device(config-subif)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Egress MQC queuing on Port-Channel Member Links

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map that uses queuing features should be configured using previously defined class maps. The Etherchannel member link interface should already be configured to be part of the channel group (Etherchannel group). No policy-maps that contain queuing commands should be configured on any port-channel subinterfaces. Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command, or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service policy configuration.

Step 4	**service-policy output** *policy-map-name*
	Example:
	`Device(config-if)#`
	`service-policy output WAN-GEC-sub-Out`
		Specifies the name of the service policy that is applied to output traffic for this physical interface that is part of the Etherchannel.

Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring QoS Policies Aggregation—Egress MQC Queuing at Subinterface

Before you begin

Default class traffic from multiple Port-channel subinterfaces can be aggregated into a common policy-map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and the **service-fragment** configuration at the main interface class. Queuing occurs at the subinterface for other traffic classes that are defined with queuing features in the subinterface policy-map.

This feature is configured using Modular QoS CLI (MQC). It is most useful in QoS configurations where several policy-maps attached to the same physical interface want aggregated treatment of multiple default traffic classes from multiple port-channel sub-interfaces. Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command, or the port-channel main-interface must have the **load-balancing vlan** command. It is assumed that these commands have already been executed.

Note This feature is supported when policy-maps are attached to multiple port-channel subinterfaces and the port-channel member link interfaces. This feature cannot be used to collectively classify default traffic classes of policy-maps on different physical interfaces. It can collectively classify all traffic directed toward a given port-channel member link when designated by the **primary** or **secondary** directives on the subinterface **encapsulation** command. All subinterface traffic classes should have queues. However, when a traffic class in the subinterface policy-map is not configured with any queuing feature (commands such as **priority**, **shape**, **bandwidth**, **queue-limit**, **fair-queue**, or **random-detect**), the traffic is assigned to the class-default queue. No classification occurs or is supported at the main interface policy-map for any subinterface traffic classes that do not use the **fragment** and **service-fragment** configuration.

A multistep process is involved with the complete configuration of the QoS Policies Aggregation feature. The following sections detail those steps.

Note the following about attaching and removing a policy-map:

- To configure QoS Policies Aggregation, you must attach the policy-map that contains the **service-fragment** keyword to the main interface first, and then you must attach the policy-map that contains the **fragment** keyword to the subinterface.
- To disable QoS Policies Aggregation, you must remove the policy-map that contains the fragment keyword from the subinterface first, and then you must remove the policy-map that contains the service-fragment keyword from the main interface.

Configuring a Fragment Traffic Class in a Policy-Map

Before you begin

This procedure shows only how to configure the default traffic class as a fragment within a policy-map. It does not include steps on configuring other classes within the policy-map, or other policy-maps on the device.

Example

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a fragment named BestEffort is created in policy-map subscriber1 and policy-map subscriber 2. In this example, queuing features for other traffic classes are supported at the subinterface policy-map.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example also shows how to configure a fragment named BestEffort for the default class in a policy-map on a subinterface using the QoS Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface implementation. In this example, notice that queuing features are not supported for the other classes in the policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

After configuring default class statements as fragments in multiple subinterface policy-maps, a separate policy-map with a class statement using the service-fragment keyword must be configured to apply QoS to the class stratements configured as fragments.

What to Do Next

After configuring multiple default class statements as fragments in a policy-map, a separate policy-map with a class statement using the **service-fragment** keyword must be configured to apply QoS to the class statements configured as fragments.

This process is documented in the “Configuring a Service Fragment Traffic Class” section.

Configuring a Service Fragment Traffic Class

Before you begin

This task describes how to configure a service fragment traffic class statement within a policy-map. A service fragment traffic class is used to apply QoS to a collection of default class statements that have been configured previously in other policy-maps as fragments.

This procedure assumes that fragment default traffic classes were already created. The procedure for creating fragment default traffic classes is documented in the “Configuring a Fragment Traffic Class in a Policy-Map” section.

Like any policy-map, the configuration does not manage network traffic until it has been attached to an interface. This procedure does not cover the process of attaching a policy-map to an interface.

Note A service fragment can be used to collectively classify fragments only from the same physical interface. Fragments from different interfaces cannot be classified using the same service fragment.

Only queueing features are allowed in classes where the **service-fragment** keyword is entered, and at least one queueing feature must be entered in classes when the **service-fragment** keyword is used.

A policy-map with a class using the **service-fragment** keyword can be applied only to traffic leaving the interface (policy-maps attached to interfaces using the **service-policy output** command).

A class configured using the **service-fragment** keyword cannot be removed when it is being used to collectively apply QoS to fragments that are still configured on the interface. If you wish to remove a class configured using the **service-fragment** keyword, remove the fragment traffic classes before removing the service fragment.

The **service-fragment** keyword cannot be entered in a child policy-map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name* **service-fragment** *fragment-class-name*
5. **shape average percent** *percent*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map BestEffortFragments`	Specifies the name of the traffic policy to configure and enters policy-map configuration mode.
Step 4	**class** *class-name* **service-fragment** *fragment-class-name* **Example:** `Device(config-pmap)# class data service-fragment BestEffort`	Specifies a class of traffic that is the composite of all fragments matching the *fragment-class-name*. The *fragment-class-name* when defining the fragments in other policy-maps must match the *fragment-class-name* in this command line to properly configure the service fragment class.

	Command or Action	Purpose
Step 5	**shape average percent** *percent* **Example:** `Device(config-pmap-c)# shape average percent 50`	Enters a QoS configuration command. Only queueing features are supported in default traffic classes configured as fragments. The queueing features that are supported are **bandwidth**, **shape**, and **random-detect exponential-weighting-constant**. Multiple QoS queueing commands can be entered.
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	Exits policy-map class configuration mode and returns to privileged EXEC mode.

Examples

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a policy-map is created to apply QoS to all fragments named BestEffort.

```
policy-map main-interface
 class data service-fragment BestEffort
  shape average 400000000
```

In the following example, two fragments are created and then classified collectively using a service fragment.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example shows the creation of two fragments called BestEffort in the subinterface policy-maps, followed by a sample configuration for the **service-fragment** called BestEffort to aggregate the queues at the main interface policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber2
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map main-interface
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
```

Troubleshooting Tips

Ensure that all class statements that should be part of the same service fragment share the same *fragment-class-name*.

What to Do Next

Attach the service fragment traffic classes to the main physical interfaces.

Attach the fragment traffic classes to the member-link subinterfaces.

Configuring Service Fragments on a Physical Interface Supporting a Gigabit Etherchannel Bundle

Before you begin

This procedure assumes that a service fragment traffic class has already been created. A service fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy-Map" section. The procedure for creating a service fragment traffic classes is documented in the "Configuring a Service Fragment Traffic Class" section.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions document only the procedure for attaching a policy-map that already has a fragment traffic class to a member link subinterface.

Note For proper behavior, when a port-channel member link goes down, all member links should have the same policy-map applied.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *service-fragment-class-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service-policy configuration.

	Command or Action	Purpose
Step 4	**service-policy output** *service-fragment-class-name* **Example:** `Device(config-if)# service-policy output aggregate-member-link`	Attaches a service policy that contains a service fragment default traffic class to the physical Gigabit Ethernet interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Examples

In the following example, the policy-map aggregate-member-link is attached to the physical interface.

```
interface GigabitEthernet1/1/1
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 service-policy output aggregate-member-link
```

What to do next

Ensure that the fragment class name is consistent across service-fragment and fragment class definitions. Continue to the "Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces" section.

Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces

Before you begin

This procedure assumes that a service fragment traffic class has already been created. A service fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy-Map" section. The procedure for creating a service fragment traffic class is documented in the "Configuring a Service Fragment Traffic Class" section.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions only document the procedure for attaching a policy-map that already has a fragment traffic class to a member link subinterface.

Fragments cannot be used for traffic on two or more physical interfaces.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-interface-number* **.** *port-channel-subinterface-number*
4. **service-policy output** *fragment-class-name*

5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-interface-number* **.** *port-channel-subinterface-number* **Example:** `Device(config)# interface port-channel 1.100`	Enters subinterface configuration mode to configure an Etherchannel member link subinterface.
Step 4	**service-policy output** *fragment-class-name* **Example:** `Device(config-subif)# service-policy output subscriber`	Attaches a service policy that contains a fragment default traffic class to the Etherchannel member link subinterface
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named subscriber has a fragment default traffic class and is attached to the port-channel subinterface of an Etherchannel bundle.

```
interface port-channel 1.100
 service-policy output subscriber
```

Configuring Ingress Policing and Marking on Port-Channel Subinterface

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The Etherchannel member link interface should

already be configured to be part of the channel group (Etherchannel group). Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-number* **.** *port-channel-interface-number* **.** *sub-interface-number*
4. **service-policy input** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-number . port-channel-interface-number . sub-interface-number* **Example:** `Device(config)# interface port-channel 1.100.100`	Enters subinterface configuration mode to configure an Etherchannel member link subinterface.
Step 4	**service-policy input** *policy-map-name* **Example:** `Device(config-subif)# service-policy input sub-intf-input`	Specifies the name of the service policy that is applied to input traffic for the port-channel subinterface previously specified.
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named sub-intf-input is defined and attached to the port-channel subinterface in the input direction.

```
policy-map sub-intf-input
  class voice
    set precedence 5
  class video
    set precedence 6
  class class-default
    set precedence 3
!
interface Port-channel 1.100
 service-policy input sub-intf-input
```

Configuring Egress Policing and Marking on Port-Channel Member Links

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The Etherchannel member link interface should already be configured to be part of the channel group (Etherchannel group). Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-number . port-channel-interface-number . sub-interface-number*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-number . port-channel-interface-number . sub-interface-number* **Example:** `Device(config)# interface port-channel 1.100.100`	Enters subinterface configuration mode to configure an Etherchannel member link subinterface.

	Command or Action	Purpose
Step 4	**service-policy output** *policy-map-name* **Example:** `Device(config-subif)# service-policy output WAN-GEC-member-Out-police`	Specifies the name of the service policy that is applied to output traffic for the Etherchannel member link subinterface specified in the previous step.
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named WAN-GEC-member-Out-police is defined and attached to the port-channel subinterface in the output direction.

```
policy-map WAN-GEC-member-Out-police
  class voice
    set precedence 5
  class video
    set precedence 6
  class class-default
    set precedence 3
!
interface port-channel 1.100
 service-policy output WAN-GEC-member-Out-police
```

Configuring Policies Aggregation—MQC Support for Multiple Queue Aggregation at Main Interface

Before you begin

This feature is configured using the MQC. It is most useful in QoS configurations where several policy-maps attached to the same physical interface want aggregated treatment of multiple user-defined traffic classes from multiple port-channel subinterfaces. Cisco IOS XE Release 2.6 or later software is required. The global configuration must contain the following command: **port-channel load-balancing vlan-manual** or the main interface of the port-channel being configured must have the following command: **port-channel load-balancing vlan**. It is assumed that these commands have already been executed.

This feature is supported when policy-maps are attached to multiple port-channel subinterfaces and the port-channel member link interfaces. This feature cannot be used to collectively classify default traffic classes of policy-maps on different physical interfaces. It can collectively classify all traffic directed towards a given Port-channel member-link when designated by the **primary** or **secondary** directives on the sub-interface **encapsulation** command. The following items describe the behavior and restrictions on configuring this type of QoS Policy Aggregation with Etherchannel:

- Subinterface traffic classes without configured queuing features do not have queues at the subscriber level

- Default class traffic from multiple subinterfaces can be aggregated into a common policy-map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main interface class
- This configuration additionally enables support for other subinterface traffic classes (such as DSCP-based classes) to be aggregated into a common policy-map at the main interface.
- This feature is enabled by using the **fragment** keyword in the subinterface **class-default** class, and **service-fragment** configuration in the main interface class (this also enables aggregation of the default class.
- Queuing features are not configured at the subinterface policy-map for the other traffic classes.
- Queuing occurs at the main interface policy-map for other subinterface traffic classes as an aggregate.
- Optional tracking of statistics is supported using the **account** command for other traffic classes in the subinterface policy-map.

A multistep process is involved with the complete configuration of QoS multiple queue aggregation at a main interface feature, as follows:

1. Configure default class statements as fragments in multiple subinterface policy-maps as described in the "Configuring a Fragment Traffic Class in a Policy-Map" section.
2. Configure a separate policy-map with a class statement using the **service-fragment** keyword in order to apply QoS to the class statements configured as fragments as described in the "Configuring a Service Fragment Traffic Class" section.
3. Configure service fragment traffic classes and attach them to the main physical interfaces as described in the "Configuring Service Fragments on a Physical Interface Supporting a Gigabit Etherchannel Bundle" section.
4. Configure fragment traffic classes and attach them to the member link subinterfaces as described in the "Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces" section.

Configuring MQC Queuing on Port-Channel Member Link—No Etherchannel Load Balancing

Before you begin

Traffic classes must be configured using the **class-map** command. A one or two level hierarchical policy-map should be configured using previously defined class maps.

Cisco IOS XE Release 2.4 or later software is required.

The port-channel main interface should also contain the following commands that create an active/standby scenario. Such a configuration will allow only a single interface to be active and forwarding traffic at any time.

- **interface Port-channel1**
- **lacp fast-switchover**
- **lacp max-bundle 1**

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service policy configuration.

Step 4	**service-policy output** *policy-map-name*
	Example:
	`Device(config-if)#`
	`service-policy output WAN-GEC-member-Out`
		Specifies the name of the service policy that is applied to output traffic.

Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named main-intf is defined and attached to the port-channel member links in the output direction.

```
interface Port-channel 1
 lcap fast-switchover
 lacp max-bundle 1
!
policy-map main-intf
 class voice
  priority
  police cir 10000000
```

```
 class video
  bandwidth remaining ratio 10
 class class-default
  bandwidth remaining ratio 3
!
interface GigabitEthernet0/0/0
 channel-group 1 mode active
 service-policy output main-intf
!
interface GigabitEthernet0/0/1
 channel-group 1 mode active
 service-policy output main-intf
```

Configuring MQC Queuing Configuration on Port-Channel Member Link—Etherchannel Load Balancing

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The port-channel subinterface should have been previously configured with the appropriate encapsulation subcommand to match the select primary and secondary physical interfaces on the Etherchannel. Cisco IOS XE Release 2.5 or later software is required.

The Etherchannel setup may have multiple active interfaces with flow-based load balancing enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service policy configuration.

	Command or Action	Purpose
Step 4	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output WAN-GEC-member-Out`	Specifies the name of the service policy that is applied to output traffic.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named main-intf is defined and attached to the port-channel member links in the output direction.

```
  class voice
   priority
   police cir 10000000
  class video
   bandwidth remaining ratio 10
  class class-default
   bandwidth remaining ratio 3
!
interface GigabitEthernet0/0/0
 channel-group 1 mode active
 service-policy output main-intf
!
interface GigabitEthernet0/0/1
 channel-group 1 mode active
 service-policy output main-intf
```

Configuration Examples for QoS for Etherchannels

Example: Configuring QoS Policies Aggregation—Egress MQC Queuing at Subinterface

```
port-channel load-balancing vlan-manual
!
class-map match-all BestEffort
!
class-map match-all video
 match precedence 4
!
class-map match-all voice
 match precedence 5
!
```

```
policy-map subscriber
 class voice
  priority level 1
 class video
  priority level 2
 class class-default fragment BE
  shape average 100000000
  bandwidth remaining ratios 80

policy-map aggregate-member-link
 class BestEffort service-fragment BE
 shape average 100000000
!
interface Port-channel1
 ip address 209.165.200.225 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 209.165.200.226 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.200
 encapsulation dot1Q 200
 ip address 209.165.200.227 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 209.165.200.228 255.255.255.0
 service-policy output subscriber
!
interface GigabitEthernet1/1/1
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
```

Example: Configuring QoS Policies Aggregation—MQC Support for Multiple Queue Aggregation at Main Interface

```
port-channel load-balancing vlan-manual
!
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
!
policy-map subscriber2
```

```
 class voice
  set cos 2
  account
 class video
  set cos 3
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
!
policy-map main-interface-out
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
!
interface GigabitEthernet1/1/1
 no ip address
 channel-group 1 mode on
 service-policy output main-interface-out
!
interface GigabitEthernet1/1/2
 no ip address
 channel-group 1 mode on
 service-policy output main-interface-out
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 10.0.0.1 255.255.255.0
 service-policy output subscriber1
!
interface Port-channel1.200
 encapsulation dot1Q 200
 ip address 10.0.0.2 255.255.255.0
 service-policy output subscriber2
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 10.0.0.4 255.255.255.0
 service-policy output subscriber2
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular Quality of Service Command-Line Interface	"Applying QoS Features Using the MQC" module
Configuring RADIUS-based policing	*Intelligent Services Gateway Configuration Guide*
CISCO ASR 1000 Series software configuration	*Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Quality of Service for Etherchannel Interfaces

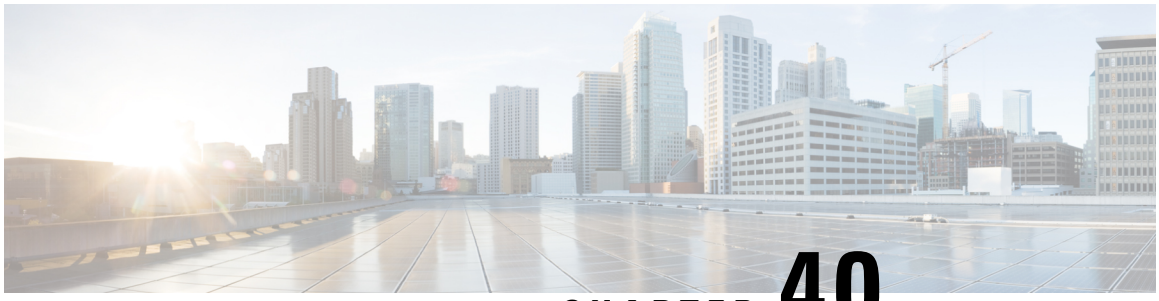
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 46: Feature Information for Quality of Service for Etherchannel Interfaces

Feature Name	Releases	Feature Information
Egress MQC Queuing Configuration on Port-Channel Subinterface	Cisco IOS XE Release 2.1	This feature supports the configuration of Egress MQC queuing on port-channel subinterface. This feature was introduced on Cisco ASR 1000 Series Routers.

Feature Name	Releases	Feature Information
Egress MQC Queuing Configuration on Port-Channel Member Link	Cisco IOS XE Release 2.1	This feature supports the configuration of Egress MQC queuing on port-channel member link. This feature was introduced on Cisco ASR 1000 Series Routers.
QoS Policies Aggregation—Egress MQC Queuing at Subinterface	Cisco IOS XE Release 2.1	This feature supports the configuration of QoS Policies Aggregation - Egress MQC queuing at subinterface. This feature was introduced on Cisco ASR 1000 Series Routers.
Ingress Policing and Marking on Port-Channel Subinterface	Cisco IOS XE Release 2.1	This feature supports the configuration of Ingress Policing and Marking on port-channel subinterface. This feature was introduced on Cisco ASR 1000 Series Routers.
Egress Policing and Marking on Port-Channel Member Link	Cisco IOS XE Release 2.1	This feature supports the configuration of Egress policing and marking on port-channel member link. This feature was introduced on Cisco ASR 1000 Series Routers.
Egress MQC Queuing Configuration on Port-Channel Member Link - No Etherchannel Load Balancing	Cisco IOS XE Release 2.4	This feature supports the configuration of Egress MQC Queuing on Port-Channel Member Link - no Etherchannel Load Balancing. This feature was introduced on Cisco ASR 1000 Series Routers.
Egress MQC Queuing Configuration Supported on Port-Channel Member Link - Etherchannel Load Balancing	Cisco IOS XE Release 2.5	This feature supports the configuration of Egress MQC Queuing on Port-Channel Member Link - Etherchannel Load Balancing. This feature was introduced on Cisco ASR 1000 Series Routers.
QoS Policies Aggregation - MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface	Cisco IOS XE Release 2.6	This feature supports the configuration of QoS Policies Aggregation - MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface. This feature was introduced on Cisco ASR 1000 Series Routers.



CHAPTER 40

Aggregate EtherChannel Quality of Service

The Aggregate EtherChannel Quality of Service (QoS) feature allows you to apply an aggregate egress-queuing policy-map on a port-channel main interface or subinterface. This feature enables QoS support on the aggregate port-channel main interface for the Cisco ASR 1000 Series Aggregation Services Routers.

Restrictions for Aggregate EtherChannel Quality of Service

- The configuration of QoS on Ethernet Virtual Circuit (EVC) with an aggregate port-channel interface is not supported.
- Point-to-Point Protocol over Ethernet (PPPoE) and IP over Ethernet (IPoE) sessions in the context of the Intelligent Services Gateway (ISG) and Intelligent Wireless Access Gateway (iWAG) (with or without QoS) across an aggregate port-channel interface is not supported.
- Virtual Private LAN Services (VPLS) with QoS on an aggregate port-channel interface is not supported.
- Xconnect with QoS on an aggregate port-channel interface is not supported.
- The use of fragment and service-fragment Modular QoS CLI (MQC) keywords in conjunction with the aggregate port-channel interface type is not supported.
- The aggregate-type port-channel interfaces have the following limitations:
  - All the member links of a port channel must be of the same speed. This prevents a potential packet reordering issue. It is not supported to combine Gigabit Ethernet, Fast Ethernet, or Ethernet interfaces into the same port channel.

- 10-Gigabit Ethernet is supported in Cisco IOS XE 3.16.3 or later (it is not supported in Cisco IOS XE 3.17). 10-Gigabit Ethernet is also supported in Cisco IOS XE Denali 16.3 and later.

- MPOL policy applied on both aggregate port-channel main interface and port-channel sub-interface is not supported by any Cisco IOS XE 3S release and is not supported on Cisco IOS XE Everest 16.5.x or earlier.
- QoS on an aggregate port-channel subinterface is not supported for Cisco IOS XE 3.16.2 or earlier (and it is also not supported in Cisco IOS XE 3.17).

Restrictions for PPPOE Session QoS over Aggregate EtherChannel

- All the member links of a port channel must be of the same speed. This prevents a potential packet reordering issue. It is not supported to combine Gigabit Ethernet, Fast Ethernet, or Ethernet interfaces into the same port channel.
- MPOL policy that is applied on both aggregate port-channel main interface and port-channel sub-interface is not supported.
- MPOL policy applied on both aggregate port-channel interface and PPPOE session is supported. The main interface or sub-interface QoS service policy is limited to only a class-default shaper (it can only contain the class class-default and shape command). Additional QoS configurations are not supported on the main interface or sub-interface when QoS service policies are applied to the main or sub-interface and the PPPOE session simultaneously.
- Before PPPOE session QoS is applied, the following command is required:

  platform qos port-channel-aggregate *port-channel interface*

  If the port-channel is already configured in any form, the above command fails.
- The QoS policy can be applied to an aggregate port-channel interface subject to the following scalability limits:
  - Upto 8 port channels
  - Upto 4 member links in a port channel
  - Member links can be split across multiple shared port adapters (SPAs) and SPA interface processor (SIP) cards

Restrictions for Non-Aggregate EtherChannel Quality of Service

The following restrictions apply if the port-channel is in non-aggregate mode:

- Applying Queuing policy on tunnels sourced on port-channel interface and port-channel sub-interface is not allowed.
- Per-Session QoS (VPN, DMVPN, PPP, PPPoE, PPPoVPDN, and DLEP) is not applied if the port-channel has more than one active member link. For multiple active member links, it is recommended to change the port-channel to aggregate mode.
- Per-Session QoS and Tunnel QoS with queuing features are not supported in load-balancing mode when the session or tunnel destination is reachable via multiple interfaces.

Note For a policy with queueing features, QoS queries CEF to obtain physical interface and port-channel association. If load-sharing is enabled, CEF returns a list of interfaces through which the session is reachable. QoS internally changes the CEF load-sharing method to per-prefix load-sharing to move traffic to a single interface; QoS policy is then applied to this interface. If a single interface is not available, the QoS policy is suspended.

If the interface load-sharing method was changed to per-prefix load-sharing, you must reload the device to return to default load-sharing.

Information About Aggregate EtherChannel Quality of Service

Supported Features for Aggregate EtherChannel Quality of Service

The Aggregate EtherChannel Quality of Service feature supports:

- Flow-based load balancing
- Up to three levels of hierarchy
- Configuration of shaping, absolute bandwidth, and relative bandwidth
- A minimum amount of bandwidth for subclasses (VLANs)
- Input QoS (policing and marking) and output QoS (all queuing features) that are enabled simultaneously on an aggregate port-channel main interface and subinterface

Unsupported Feature Combinations for Aggregate EtherChannel Quality of Service

The following combinations of tunnel-type interfaces with QoS are not supported:

- Generic Routing Encapsulation (GRE) tunnels with queuing policy-maps applied, which egress via a port channel with aggregate queuing
- Static virtual tunnel interface (SVTI) and dynamic virtual tunnel interface (DVTI) with queuing QoS applied, which egress via a port channel with aggregate queuing
- Sub-interface belongs to service group and sub-interface applied with service-policy cannot be configured on the same aggregate port-channel simultaneously
- MPOL - policy applied on both aggregate port-channel main interface and port-channel sub-interface

Note Tunnels without queuing QoS (described above) are supported, but are not recommended because hashing algorithms may overload a given physical interface without adequate diversity in IP addresses.

Scalability for Aggregate EtherChannel Quality of Service

The QoS policy can be applied to an aggregate port-channel interface subject to the following scalability limits:

- Up to 8 port channels
- Up to 4 member links in a port channel
- Member links can be split across multiple shared port adapters (SPAs) and SPA interface processor (SIP) cards

How to Configure Aggregate EtherChannel Quality of Service

This procedure describes how to configure Aggregate EtherChannel QoS on the Cisco ASR 1000 Series Aggregation Services Routers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos port-channel-aggregate** *port-channel-number*
4. **interface port-channel** *port-channel-number*
5. **service-instance** *service-instance-number*
6. **service-policy** {**output**} *policy-map*
7. **service-policy** {**input**} *policy-map*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**platform qos port-channel-aggregate** *port-channel-number* **Example:** `router(config)# platform qos port-channel-aggregate 1`	Enables the aggregate port-channel interface.
Step 4	**interface port-channel** *port-channel-number* **Example:** `router(config)# interface port-channel 1`	Enters interface configuration mode to configure a specific port channel.

	Command or Action	Purpose
Step 5	**service-instance** *service-instance-number* **Example:** `router(config)# service-instance 697`	Enables the service instance on the port channel.
Step 6	**service-policy** {**output**} *policy-map* **Example:** `router(config-if)# service-policy output egress_policy`	Attaches a policy-map to an output interface to be used as the service policy for that interface.
Step 7	**service-policy** {**input**} *policy-map* **Example:** `router(config-if)# service-policy input ingress_policy`	Attaches a policy-map to an input interface to be used as the service policy for that interface.

How to Unconfigure Aggregate EtherChannel Quality of Service

This procedure describes how to unconfigure Aggregate EtherChannel QoS on the Cisco ASR 1000 Series Aggregation Services Routers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no interface port-channel** *port-channel-number*
4. **no platform qos port-channel-aggregate** *port-channel-number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**no interface port-channel** *port-channel-number* **Example:** `router(config)# no interface port-channel 1`	Unconfigures a specific port channel.
Step 4	**no platform qos port-channel-aggregate** *port-channel-number* **Example:**	Disables the aggregate port-channel interface and removes the required QoS policies on it.

	Command or Action	Purpose
	router(config)# no platform qos port-channel-aggregate 1	

Configuration Examples for Aggregate EtherChannel Quality of Service

Enable Service Instance on Aggregate Port-Channel Interface

```
Router(config)#interface po5
Router(config-if)#service instance 697 ethernet
Router(config-if-srv)#service-policy output CUST-PMAP-S-L2ETH-4.65M-20-49-49-1_femo
Router(config-if-srv)#
```

Service Instance on Aggregate Port-Channel Interface

```
Router #show hqf interface po5 | inc layer
   blt (0x7FBAA95BD4D8, index 2, qid 13, fast_if_number 54) layer PHYSICAL
   (max entries 65536) (layer flags 0x10)
     next layer HQFLAYER_SERVICE_GROUP (max entries 65536)
     blt (0x7FBAA95BD3D8, index 0, qid 14, fast_if_number 54) layer SERVICE_GROUP
     (max entries 712) (layer flags 0x10)
       next layer HQFLAYER_SUB_IFC (max entries 712)
       blt (0x7FBAA95BD2D8, index 0, qid 15, fast_if_number 54) layer SUB_IFC
       blt (0x7FBAA95BD1D8, index 697, qid 16, fast_if_number 54) layer SUB_IFC
   (max entries 8) (layer flags 0xD)
   next layer HQFLAYER_CLASS_HIER1 (max entries 8)
           blt (0x7FBAA95BD0D8, index 0, qid 17, fast_if_number 54) layer CLASS_HIER1
           blt (0x7FBAA95BCFD8, index 1, qid 29, fast_if_number 54) layer CLASS_HIER1
           blt (0x7FBAA95BCAD8, index 2, qid 30, fast_if_number 54) layer CLASS_HIER1
           blt (0x7FBAA95BCBD8, index 3, qid 31, fast_if_number 54) layer CLASS_HIER1
           blt (0x7FBAA95BCCD8, index 4, qid 32, fast_if_number 54) layer CLASS_HIER1
```

Example: Configuring Aggregate Port-Channel Interface

```
Router# configure terminal
Router(config)# platform qos port-channel-aggregate 1
Router(config)# interface port-channel 1
Router(config-if)# interface GigabitEthernet1/0/1
Router(config-if)# channel-group 1
Router(config-if)# interface GigabitEthernet1/0/0
Router(config-if)# channel-group 1
Router(config-if)# interface port-channel 1.1
Router(config-subif)# encap
Router(config-subif)# encapsulation dot
Router(config-subif)# encapsulation dot1Q 2
Router(config-subif)# ip addr 14.0.1.2 255.255.255.0
Router(config-subif)# interface port-channel 1.2
Router(config-subif)# encapsulation dot1Q 3
Router(config-subif)# ip addr 14.0.2.2 255.255.255.0
Router(config-subif)# interface port-channel 1.3
Router(config-subif)# encapsulation dot1Q 4
Router(config-subif)# ip addr 14.0.3.2 255.255.255.0
Router(config-subif)# end
```

Example: Configuring a Class Map for QoS

```
Router# configure terminal
Router(config)# class-map vlan_2
Router(config-cmap)# match vlan  2
Router(config-cmap)# class-map vlan_3
Router(config-cmap)# match vlan  3
Router(config-cmap)# class-map vlan_4
Router(config-cmap)# match vlan  4
Router(config-cmap)# class-map prec1
Router(config-cmap)# match precedence 1
Router(config-cmap)# class-map prec2
Router(config-cmap)# match precedence 2
Router(config-cmap)# class-map prec3
Router(config-cmap)# match precedence 3
Router(config-cmap)# class-map prec4
Router(config-cmap)# match precedence 4
Router(config-cmap)# end
```

Example: Configuring a Policy-Map for QoS

```
Router# configure terminal
Router(config)# policy-map child-vlan
Router(config-pmap)# class prec1
Router(config-pmap-c)# police cir percent 20
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# priority level 1
Router(config-pmap-c)# class prec2
Router(config-pmap-c)# police cir percent 40
Router(config-pmap-c-police)# exit
Router(config-pmap-c)# priority level 2
Router(config-pmap-c)# class prec3
Router(config-pmap-c)# bandwidth remaining ratio 3
Router(config-pmap-c)# class class-default
Router(config-pmap-c)# bandwidth remaining ratio 1
Router(config-pmap-c)# random-detect
Router(config-pmap-c)#!
Router(config-pmap-c)# policy-map egress_policy
Router(config-pmap)# class vlan_2
Router(config-pmap-c)# shape average 100000000
Router(config-pmap-c)# service-policy child-vlan
Router(config-pmap-c)# class vlan_3
Router(config-pmap-c)# shape average 200000000
Router(config-pmap-c)# service-policy child-vlan
Router(config-pmap-c)# class vlan_4
Router(config-pmap-c)# shape average 300000000
Router(config-pmap-c)# service-policy child-vlan
Router(config-pmap-c)#!
Router(config-pmap-c)# policy-map ingress_policy
Router(config-pmap)# class vlan_2
Router(config-pmap-c)# police cir 80000000
Router(config-pmap-c-police)# conform-action set-prec-transmit 1
Router(config-pmap-c-police)# class vlan_2
Router(config-pmap-c)# set dscp AF21
Router(config-pmap-c)# class class-default
Router(config-pmap-c)# set dscp 0
Router(config-pmap-c)# end
```

Example: Applying QoS to Port Channel Interface

```
Router# configure terminal
Router(config)# interface port-channel 1
Router(config-if)# service-policy output egress_policy
Router(config-if)# service-policy input ingress_policy
Router(config-if)# end
```

How to Configure Aggregate EtherChannel Subinterface Quality of Service

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **platform qos port-channel-aggregate** *port-channel-number*
4. **interface port-channel** *port-channel-number*
5. **interface port-channel** *port-channel-number.subinterface-number*
6. **service-policy** {**output**} *policy-map*
7. **service-policy** {**input**} *policy-map*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**platform qos port-channel-aggregate** *port-channel-number* **Example:** `Device(config)# platform qos port-channel-aggregate 1`	Enables the aggregate port-channel interface.
Step 4	**interface port-channel** *port-channel-number* **Example:** `Device(config)# interface port-channel 1`	Enters interface configuration mode to configure a specific port channel.
Step 5	**interface port-channel** *port-channel-number.subinterface-number*	Enters interface configuration mode to configure a specific port channel subinterface.

	Command or Action	Purpose
	Example: `Device(config)# interface port-channel 1.2`	
Step 6	**service-policy** {**output**} *policy-map* **Example:** `Device(config-if)# service-policy output egress_policy`	Attaches a policy-map to an output interface to be used as the service policy for that interface.
Step 7	**service-policy** {**input**} *policy-map* **Example:** `Device(config-if)# service-policy input ingress_policy`	Attaches a policy-map to an input interface to be used as the service policy for that interface.
Step 8	**end** **Example:** `Device(config)# end`	Exits global configuration mode.

How to Unconfigure Aggregate EtherChannel Subinterface Quality of Service

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no interface port-channel** *port-channel-number.subinterface*
4. **no platform qos port-channel-aggregate** *port-channel-number*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no interface port-channel** *port-channel-number.subinterface* **Example:**	Unconfigures a specific port channel subinterface.

	Command or Action	Purpose
	`Device(config)# no interface port-channel 1.2`	
Step 4	**no platform qos port-channel-aggregate** *port-channel-number* **Example:** `Device(config)# no platform qos port-channel-aggregate 1`	Disables the aggregate port-channel interface and removes the required QoS policies on it.
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode.

Configuration Examples for Aggregate EtherChannel Subinterface Quality of Service

Example: Configuring Aggregate Port-Channel Interface and Subinterface

```
Device# configure terminal
Device(config)# platform qos port-channel-aggregate 2
Device(config)# interface port-channel 2
Device(config-if)# interface GigabitEthernet1/1/1
Device(config-if)# channel-group 2
Device(config-if)# interface GigabitEthernet1/1/0
Device(config-if)# channel-group 2
Device(config-if)# interface port-channel 2.200
Device(config-subif)# encapsulation dot1Q 200
Device(config-subif)# ip addr 15.0.1.2 255.255.255.0
Device(config-subif)# interface port-channel 2.300
Device(config-subif)# encapsulation dot1Q 300
Device(config-subif)# ip addr 15.0.2.2 255.255.255.0
Device(config-subif)# end
```

Example: Configuring a Class Map for QoS

```
Device# configure terminal
Device(config)# class-map vlan_2
Device(config-cmap)# match vlan  2
Device(config-cmap)# class-map vlan_3
Device(config-cmap)# match vlan  3
Device(config-cmap)# class-map vlan_4
Device(config-cmap)# match vlan  4
Device(config-cmap)# class-map prec1
Device(config-cmap)# match precedence 1
Device(config-cmap)# class-map prec2
Device(config-cmap)# match precedence 2
Device(config-cmap)# class-map prec3
Device(config-cmap)# match precedence 3
Device(config-cmap)# class-map prec4
```

```
Device(config-cmap)# match precedence 4
Device(config-cmap)# end
```

Example: Configuring a Policy-Map for QoS

```
Device# configure terminal
Device(config)# policy-map subinterface_child
Device(config-pmap)# class prec1
Device(config-pmap-c)# police cir percent 30
Device(config-pmap-c-police)# exit
Device(config-pmap-c)# priority level 1
Device(config-pmap-c)# class prec2
Device(config-pmap-c)# police cir percent 30
Device(config-pmap-c-police)# exit
Device(config-pmap-c)# priority level 2
Device(config-pmap-c)# class prec3
Device(config-pmap-c)# bandwidth remaining ratio 3
Device(config-pmap-c)# class class-default
Device(config-pmap-c)# bandwidth remaining ratio 1
Device(config-pmap-c)#!
Device(config-pmap-c)# policy-map sub_egress_policy
Device(config-pmap-c)# class class-default
Device(config-pmap-c)# shape average 300000000
Device(config-pmap-c)# service-policy subinterface_child
Device(config-pmap-c)#!
Device(config-pmap-c)# policy-map sub_ingress_policy
Device(config-pmap)# class class-default
Device(config-pmap-c)# police cir 80000000
Device(config-pmap-c)# end
```

Example: Applying QoS to Port Channel Subinterface

```
Device# configure terminal
Device(config)# interface port-channel 2.200
Device(config-if)# service-policy output egress_policy
Device(config-if)# service-policy input ingress_policy
Device(config)# interface port-channel 2.300
Device(config-if)# service-policy output egress_policy
Device(config-if)# service-policy input ingress_policy
Device(config-if)# end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Commands List, All Releases*
QoS commands	Cisco IOS Quality of Service Solutions Command Reference

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

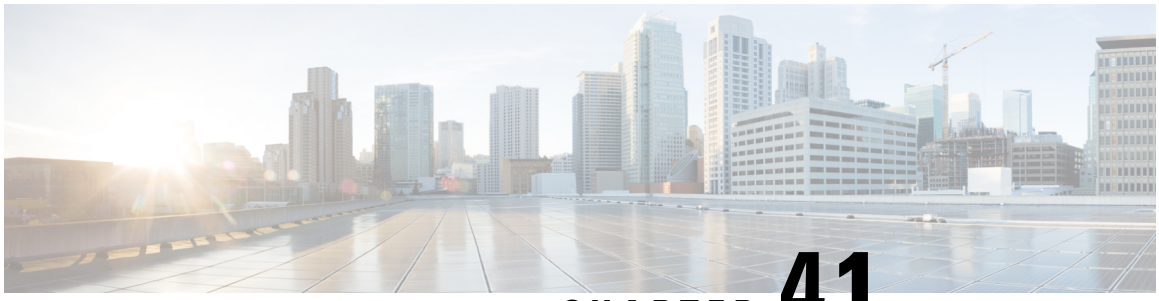
Feature Information for Aggregate EtherChannel Quality of Service

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 47: Feature Information for Aggregate EtherChannel Quality of Service

Feature Name	Releases	Feature Information
Aggregate EtherChannel Quality of Service	Cisco IOS XE Release 3.12S	The Aggregate EtherChannel Quality of Service (QoS) feature allows you to apply an aggregate egress-queuing policy-map on a port-channel main interface or subinterface. This feature enables QoS support on the aggregate port-channel main interface for the Cisco ASR 1000 Series Aggregation Services Routers. In Cisco IOS XE Release 3.12S, this feature was implemented on the Cisco ASR 1000 Series Aggregation Services Routers.
Aggregate GEC QoS 10G support	Cisco IOS XE Release 3.16.3S Cisco IOS XE Denali 16.3.1	In Cisco IOS XE Release 3.16.3S, this feature was implemented on the Cisco ASR 1000 Series Aggregation Services Routers.
QoS on GEC portchannel subinterface on ASR1K	Cisco IOS XE Release 3.16.3S Cisco IOS XE Denali 16.3.1	In Cisco IOS XE Release 3.16.3S, this feature was implemented on the Cisco ASR 1000 Series Aggregation Services Routers.
QoS on GEC portchannel subinterface on ISR 4000	Cisco IOS XE Everest 16.6.1	In Cisco IOS XE Everest 16.6.1 release, this feature was implemented on the Cisco ISR 4000 Series Integrated Services Routers.



CHAPTER 41

PPPoGEC Per Session QoS

The PPPoGEC Per Session QoS feature supports the configuration of specific QoS policies on PPPoE sessions on the PPP Termination and Aggregation (PTA), L2TP Access Concentrator (LAC), or L2TP Network Server (LNS) devices in a PPPoE /L2TP environment (broadband deployments). PPPoE sessions with Etherchannel Active/Standby functionality is also supported on Cisco ASR 1000 Series Routers acting as PTA, LAC, or LNS devices in a PPPoE/L2TP environment.

Information About PPPoGEC Per Session QoS

Restrictions for PPPoGEC Per Session QoS

- QoS policy-maps cannot be configured on member links, a port-channel main interface, or a port-channel subinterface that is associated with the transmit path for PPPoE sessions with QoS.

PPPoGEC Sessions with Active/Standby Etherchannel

PPPoE sessions with active/standby Etherchannel support one-level or two-level hierarchical output policy-maps (with queueing settings) also support flat input policy-maps (without queueing settings). The policy-maps are configured using previously defined class maps. The traffic classes must be configured using the **class-map** command.

The output hierarchical policy-map and the input policy-map can be associated with the PPPoE sessions in one of the following ways:

- Configuration settings on a virtual template interface
- Dynamic configuration settings via external tools configured in the authentication, authorization, and accounting (AAA) model (for example, a radius server). For more information, see the *Intelligent Services Gateway Configuration Guide* and the *Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*.

The port-channel main interface must contain the following commands that create an active/standby scenario. Such a configuration will allow only a single interface to be active and forwarding traffic at any time.

- **interface port-channel1**
- **lacp fast-switchover**
- **lacp max-bundle 1**

How to Configure PPPoGEC Per Session QoS

Configuring QoS on PPPoE Sessions with Etherchannel Active/Standby

To configure QoS on PPPoE sessions, you must specify the virtual template to use for PPP sessions on the Etherchannel interface, specify the name of the service policy that is applied to input traffic, and specify the output traffic. This configuration shows how to associate the output hierarchical policy-map and the input policy-map with the PPPoE sessions by defining a virtual template interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface virtual-template** *number*
4. **service-policy output** *policy-map-name*
5. **service-policy input** *policy-map-name*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface virtual-template** *number* **Example:** `Device(config)# interface virtual-template 99`	Creates a virtual template interface that can be configured and applied dynamically in creating virtual access interfaces, and enters interface configuration mode. • Specify the virtual template to use for PPP sessions on the Etherchannel interface.

	Command or Action	Purpose
Step 4	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output session_parent`	Specifies the name of the service policy that is applied to output traffic.
Step 5	**service-policy input** *policy-map-name* **Example:** `Device(config-if)# service-policy input session_ingress`	Specifies the name of the service policy that is applied to input traffic.
Step 6	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuration Examples for PPPoGEC Per Session QoS

Example: QoS on PPPoE Sessions with Etherchannel Active/Standby

The following example shows the session_parent hierarchical policy-map and the session_ingress policy-map. These policy-maps are attached to a virtual template interface using the **service-policy** command.

```
policy-map session_child
  class voice
    priority level 1
    police cir 256000
    set precedence 5
  class web
    bandwidth remaining ratio 10
  class p2p
    bandwidth remaining ratio 1
    set precedence 1
  class class-default
    set precedence 2
    bandwidth remaining ratio 5
!
policy-map session_parent
  class class-default
    bandwidth remaining ratio 1
    shape average 25000000
    service-policy session_child
!
policy-map session_ingress
  class voip
    police cir 256000
  class p2p
    police cir 256000 pir 512000
      conform-action set-prec-transmit 1
```

```
      exceed  set-prec-transmit 0
      violate drop
  class class-default
    police cir 5000000
      conform-action set-prec-transmit 2
      exceed  drop
!
interface Virtual-template 99
 service-policy output session_parent
 service-policy input session_ingress
```

Additional References for PPPoGEC Per Session QoS

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular Quality of Service Command-Line Interface	"Applying QoS Features Using the MQC" module
Configuring RADIUS-based policing	*Intelligent Services Gateway Configuration Guide*
CISCO ASR 1000 Series software configuration	*Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

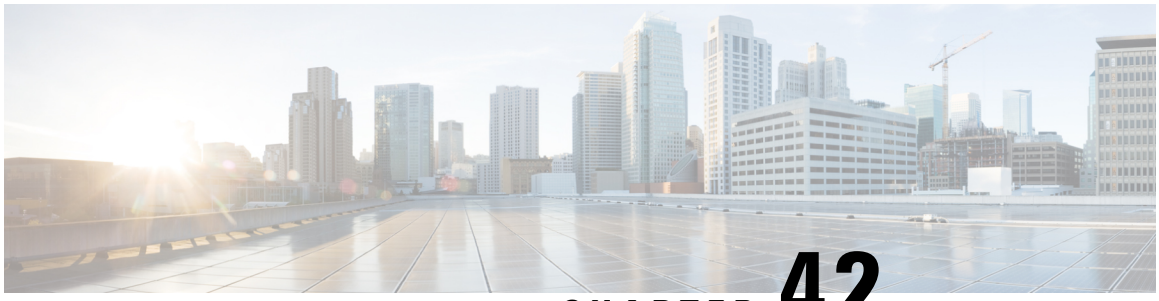
Feature Information for PPPoGEC Per Session QoS

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 48: Feature Information for PPPoGEC Per Session QoS

Feature Name	Releases	Feature Information
PPPoGEC: Per Session QoS	Cisco IOS XE Release 3.7S	This feature supports the configuration of specific QoS policies on PPPoE sessions on the PTA, LAC, and LNS for broadband deployments. This feature was introduced on Cisco ASR 1000 Series Routers. In Cisco IOS XE Release 3.8S, support was added for per-session QoS in 1:1 mode for PPPoGEC. Also, support for Point-to-Point Protocol (PPP) and IP over PPPoE was also added for PPPoGEC. In Cisco IOS XE Release 3.9S, support was added for IP session over GEC in 1:1 mode.



CHAPTER 42

IPv6 Selective Packet Discard

The selective packet discard (SPD) mechanism manages the process level input queues on the RP. SPD provides priority to routing protocol packets and other important traffic control Layer 2 keepalives during periods of process level queue congestion

Information About IPv6 Selective Packet Discard

SPD in IPv6 Overview

The SPD mechanism manages the process level input queues on the RP. SPD provides priority to routing protocol packets and other important traffic control Layer 2 keepalives during periods of process level queue congestion.

SPD State Check

The SPD state check is performed on the IPv6 process input queue on the RP. High-priority packets, such as those of IP precedence 6, are not applied to SPD and are never dropped. All remaining packets, however, can be dropped depending on the length of the IPv6 packet input queue and the SPD state. The possible SPD states are as follows:

- Normal: The process input queue is less than the SPD minimum threshold.
- Random drop: The process input queue is between the SPD minimum and maximum thresholds.
- Max: The process input queue is equal to the SPD maximum threshold.

The size of the process input queue governs the SPD state: normal (no drop), random drop, or max. When the process input queue is less than the SPD minimum threshold, SPD takes no action and enters normal state. In the normal state, no packets are dropped. When the input queue reaches the maximum threshold, SPD enters max state, in which normal priority packets are discarded. If the input queue is between the minimum and maximum thresholds, SPD enters the random drop state, in which normal packets may be dropped.

SPD Mode

Three IPv6 SPD modes are supported: none (which is the default), aggressive drop, and OSPF mode. The aggressive drop mode discards incorrectly formatted packets when the IPv6 is in the random drop state. OSPF mode provides a mechanism whereby OSPF packets are handled with SPD priority.

SPD Headroom

With SPD, the behavior of normal IPv6 packets is not changed. However, routing protocol packets are given higher priority, because SPD recognizes routing protocol packets by the IPv6 precedence field. Therefore, if the IPv6 precedence is set to 6, then the packet is given priority.

SPD prioritizes IPv6 packets with a precedence of 6 by allowing the Cisco IOS software to queue them into the process level input queue above the normal input queue limit. The number of packets allowed in excess of the normal limit is called the SPD headroom. The SPD headroom default is 100, which means that a high precedence packet is not dropped if the size of the input hold queue is lower than 175 (which is the input queue default size + SPD headroom size).

Because Interior Gateway Protocols (IGPs) and link stability are tenuous and crucial, such packets are given the highest priority and are given extended SPD headroom with a default of 10 packets. These packets are not dropped if the size of the input hold queue is lower than 185 (input queue default size + SPD headroom size + SPD extended headroom).

Non-IPv6 packets such as Connectionless Network Service Intermediate System-to-Intermediate System (CLNS IS-IS) packets, PPP packets, and High-Level Data Link Control (HDLC) keepalives are treated as normal priority as a result of being Layer 2 instead of Layer 3. In addition, IGPs operating at Layer 3 or higher are given priority over normal IPv6 packets, but are given the same priority as Border Gateway Protocol (BGP) packets. Therefore, during BGP convergence or during times of very high BGP activity, IGP hellos and keepalives often are dropped, causing IGP adjacencies to fail.

How to Configure IPv6 Selective Packet Discard

Configuring the SPD Process Input Queue

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 spd queue max-threshold** *value*
4. **ipv6 spd queue min-threshold** *value*
5. exit
6. **show ipv6 spd**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ipv6 spd queue max-threshold** *value* **Example:** `Router(config)# ipv6 spd queue max-threshold 60000`	Configures the maximum number of packets in the SPD process input queue.
Step 4	**ipv6 spd queue min-threshold** *value* **Example:** `Router(config)# ipv6 spd queue max-threshold 4094`	Configures the minimum number of packets in the IPv6 SPD process input queue.
Step 5	exit **Example:** `Router(config)# exit`	Returns the router to privileged EXEC mode.
Step 6	**show ipv6 spd** **Example:** `Router# show ipv6 spd`	Displays IPv6 SPD configuration.

Configuring an SPD Mode

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 spd mode** {**aggressive** | **tos protocol ospf**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**ipv6 spd mode** {**aggressive** | **tos protocol ospf**} **Example:** `Router(config)# ipv6 spf mode aggressive`	Configures an IPv6 SPD mode.

Configuring SPD Headroom

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spd headroom** *size*
4. **spd extended-headroom** *size*
5. exit
6. **show ipv6 spd**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**spd headroom** *size* **Example:** `Router(config)# spd headroom 200`	Configures SPD headroom.
Step 4	**spd extended-headroom** *size* **Example:** `Router(config)# spd extended-headroom 11`	Configures extended SPD headroom.
Step 5	exit **Example:** `Router(config)# exit`	Returns the router to privileged EXEC mode.

	Command or Action	Purpose
Step 6	**show ipv6 spd** **Example:** `Router# show ipv6 spd`	Displays the IPv6 SPD configuration.

Configuration Examples for IPv6 Selective Packet Discard

Example: Configuring the SPD Process Input Queue

The following example shows the SPD process input queue configuration. The maximum process input queue threshold is 60,000, and the SPD state is normal. The headroom and extended headroom values are the default:

```
Router# ipv6 spd queue max-threshold 5000
Router# show ipv6 spd

Current mode: normal
Queue max threshold: 60000, Headroom: 100, Extended Headroom: 10
IPv6 packet queue: 0
```

Additional References

Related Documents

Related Topic	Document Title
IPv6 addressing and connectivity	*IPv6 Configuration Guide*
Cisco IOS commands	Master Commands List, All Releases
IPv6 commands	IPv6 Command Reference
Cisco IOS IPv6 features	IPv6 Feature Mapping

Standards and RFCs

Standard/RFC	Title
RFCs for IPv6	*IPv6 RFCs*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

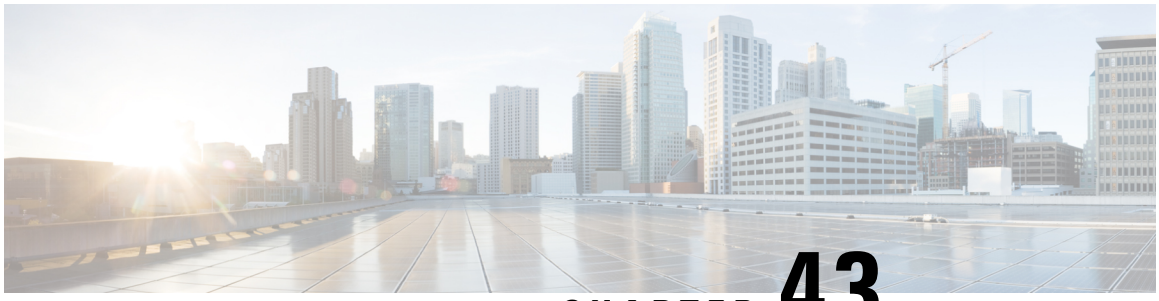
Feature Information for IPv6 Selective Packet Discard

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 49: Feature Information for IPv6 Selective Packet Discard

Feature Name	Releases	Feature Information
IPv6: Full Selective Packet Discard Support	Cisco IOS XE Release 2.6	The SPD mechanism manages the process level input queues on the RP. SPD provides priority to routing protocol packets and other important traffic control Layer 2 keepalives during periods of process level queue congestion. The following commands were introduced or modified: **clear ipv6 spd**, **debug ipv6 spd**, **ipv6 spd mode**, **ipv6 spd queue max-threshold**, **ipv6 spd queue min-threshold**, **monitor event-trace ipv6 spd**, **show ipv6 spd**, **spd extended-headroom**, **spd headroom**.



CHAPTER 43

Per ACE QoS Statistics

The Per ACE QoS Statistics feature extends the QoS Packet Matching Statistics feature to allow you to track the number of packets and bytes matching individual access control elements (ACEs) used in a filter. The filter is part of the class-map definition of a quality of service (QoS) policy-map.

You can use the **show access-lists** command to display per-ACE statistics.

See the "QoS Packet Matching Statistics" module for information on defining a QoS packet filter and displaying the number of packets and bytes matching that filter.

Prerequisites for Per ACE QoS Statistics

Before you configure the **platform qos match-statistics per-ace** command to enable QoS per-ACE packet-matching statistics, you must configure the **platform qos match-statistics per-filter** command to enable QoS per-filter packet-matching statistics. If you do not, the CLI rejects the command and displays an error message.

Restrictions for Per ACE QoS Statistics

If a QoS policy-map is attached to the device when you configure the **platform qos match-statistics per-ace** command, the command does not take effect until you do one of the following:

- Reload the device.
- Detach all QoS policies and configure the command again.

Enabling the Per ACE QoS Statistics feature may increase CPU utilization on a scaled configuration. Before you enable it, you should weigh the benefits of the statistics information against the increased CPU utilization on the system.

Note You must configure the **platform qos match-statistics per-filter** command before you configure the **platform qos match-statistics per-ace** command.

Information About Per ACE QoS Statistics

Per ACE QoS Statistics Overview

The Per ACE QoS Statistics feature provides hit counters for ACEs used in QoS policies. When enabled, the feature adds QoS hit counters for any ACEs used in a QoS policy to the existing security access-list counters for that ACE. You can use the **show ip access-lists** command to display the access-list counters, as shown in this example:

```
Device# show ip access-lists

Extended IP access list A1
10 permit ip 10.1.1.0 0.0.0.255 any (129580275 matches)
Extended IP access list A6and7
10 permit ip 10.1.6.0 0.0.0.255 any (341426749 matches)
20 permit ip 10.1.7.0 0.0.0.255 any (398245767 matches)
Extended IP access list source
10 permit ip any host 10.1.1.5 (16147976 matches)
```

The QoS hit counters (for ACEs used in QoS policies) will be added to the counters shown in the sample output.

Note the following conditions when you enable the Per ACE QoS Statistics feature:

- The **show ip access-lists** command does not display interface information. This means that access-list counts are not interface-specific; they are aggregate counters of all hits for all features that use the ACEs and support the counts across all interfaces and directions.
- You can use the **show policy-map interface** command to display interface-specific counts if QoS per-filter packet matching statistics is enabled. However, this command displays only counts per-filter [access-control list (ACL) or access-group], not counts per-ACE, as shown in this example:

```
Device# show policy-map interface GigabitEthernet0/0/2

 GigabitEthernet0/0/2

  Service-policy input: test-match-types

    Class-map: A1orA2-class (match-any)
      482103366 packets, 59780817384 bytes
      5 minute offered rate 6702000 bps
      Match: access-group name A1
        62125633 packets, 7703578368 bytes
        5 minute rate 837000 bps
      Match: access-group name A2
        419977732 packets, 52077238892 bytes
        5 minute rate 5865000 bps
```

- If an ACE is present in a QoS filter (that is, a match statement within a class map) but the packet does not match the ACE, the ACE counter is not incremented for that packet. This can happen in the following circumstances:
  - The ACE is used in a “deny” statement.
  - Other matching criteria in a “match-all” class-map definition (for example, “match ip prec 1”) prevent the packet from matching the class.
  - Other matching criteria in a “match-any” class-map definition (for example, “match ip prec 1”) match the packet and prevent it from matching the ACE match criteria (that filter precedes the ACE filter and the packet matches both statements).

- Access-list counts are an aggregate, for a particular ACE, of the hit counts for all features that use that ACE and support per-ACE counts. This means that a single packet might hit, and be counted by, multiple features using the same ACE, and, therefore, result in multiple counts for the same packet as it traverses each feature.

  The following example shows these multiple counts. Only 1,000 packets were received on the interface but the access-list counts show 2,000 hits, 1,000 for the security access list and 1,000 for the QoS service policy.

```
Device(config)# ip access-list extended A1
permit ip 32.1.1.0 0.0.0.255 any
class-map match-all A1-class
match access-group name A1
interface GigabitEthernet0/0/2
ip address 10.0.0.1 240.0.0.0
ip access-group A1 in
duplex auto
speed auto
media-type rj45
no negotiation auto
service-policy input simple
end

Device# show access-lists

Extended IP access list A1
10 permit ip 10.1.1.0 0.0.0.255 any (2000 matches)

Device# show policy-map interface GigabitEthernet0/0/2

Service-policy input: simple
Class-map: A1-class (match-all)
1000 packets, 124000 bytes
5 minute offered rate 4000 bps
Match: access-group name A1
Class-map: class-default (match-any)
0 packets, 0 bytes
5 minute offered rate 261000 bps, drop rate 0 bps
Match: any
```

How to Configure Per ACE QoS Statistics

Configuring Per ACE QoS Statistics

Before you begin

The **platform qos match-statistics per-filter** command must be configured to enable QoS per-filter packet-matching statistics. You can use the **show platform hardware qfp active feature qos config global** command to verify the status of packet-matching statistics.

```
Device# show platform hardware qfp active feature qos config global

Marker statistics are: disabled
Match per-filter statistics are: enabled <<<<<<<
Match per-ace statistics are: disabled <<<<<<
Performance-Monitor statistics are: disabled
```

SUMMARY STEPS

1. **platform qos match-statistics per-filter**
2. **platform qos match-statistics per-ace**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**platform qos match-statistics per-filter** **Example:** `Device(config)# platform qos match-statistics per-filter`	Enables QoS packet-matching statistics for individual filters in a class map.
Step 2	**platform qos match-statistics per-ace** **Example:** `Device(config)# platform qos match-statistics per-ace`	Enables QoS packet-matching statistics for ACEs used in QoS filters.

Additional References for Per ACE QoS Statistics

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*

Related Topic	Document Title
QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*
Defining a QoS packet filter and displaying the number of packets and bytes matching it	"QoS Packet Matching Statistics"

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

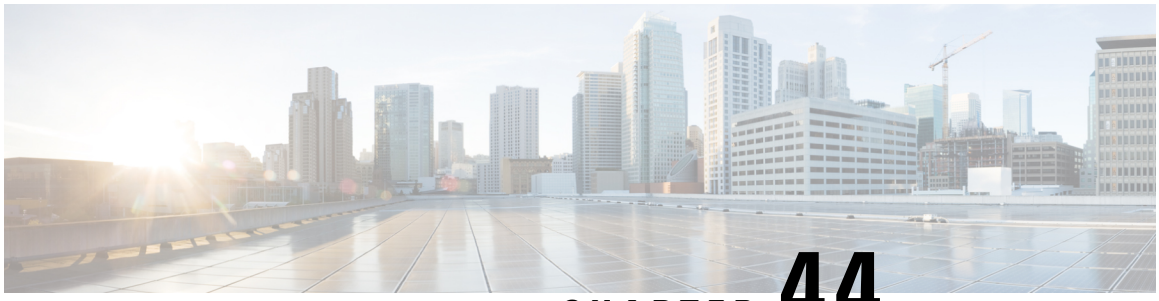
Feature Information for Per ACE QoS Statistics

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 50: Feature Information for Per ACE QoS Statistics

Feature Name	Releases	Feature Information
Per ACE QoS Statistics	Cisco IOS XE Release 3.10S	Allows you to configure per ACE QoS statistics to track the number of packets and bytes matching individual ACEs used in a filter within a QoS service policy. The following command was introduced or modified: **platform qos match-statistics per-ace**.



CHAPTER 44

QoS Packet Policing

Traffic policing allows you determine whether network traffic is above or below a predetermined rate and to provide different treatment for such traffic. In its simplest form a policer (rate limiter) drops any traffic that exceeds a predetermined rate.

About QoS Policing

Why Traffic Policing

Allowing you to control the maximum rate of traffic transmitted or received on an interface, traffic policing is typically configured on interfaces at the edge of a network to limit traffic into the network. In most traffic policing configurations, traffic that falls within the rate parameters is transmitted whereas traffic that exceeds the parameters is dropped or marked (and transmitted).

Note Unlike a shaper, a policer does not buffer packets. Rather, the specified action is taken immediately.

Typically, we use policers for admission control: queue or network.

Queue Admission Control limits the amount of data that can enter a queue. A *priority queue* is representative of this category wherein we avoid latency by limiting the rate at which packets may be enqueued.

Network Admission Control enforces a contract between the network administrator (service provider) and his customers. Generally both will agree on the rate at which the provider should accept traffic. This could be the

service-rate (max rate for all the traffic customer sends to provider) or a *per-class restriction* (e.g., the amount of priority traffic a customer may send).

- Using network admission control, you may decide to either drop excess traffic immediately or mark that traffic as 'out of contract.' If the latter, you can either provide that traffic a lesser treatment or drop it first if (and when) congestion occurs within this network.

Policer Definitions

Note The terms *Policer* and *Rate Limiter* usually refer to the same QoS mechanism. Policer (*Policing*) will be used throughout this document.

A policer is a device that allows you to define different treatments for packets within the same traffic class depending on whether packets are received above or below a specified rate(s).

In its simplest form, a policer indicates that traffic above a specified rate should be dropped:

```
policy-map police-all-traffic
  class class-default
    police 1m
```

Traffic through this class arriving at a rate less than 1 Mbps is considered *conforming* (adhering to the specified rate). The default action for conforming traffic is to forward packets.

Traffic arriving at a rate exceeding 1 Mbps is considered *exceeding* the configured rate. The default action for exceeding traffic is to drop packets.

The following definitions are relevant to understanding the sections that follow.

Policer Actions

In the previous example, copied below, we used a policer in its most basic form:

```
policy-map police-all-traffic
  class class-default
    police 1m
```

Conforming traffic was allowed to pass through the policer (transmitted traffic below 1m) whereas exceeding traffic was dropped. We took *immediate action* when we recognized that traffic had exceeded the specified rate. However, you may not want to always take immediate action. You might want to *defer action* rather than immediately drop traffic.

For example, you may decide that traffic above the predetermined rate should only be dropped if the network is congested. If so, you might choose to forward all traffic but mark something in the packet (e.g., DSCP) differently for conforming and exceeding traffic. The decision on whether or not to drop can then be made at the congestion point.

In the following example, we mark rather than drop traffic. We define a traffic class as any traffic arriving with a DSCP value of AF41 and demote traffic exceeding a specified rate to AF42:

```
policy-map ma
rk-out-of-contract
  class AF41
```

```
police 1m conform-action transmit exceed-action set-dscp-transmit AF42
```

The *conform-action* is to transmit traffic (simply forward, default action) arriving at a rate less than or equal to the specified 1 Mbps rate.

The *exceed-action* for traffic exceeding 1 Mbps is to mark the packet's DSCP value rather than drop traffic.

Transmit and drop represent *actions* specified for traffic *conforming* to or *exceeding* the specified rate. You specify an action with the **police** command. Supported actions are listed in the following table.

Note The rules for policer actions are very similar to those for the **set** command.

You can only mark Layer 2 and outer Layer 3 headers.

Multi-Action Policer

In the previous section we saw how a policer can be configured to mark some field in the packet. In fact, we can mark multiple fields in the packet.

You can apply multiple actions to traffic within each rate designation, analogous to how you configure multiple set actions within a traffic class. For example, if you know a packet will be transmitted through both a TCP/IP and a Frame Relay environment, you can change the DSCP value of the exceeding or violating packet, and also set the Frame Relay Discard Eligibility (DE) bit from 0 to 1 to indicate lower priority.

When specifying multiple policing actions, observe the following:

- You must enter policy-map class police configuration (config-pmap-c-police) submode.
- You can specify a maximum of four actions simultaneously, one line per action.
- You cannot specify contradictory actions such as **conform-action transmit** and **conform-action drop**.

Analogous to the **set** command, you can either configure multiple actions on the same packet (e.g., marking Layer 2 and Layer 3 fields) or define actions for different traffic types (e.g., marking the DSCP value in IPv4 packets and experimental (EXP) bits in MPLS packets).

In the following example, we cap RTP traffic (`rtp-traffic`) at 1 Mbps and drop traffic exceeding that rate (`exceed-action drop`). For conforming traffic, we mark both the COS and DSCP values in IPv4 packets and the COS and EXP bits in MPLS packets:

```
class rtp-traffic
  police cir 1000000
    conform-action set-cos-transmit 4
    conform-action set-dscp-transmit af41
    conform-action set-mpls-exp-topmost-transmit 4
    exceed-action drop
```

All packets in a traffic class count towards the rate seen by that class but actions are applied only to applicable traffic. For example, imagine that IPv4 and MPLS packets are classified into the same traffic class and a policer is configured to mark a specific DSCP value. Both IPv4 and MPLS packets count towards the observed rate, but only IPv4 packets can be marked.

Note Configuring multiple actions is supported for single and dual-rate policers.

A Note on CLI Variants

This section shows how multiple variants of the CLI can achieve the same result.

Context

The variations have emerged in different Cisco IOS software releases over time and as software trains have merged. Within the same software release, three equivalent variants exist. To avoid backwards compatibility issues, we decided to retain the variants. Please note, however, that the software implementing the policing is identical regardless of the CLI variant used.

Illustration

For the following examples, we set **police** to 10 Mbps, **conform action** to transmit (default), and **exceed action** to drop (default). At a "high" level we have three variants of the **police** command that achieve the same result: **police** *value*, **police cir** *value*, and **police rate** *value*. This set of variants is equivalent to: **police [cir|rate]***value*, where **cir** and **rate** are optional. With a rate of 10 Mbps, we can build the following command: **police [cir|rate] 10m**.

Using each variant to configure policing:

```
policy-map policer-cli-example
  class class-default
    police 10000000


policy-map policer-cli-example
  class class-default
    police cir 10m


policy-map policer-cli-example
  class class-default
    police rate 10m
```

To verify that the three variants yield the same result, you can use two stages of verification:

1. Issue **show policy-map interface** to display the configuration within IOS.
2. Issue **show platform hardware qfp active feature qos interface** to illustrate how we program hardware. This display is unchanged regardless of the CLI variant used.

Let's run Step 1:

```
show policy-map int GigabitEthernet1/0/0

  Service-policy input: policer-cli-example

    Class-map: class-default (match-any)
      162 packets, 9720 bytes
      5 minute offered rate 2000 bps, drop rate 0000 bps
      Match: any
      police:
```

```
    cir 10000000 bps, bc 312500 bytes
  conformed 212 packets, 12720 bytes; actions:
    transmit
  exceeded 0 packets, 0 bytes; actions:
    drop
  conformed 2000 bps, exceeded 0000 bps
```

Next, let's run Step 2:

```
show platform hardware qfp active feature qos int g1/0/0
```

```
Interface: GigabitEthernet1/0/0, QFP interface: 12
  Direction: Input
    Hierarchy level: 0
    Policy name: policer-cli-example
      Class name: class-default, Policy name: policer-cli-example
        Police:
            cir:  10000000 bps,  bc: 315392 bytes
            pir: 0 bps,  be: 315392 bytes
            rate mode: Single Rate Mode
            conformed: 16 packets, 960 bytes; actions:
              transmit
            exceeded: 0 packets, 0 bytes; actions:
              drop
            violated: 0 packets, 0 bytes; actions:
              drop
            color aware: No
            green_qos_group: 0, yellow_qos_group: 0
```

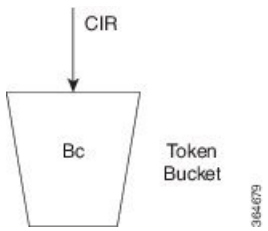
Single-Rate, Two-Color Policer

A single-rate, two-color policer (1R2C) determines whether traffic is above or below a predetermined rate (CIR in bps) and allows you to take action in either instance. The possible actions for any arriving packet are conform (packet counts as traffic falling below the CIR) and exceed (packet counts as traffic exceeding the CIR).

We need to allow for any *potential burstiness*. This behavior occurs when many packets arrive together, and the arrival rate over a short interval exceeds the CIR while the arrival rate over a longer range might conform to the CIR. To accommodate bursts yet enforce our predetermined CIR over time, we use a *token bucket* scheme.

Applying this scheme, we can represent a single-rate, two-color policer with a single-token bucket:

Figure 76: Single-Rate, Two-Color Policer



Tokens are continuously replenished at CIR and the depth of the bucket is Bc. If the bucket is full, additional tokens arriving are lost.

When a packet arrives, the policer assesses whether the bucket contains enough tokens (bytes) to *cover that incoming packet* (sufficient bytes to match the packet length). If so, the packet is regarded as conforming, the action is taken and the appropriate number of tokens (packet length) is removed from the bucket.

If the packet arrives and the bucket contains insufficient tokens to cover the packet, the exceed action is taken; the number of tokens in the bucket are unchanged. Subsequent packets may find that the bucket has replenished sufficiently to be now designated "conforming." If no packets arrive, the bucket continues to fill to the burst limit (Bc).

Specifying the bucket depth determines the allowable amount of burstiness for conforming traffic (how many bytes/packets) that may arrive closely together, assuming the bucket has had time to refill.

In this example we have specified a CIR of 10 Mbps and a burst allowance of 15000 bytes. So, a burst of 10 MTU-sized packets on an Ethernet interface could be designated conforming:

```
policy-map police-with-burst
  class class-default
    police cir 10m bc 15000
```

Note The current IOS CLI enables you to configure policing in multiple ways yet accomplish the same result.

Single-Rate, Three-Color Policer

A single-rate, three-color policer (1R3C) supports three possible output states: conform, exceed and violate. The definition of conform is analogous to that in a 1R2C policer – traffic that adheres to a predetermined rate allowing for some burst tolerance.

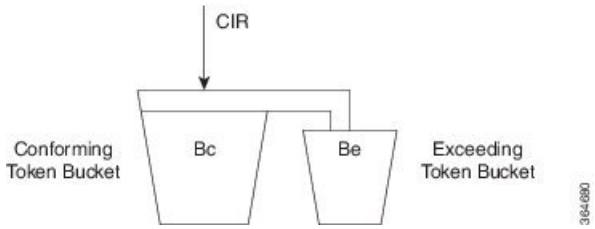
The difference stems from how we designate traffic that does not conform – traffic that a two-color policer would designate as exceed. We introduce further granularity where this traffic could be exceed or violate. Essentially, traffic that bursts ‘minimally’ above the CIR is designated as exceed but more sustained bursts above the CIR would be designated as violate.

To achieve this behavior we introduce a second token bucket. Just as the conforming token bucket is used to differentiate between traffic that conforms or exceeds, the excess token bucket enables us to differentiate between traffic that exceeds or violates.

Here are the bucket scenarios:

- The conforming token bucket is initially full (the number of bytes specified as Bc (conforming burst size).
- The exceeding token bucket is initially full (the number of bytes specified in the Be (excess burst size).
- If the conforming token bucket is full when tokens arrive (at the CIR, analogous to a 1R2C policer), they overflow into the excess token bucket.
- If both buckets are full, further tokens are lost.

Figure 77: Single-Rate, Three-Color Policer



The conforming bucket here behaves as it does in the 1R2C scenario. If the bucket contains sufficient tokens to cover the incoming packet, the packet is considered "conforming," the conforming action occurs and we remove an appropriate number of tokens from the bucket. The exceeding bucket is unaffected and we continue to replenish the conforming bucket (Bc) at CIR.

However, if the conforming bucket is full and additional tokens arrive they are not immediately lost. Instead, they overflow into the exceeding bucket. If this bucket is full, excess tokens are lost.

Similarly, when a packet arrives and the conforming bucket has insufficient tokens to cover that packet we cannot immediately declare it as exceeding; it might be exceeding or violating. If the exceeding bucket has enough tokens to cover the packet, the exceeding action is taken, and we remove the necessary number of tokens from the exceeding bucket. No bytes are removed from the conforming bucket.

If neither bucket, conforming or exceeding, has enough tokens to cover the packet, it is categorized as violating and the appropriate action is taken. Neither the conforming nor exceeding bucket is decremented:

If neither bucket, conforming or exceeding, has enough tokens to cover the packet, it is categorized as violating and the appropriate action is taken. Neither the conforming nor exceeding bucket is decremented:

```
policy-map ingress-enforcement
  class af41-metering
    police cir percent 10 bc 5 ms be 10 ms
    conform-action set-dscp-transmit af41
    exceed-action set-dscp-transmit af42
    violate-action drop
```

In this example we are policing traffic (for class af41) to 10% of the interface's bandwidth and the following apply:

- Traffic (**`conform-action set-dscp-transmit af41`**) burst up to 5 ms is forwarded and still marked as af41.
- Traffic (**`exceed-action set-dscp-transmit af42`**) burst exceeding 5 ms and up to an additional 10ms of burst is marked as af42. Elsewhere in the network, when we detect af42, we know it was received beyond the agreed contract [at the edge of the network]; under congestion, we could drop it first.
- Traffic (**`violate-action drop`**) burst beyond 15 ms above our CIR is considered violating and dropped immediately.

Note We only replenish the exceeding bucket when the conforming bucket is full. So, if you send a non-bursty stream at a rate exceeding the CIR, shortly, both the conforming and exceeding buckets will be drained; we do not replenish the exceeding bucket. All subsequent packets are considered either conforming or violating.

Dual-Rate, Three-Color Policer

Traffic rates are easier to understand than traffic burstiness. When specifying a contract for network admission control , you might have trouble describing expectations in terms of multiple burst sizes above a single rate. The dual-rate, three-color (2R3C) policer simplifies matters by primarily employing rates to differentiate conform, exceed and violate. It also introduces a second rate, PIR (Peak Information Rate)

CIR and PIR have the following characteristics:

- Traffic below the CIR is conforming.
- Traffic greater than CIR but less than PIR is exceeding.
- Traffic above PIR is violating.

You specify these rates with the **cir** and **pir** keywords of the **police** command.

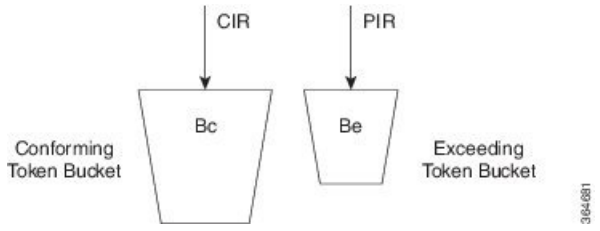
With a 2R3C policer, unlike a 1R3C, we replenish token buckets independently whenever a packet arrives at the policer. We refill conforming buckets at rate CIR; it can contain up to value Bc. ; exceeding buckets, at PIR; it can contain up to value Be.

Note PIR must exceed CIR and overflow between buckets is disallowed.

If a steady stream of packets arrives at a rate exceeding the CIR but less than the PIR, all packets are marked either conforming or exceeding. With the 1R3C policer, this scenario would have resulted in marking a minimal number of packets as exceeding and a majority as conforming or violating.

A 2R3C policer supports three possible actions for each packet: conform, exceed, and violate. Traffic entering the interface configured with a dual-rate policer is placed into one of these action categories, which dictates how we treat a packet. For instance, in the most common configuration, you can configure to send packets that either conform or exceed (with a decreased priority), and to drop packets that violate.

Figure 78: Dual-Rate, Three-Color Policer



When a packet arrives, we assess whether ample tokens exist in the conforming and exceeding buckets to cover that packet. If so, we take the conforming action (typically, transmit or transmit and mark) and remove the necessary tokens to transmit the packet from both buckets.

If the Exceeding Token Bucket (but not the Conforming Token Bucket) contains sufficient tokens to cover the packet, we take the exceeding action (typically, transmit or transmit and marking). The appropriate number of tokens are removed from the exceeding bucket only.

If neither bucket has sufficient tokens to cover the packet, the violating action is taken (typically, transmit, transmit and marking, or drop):

```
policy-map ingress-enforcement
  class af41-metering
    police cir 100k bc 3000 pir 150k be 3000 conform-action set-dscp-transmit af41
exceed-action set-dscp-transmit af42 violate-action drop
```

Observe how code from the preceding example and the corresponding code from **Single-Rate, Three-Color Policer** differ:

```
cir 100k bc 3000 pir 150k be 3000
cir percent 10 bc 5 ms be 10 ms
```

In the immediate example, we handle traffic accordingly:

- Up to 100Kbps (allowing for bursts up to 3,000 bytes) as conforming and forward it with DSCP marked as af41.
- Above 100Kbps but less than 150Kbps (again allowing a 3,000 byte burst) as exceeding and forward it marked as af42.
- Above 150Kbs as violating; we drop it.

Configuring Rates and Burst Parameters

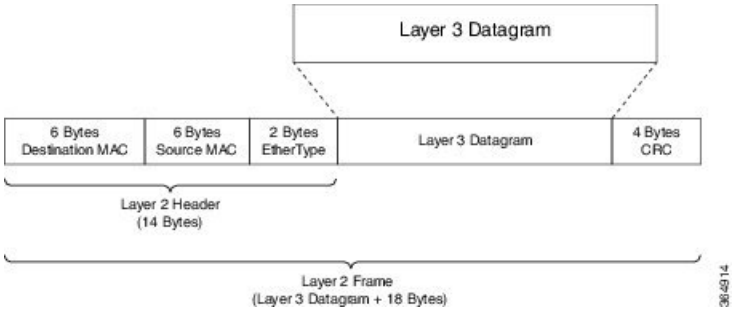
What's Included in the Policer-Rate Calculation (Overhead Accounting)

When specifying a rate or burst value, you should know how the policer assesses a packet's length (subsequently referred to as the *policing length*) when you evaluate conformance to those values. Briefly, a policer includes the Layer 3 datagram Layer 2 header lengths but neither CRC nor inter-packet overhead.

To further illustrate, consider an IP datagram transported over a GigabitEthernet link.

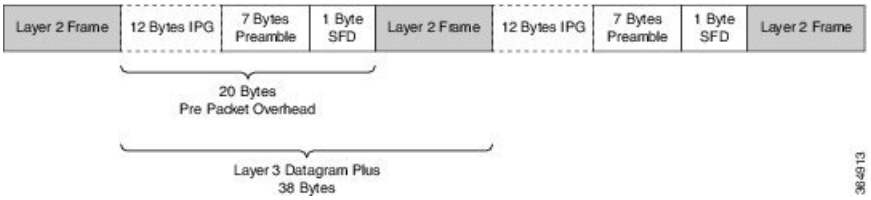
Layer 3 Datagram

First, we encapsulate it in an Ethernet frame, which adds 14 bytes of Layer 2 header and an additional 4 bytes of CRC to each datagram (18 bytes):



Ethernet Overhead

To transmit this frame over the physical medium, Ethernet requires a minimum inter-packet gap equivalent to a transmit time for 12 bytes of data. After the gap, we require seven bytes of preamble followed by a single byte start-of-frame delimiter (SFD) (Ethernet inter-packet overhead = 12 bytes IPG + 7 bytes Preamble + 1 byte SFD = 20 bytes).

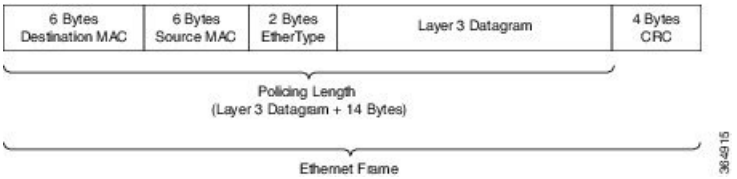


So, if you send multiple Ethernet frames sequentially, the per-packet overhead for each Layer 3 datagram is an additional 38 bytes (encapsulation [18 bytes] + Ethernet inter-packet overhead [20 bytes]). For example, if you sent 100 byte IP datagrams at line rate on a GigabitEthernet link, and used the following formula, the expected throughput in packets per second would be:

```
Linerate / Bits Per
  Byte / ( Layer 3 length + Per Packet Overhead) = Packets Per Second

1 Gbps / 8 / ( 100 + 38 ) = 905, 797 pps
```

From the policer's perspective, the packet's length is the Layer 3 datagram + Layer 2 header length (14 bytes on a GigabitEthernet interface):

Policing Length



Now consider a 500 Mbps policer configured on a GigabitEthernet interface. As in the previous example, we will send all 100 byte IP datagrams to the policer, resulting in a policing length of 100 byte datagram length + the 14 byte (Ethernet Layer 2 header). According to the following formula, the anticipated throughput would now be:

```
Policer Rate / Bits
  per Byte / (Layer 3 length + Layer 2 header length) = Packets Per Second

500 Mbps / 8 / (100 + 14) = 548,246 pps
```

Note Packets marked as *conforming* by a 500 Mbps policer will consume considerably more than 500 Mbps of physical bandwidth!

Policer on Logical Interface

On egress, a policer is unaware of the final physical interface type (tunnels can move between interfaces) and therefore the policer is unaware of the final Layer 2 overhead. So, the latter is excluded from the policing length. Similarly, because the policer cannot predict the extent of packet expansion due to overhead, if we configure encryption, we will not include encryption overhead in policer rate calculations. The egress policer will include the Layer 3 datagram and any tunnel headers (e.g., additional IP header, GRE header).

On ingress, because a policer is aware of the receiving interface type, policing on a tunnel interface includes Layer 2 overhead plus any tunnel headers.

The following table illustrates the dependencies of policer rate calculations on a ASR 1000 Series Aggregation Services Router. Be aware that we present only a subset of all permutations:

where the values are defined as follows:

- 0 - svti ('tunnel mode ipsec ipv4') has no overhead
- 14 - Layer 2 Ethernet header size
- 20 - the IP/IP header size
- 24 - the IP/GRE header size (20 + 4)

Policer on ATM Interfaces

If a policer is configured on an ATM interface, the policing length includes the Layer 3 datagram and the ATM adaption layer (AAL) header. For AAL5SNAP encapsulation length, this means that we include eight bytes of header in the policing length; for AAL5NLPID encapsulation, two bytes.

This calculation differs sharply from that applied to scheduling, where we include the complete *AAL PDU* and *cell tax*.

Changing What's Included - Overhead Accounting Adjustment

In prior sections, we described what is included by default in policer rate calculations. But what happens when you want to deviate from the default? For example, what if you want to express CIR as the physical bandwidth that would be consumed on a link? For an Ethernet interface you would include the 4 byte CRC and the 20 bytes inter-packet overhead required per packet.

Alternatively, you (a service provider) might want to police customers' traffic at Layer 3 rates. Because datagram length is unchanged as a packet traverses different interface types (or encapsulating protocols), we would not include Layer 2 header length in policer rate calculations.

Note Any interface that supports QoS policies will support overhead accounting adjustment.

Note Changing overhead accounting may impact the network. For example, if you use a policer for network admission control, you might need to configure a corresponding shaper on the equipment that connects to that network. The two views of what is included in CIR (shaper and policer) should match.

In the following example we want to include all inter-packet overhead such that a policer will allow up to 50% of the traffic on the physical link to be conforming. By adding 24 bytes per packet (**`user-defined 24`**) we address the 4 byte CRC and the 20-byte inter-packet overhead.

```
policy-map ethernet-physical-example
  class class-default
    police cir percent 50 account user-defined 24
```

Using the **atm** keyword of the **police account** command, you can direct the policer to compensate for ATM cell division and cell padding (ATM cell tax) in rate calculations.

To include cell tax and cover the AAl5 trailer, a router first adds 8 bytes to the policing length. Then, it calculates the number of ATM cells (48 bytes of data carried per 53 byte cell) required to carry the packet

and multiplies this number by 53. For example, a 46 byte datagram would require 2 cells and therefore, if cell tax is included, the policing length would be considered "106 bytes."

In the following example, we show a 5 Mbps policer, which must include the cell-tax in its rate calculations:

```
policy-map include-cell-tax-example
  class class-default
    police cir 5000000 account user-defined 0 atm
```

The **atm** in the configuration dictates that we include the cell tax.

Restrictions for Overhead Accounting Adjustment

- If you enable overhead accounting on a child policy, then you must enable overhead accounting on the parent policy.
- In a policy-map, you must either enable or disable overhead accounting for all classes in a policy. Within the same policy, you cannot enable overhead accounting for some classes and disable overhead accounting for other classes.
- Overhead accounting is not reflected in any QoS counters (e.g., classification, policing, or queuing).
- You can enable overhead accounting on top-level parent policies as well as on both middle-level and bottom-level child policies. Child policies inherit overhead accounting policies configured at the "parent" or "grandparent" level.
- The overhead accounting type or value used within a policy-map and between the parent and the child policy-maps (in a hierarchical policy-map structure) must be consistent.

Overhead Accounting Adjustment (Predefined Options)

Through some predefined CLI options (based on broadband use cases), you can specify the encapsulation while the router adds or subtracts the appropriate number of bytes (see the following table).

Imagine that we send (or receive) traffic on an Ethernet interface to a DSLAM (digital subscriber line access multiplexer) elsewhere in the network. Although we are encapsulating in Ethernet frames (e.g., Dot1Q or Q-in-Q), the DSLAM encapsulates in some form of ATM encapsulation. We want the policer to execute on traffic as it would appear after the DSLAM. In all instances, we would add cell-tax to the policing length.

In the following example, we apply predefined overhead accounting values. If we receive Dot1Q-encapsulated packets on an Ethernet interface, an upstream DSLAM receives *AAL5-Mux 1483 routed encapsulated* packets, then strips the ATM and adds the Ethernet headers. On the ATM interface, the datagram would have 3 bytes of additional AAL headers but would not have the 18 bytes of Ethernet headers (including Dot1Q). So, the PDU would be 15 bytes less on the ATM interface (we subtract 15 bytes from the policing length and then add the cell-tax):

```
policy-map atm-example
  class class-default
    police 5000000 account dot1q aal5 mux-1483routed
```

Default Burst Sizes

If you don't explicitly configure a burst tolerance value (Bc or Be), IOS will configure a default. This default burst tolerance is 250 ms of data based on the appropriate rate. For example, if the CIR is 100 Mbps then 250 mS of this rate would be 100000000/8 x 0.250 = 3125000 bytes.

Bc and Be for a single rate policer are always based on the CIR. The Be for a dual-rate policer is based on the PIR:

Note When configuring a policer for queue admission control , set Bc to something suitable for applications in that queue (e.g., for a voice application, set Bc to 10 milliseconds or less).

```
policy-map policer-default
  class af41
    police cir 20000000 pir 40000000 conform-action transmit  exceed-action
                \ set-dscp-transmit af42 violate-action set-dscp-transmit af43

 show policy-map interface
 GigabitEthernet1/0/0

  Service-policy input: policer-default

    Class-map: af41 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match:  dscp af41 (34)
      police:
          cir 20000000 bps, bc 625000 bytes           1
          pir 40000000 bps, be 1250000 bytes          2
        conformed 0 packets, 0 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          set-dscp-transmit af42
        violated 0 packets, 0 bytes; actions:
          set-dscp-transmit af43
        conformed 0000 bps, exceeded 0000 bps, violated 0000 bps
```

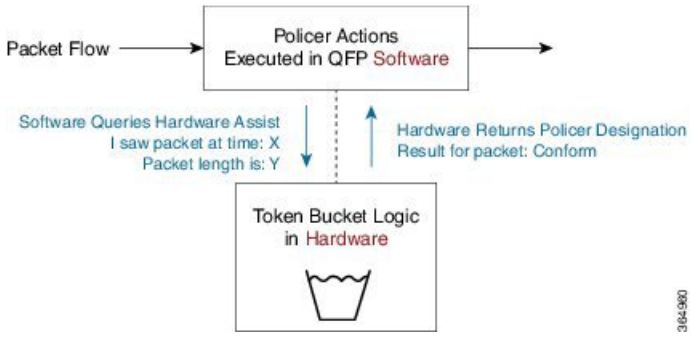
Note The dual-rate policer as well as Bc and Be default to 250ms based on the CIR (**1**) and PIR (**2**), respectively.

Rate and Burst Sizes Programmed in Hardware

On the Cisco ASR 1000 router platform, policer rate calculations are performed in *dedicated hardware*.

While *hardware assist* enables you to scale the number of policers independent of performance impact, it imposes some restrictions on the programmable rate and burst value combinations.

Figure 79:



Consider a simple policy with a single-rate, two-color policer:

```
policy-map hardware-example
  class class-default
    police cir 1m bc 3000
```

Output from the **show policy-map interface** command confirms that IOS has accepted the configured CIR and Bc values:

```
show policy-map interface g1/0/0


GigabitEthernet1/0/0

  Service-policy input: hardware-example

    Class-map: class-default (match-any)
      337 packets, 167152 bytes
      5 minute offered rate 2000 bps, drop rate 0000 bps
      Match: any
      police:
          cir 1000000 bps, bc 3000 bytes                    *
        conformed 337 packets, 167152 bytes; actions:
          transmit
        exceeded 0 packets, 0 bytes; actions:
          drop
        conformed 2000 bps, exceeded 0000 bps
```

***** CIR and Bc configured as expected

If you look at the dataplane, however, you can see the values actually programmed in hardware. Following is the output of the **show platform qfp active feature qos interface** command, which displays the actual policer values in hardware:

```
show platform hardware qfp active feature qos interface gig1/0/0


Interface: GigabitEthernet1/0/0, QFP interface: 9

  Direction: Input
    Hierarchy level: 0
    Policy name: hardware-example
      Class name: class-default, Policy name: hardware-example
        Police:
            cir:  1000000 bps,  bc: 3264 bytes              *
            pir: 0 bps,  be: 3008 bytes
```

```
rate mode: Single Rate Mode
conformed: 19 packets, 9424 bytes; actions:
  transmit
exceeded: 0 packets, 0 bytes; actions:
  drop
violated: 0 packets, 0 bytes; actions:
  drop
color aware: No
green_qos_group: 0, yellow_qos_group: 0
```

***** Bc as modified for hardware assist

Note Although we might slightly modify rate and burst parameters to accommodate the hardware assist, the platform always aims to retain the rates and resulting accuracy within 1% of what you request.

Percent-based Policer

The Percentage-based Policing feature enables you to configure traffic policing based on a percentage of the bandwidth available on the interface. Hence, you can use the same policy-map for multiple interface types with differing amounts of bandwidth. Recalculating the bandwidth for each interface or configuring a different policy-map for each type of interface is unnecessary.

Note If the interface is a shaped-ATM permanent-virtual circuit (PVC), we calculate the total bandwidth as follows:

- For a variable bit rate (VBR) virtual circuit (VC), the sustained cell rate (SCR) is used.
- For an available bit rate (ABR) VC, the minimum cell rate (MCR) is used.

You can use percentage-based policers for both CIR and PIR, calculating either from a specified percentage of either the interface bandwidth or parent shaper (if one exists).

With percent-based policing, if you choose to specify burst parameters (Bc and Be), they must be in ms rather than bytes. Given the speed of the target interface, IOS converts the value to bytes in two steps:

1. Using the speed of a target interface, IOS converts percentage to bps CIR
2. With bps CIR and *burst in time*, burst is converted to bytes.

Let's configure the Bc to 10 ms (relative to the police rate) and the CIR to 10% of the available interface bandwidth:

```
policy-map police-percent
  class class-default
    police cir percent 10 bc 10 ms
```

If we apply police-percent to a GigabitEthernet interface (1Gbps nominal bandwidth), IOS converts the CIR to 100 Mbps and the Bc to 125,000 bytes (100 Mbps x 10msec / 8):

```
show policy-map interface GigabitEthernet1/0/0

  Service-policy input: police-percent
```

```
    Class-map: class-default (match-any)
      834 packets, 413664 bytes
      5 minute offered rate 13000 bps, drop rate 0000 bps
      Match: any
      police:
        cir 10 %, bc 10
        cir 100000000 bps, bc 125000 bytes  Configured CIR and Bc converted to bps and
bytes, respectively.
```

Now, if we attach police-percent to a POS OC3 interface, the rate will be based on a nominal bandwidth of 155 Mbps. CIR will be calculated as 15.5 Mbps; the Bc, 19375 bytes (15.5 Mbps x 10msec / 8):

```
show policy-map interface POS1/1/0

  Service-policy input: police-percent

    Class-map: class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
      police:
          cir 10 %, bc 10
          cir 15500000 bps, bc 19375 bytes  Configured CIR and Bc converted to bps and
bytes, respectively.
```

Color-Aware Policers

A *color-aware policer* accounts for any preexisting markings that were determined by a previous node's policer as *in-contract* or *out-of-contract* (the previous node is typically at the edge of the network). Where color-aware policer is configured, we use such markings to determine the appropriate policing action for the packet. Traffic that was designated out-of-contract will always remain out-of-contract. Traffic that was designated in-contract may be demoted to out-of-contract by the new policer.

The ASR 1000 provides a limited implementation of color-aware policing; we restrict the contents of the class-maps used to determine the existing color of traffic:

- Only QoS group matching is supported in color-aware class-maps (only classification based on qos-group is supported.)
- Only one filter (one **match qos-group** *value* statement) is supported per color-aware class.

  You can use a child policy to set the qos-group based on a field you want in the received packet.
- Color-aware "specific" statistics are not supported.
- You cannot use the **no class-map** command to remove a color-aware map provided it is referenced in a color-aware policer. You must first remove all color-aware policers (using either the **no conform-color** or the **no exceed-color** command).

The "color" in color aware policing refers to how we educate the policer on how to interpret pre-existing markings in a received packet. Typically, we use Green to represent traffic that was pre-marked as *conforming* or in-contract. Similarly, Yellow represents traffic that was pre-marked as *exceeding* or out-of-contract.

Note that Green or Yellow are representative only; the CLI uses *conform-color* and *exceed-color* instead. Through the **police** command, these keywords specify class-maps that are used to determine the pre-existing color of that packet.

The following example shows how a child policy-map enables you to specify pre-existing color based on any field in the received packet. The color-aware policer is configured in a class that matches all packets from one of the DSCP assured forwarding traffic classes, AF4.

For this example, a packet marked AF41 is in-contract (conform or green), AF42 is out-of-contract (exceed or yellow) and AF43 is violating. The child policy mark-existing-color classifies packets based on the received DSCP, internally marking AF41 packets as qos-group 1 and AF42 packets as qos-group 2.

The color-aware policer will use the pre-conform (classify green packets) and pre-exceed (classify yellow packets) class-maps to determine the existing color of an arriving packet. Although these class-maps only support the qos-group filter, use of the child policy allows us to determine the pre-existing color based on the DSCP value in the received packet:

```
class-map af4
  match dscp af41 af42 af43
!
class-map af41
  match dscp af41
class-map af42
  match dscp af42
!
class-map pre-conform               !These are policer
  match qos-group 1                 !class-maps that
class-map pre-exceed                !only support qos-group
  match qos-group 2
!
policy-map mark-existing-color      !We use a child policy
  class af41                        !to set qos-group
    set qos-group 1                 !based on DSCP in the
  class af42                        !received packet
    set qos-group 2
!
policy-map dual-rate-color-aware
  class af4
    police cir 1m bc 5000 pir 2m be 5000
      conform-action set-dscp-transmit af41
      exceed-action set-dscp-transmit af42
      violate-action drop
      conform-color pre-conform exceed-color pre-exceed
    service-policy mark-existing-color
```

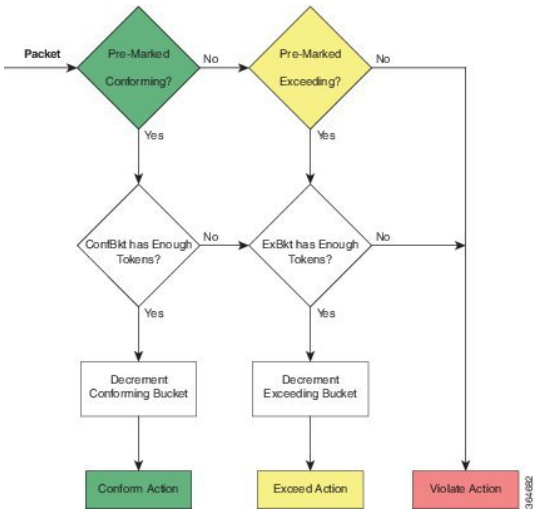
Single-Rate, Color-Aware, Three-Color Policer

The *color-aware mode* of a single-rate, three-color policer extends the standard single-rate, three-color policer.

Similar to the "color-blind" version of this type of policer, we maintain and replenish two distinct token buckets for the "color-aware" mode. The difference stems from how a packet is evaluated against these buckets. Recall that a color-aware policer honors any decision made by a previous router (the current designation of a packet) and ensures that the decision of a previous router is not undone (an exceeding or violating packet can never be promoted to conforming).

The following flowchart illustrates the algorithm used for handling traffic in single-rate, color-aware traffic policing. ConfBkt represents the conforming token bucket and ExBkt the exceeding token bucket.

Figure 80: Single-Rate, Color-Aware, Three-Color Policer



When a packet arrives, the policer uses its color-aware class-maps to determine the pre-existing color of that packet. This color may be conforming (matches the conform-color class-map), exceeding (matches the exceed-color class-map) or violating (matches neither of these class-maps).

If a packet is pre-marked as conforming it might end up as conforming, exceeding or violating. Evaluation proceeds as though the policer was operating in a color-blind mode.

- If the conforming token bucket has enough tokens the packet will take the conform action and the bucket will be decremented by the size of the packet.
- If the conforming token bucket has insufficient tokens but the exceeding token bucket does, the packet will take the exceed action and the exceeding token bucket is decremented by the size of the packet.
- If neither the conforming nor exceeding token bucket has sufficient tokens the packet will take the violate Action.

If a packet is pre-marked as exceeding it can never be promoted to conforming so evaluating the conforming token bucket is unnecessary.

- If the exceeding token bucket has sufficient tokens the packet will take the exceeding action and the bucket is decremented by the size of the packet.

If a packet is pre-marked as violating

- The violating action is taken and either token bucket is unchanged.

Dual-Rate, Color-Aware, Three-Color Policer

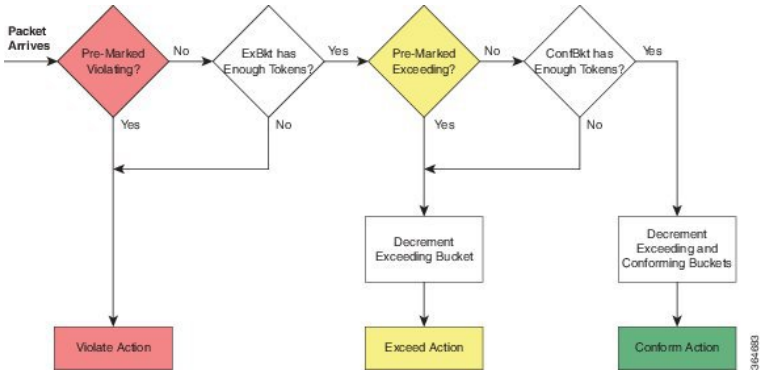
The color-aware mode of a dual-rate, three color policer extends the standard dual-rate, three-color policer.)

Similar to the "color-blind" version of this type of policer, we maintain and replenish two distinct token buckets for the "color-aware" mode. The difference arises from how a packet is evaluated against these buckets. Recall that a color-aware policer honors any decision made by a previous router (the current designation of a packet)

and ensures that the decision of a previous router is not undone (an exceeding or violating packet can never be promoted to conforming).

The following diagram illustrates the algorithm used for handling traffic in dual-rate, color-aware policing. ConfBkt represents the conforming token bucket and ExBkt the exceeding token bucket.

Figure 81: Dual-Rate, Color-Aware, Three-Color Policer



When a packet arrives, the policer uses its color-aware class-maps to determine the pre-existing color of that packet. This color may be conforming (matches the conform-color class-map), exceeding (matches the exceed-color class-map) or violating (matches neither of these class-maps).

If a packet is pre-marked as violating

- we take the violating action and neither bucket is changed (decremented).

If a packet is pre-marked as exceeding

- and the exceeding bucket has sufficient tokens, the packet will remain as exceeding and we decrement the exceeding bucket by the size of the packet.
- and the exceeding bucket has insufficient tokens, the packet will take the violate action and neither bucket is changed.

If a packet is pre-marked as conforming

- and the exceeding bucket has insufficient tokens the packet will take the violate action and neither bucket is changed.
- and the exceeding bucket has sufficient tokens but the conforming bucket does not the packet will take the exceed action and we decrement the exceeding bucket by the size of the packet.
- and both the exceeding and conforming buckets have sufficient tokens the conform action will be taken and we decrement both buckets by the size of the packet.

Hierarchical Policy Containing Policers

In hierarchical traffic policing, we introduced hierarchical policies as a way to offer more granular control over traffic classes and to have some QoS actions operate on the aggregate of a number of those classes.

The ASR 1000 Series supports at most three levels in a hierarchical policy and the policing feature (one particular QoS action) can be configured at any level of that policy.

When describing hierarchical policies we often use different language to describe the distinct levels within that hierarchy (e.g., Top/Middle/Bottom, Parent/Child/Grandchild, Root/Leaf, Child/Parent/Grandparent). Because this can lead to ambiguity, we will always refer to the levels as Parent/Child/Grandchild where meanings are defined as follows:

Parent policy is a policy-map that will be attached to an interface using the **service-policy** command.
Child policy is a policy embedded directly in a class of the parent policy (using the **service-policy** command within a class).
Grandchild policy is a policy embedded directly in a class of the child policy.

Occasionally, we will refer to a policy's child or parent. They represent more relative terms (e.g., the parent of the *grandchild policy* references the child policy when we communicate in absolute terms).

Ingress Hierarchical Policy Containing only Policers

One of the simplest and perhaps most typical use of policers in hierarchical policies is an ingress policy containing only policers. We have already described how a policer is often used for network admission control (defined inWhy Traffic Policing, on page 587). Replacing a simple policer with hierarchical policers allows the network operator to not only set an aggregate rate for network admission but also to specify rates for the individual classes of traffic that will be carried over the network.

For example, consider the following policy:

```
policy-map child
  class voice
    police cir percent 10 bc 5 ms
 !
policy-map parent
  class class-default
    police cir 50000000
    service-policy child
```

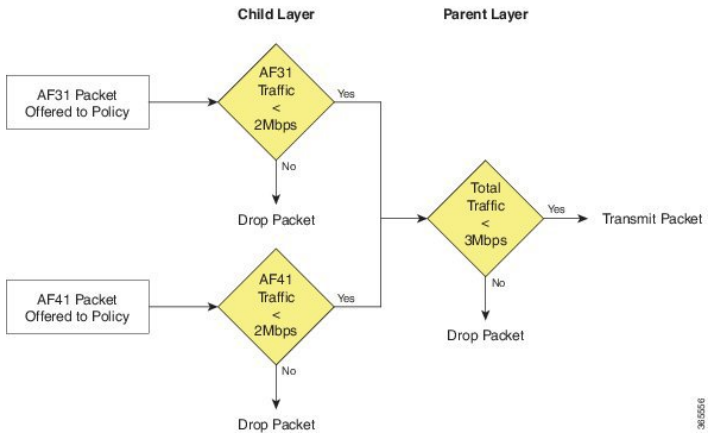
The policy-map parent, which is attached to the interface, defines the aggregate network admission rate (or service rate) for a customer so connected. In this example, the customer has contracted for 50 Mbps of network service. Within that network rate, the child policy limits individual classes of traffic. For example, the voice class specifies that traffic arriving at a rate exceeding 5 Mbps (10% of the parent) would simply be dropped.

Hierarchical Policers Order of Operation

On the ASR 1000 Series Appregation Services Router, we evaluate hierarchical policers for the child first then the parent. Although this scheme differs from IOS classic, it provides a much more meaningful construct. The following example should clarify this notion:

```
policy-map child
  class AF41
    police 2m
  class AF31
    police 2m
!
policy-map parent
  class AF41_or_AF31
    police 3m
    service-policy child
```

Figure 82: Hierarchical Policing



If a packet arrives with a marking of AF31 it must first pass through the AF31 policer in the child policy, which allows such packets to pass through up to a rate of 2 Mbps. The packet must then pass through the parent policer, which observes both AF31 and AF41 traffic.

The combined rate of AF31 and AF41 traffic from the child policy could be up to 4 Mbps as each has a 2 Mbps policer configured. Although a packet passed through the child policer, it may be dropped by the parent policer if the rate arriving at that policer is above the configured 3 Mbps rate.

When policers are used in an egress policy-map with scheduling semantics (bandwidth/shape/priority) all policers will be evaluated before a packet is enqueued. Furthermore, a policer in the parent level would be enforced before a shape value in the child level (scheduling happens).

Percent-Based Policer in Hierarchical Polices

If a percent-based policer is used in the parent level of a policy-map the meaning of the percent is fairly intuitive - it is a percent of the bandwidth available in the interface where the policy-map is attached. When we use a percent-based policer in the child or grandchild level, the meaning can be a bit more ambiguous.

If the percent-based policer is configured in the child level it examines the parent level class to assess whether the bandwidth of that class has been constrained by either a shaper or policer. If so, the child policer CIR is a percent of the shape or police rate configured in the parent level. If not, the percent is interpreted as percent of the bandwidth available in the interface where the policy-map is attached.

If the percent-based policer is configured in the grandchild level it first looks at the child level class for a shaper or policer. If it finds one, it uses that rate in the child level. If none exists, the grandchild policer looks at its class in the parent level. It either finds a rate there or uses the rate of the interface to which the policy is attached.

If the percent policer is configured in the grandchild level and a rate-limiting feature (e.g., shaper or policer) is configured in both the child and parent levels, the grandchild always uses the rate configured in the child level. This is crucial as the sum of shapers or policers at any level in a policy can be greater than the physical bandwidth available.

If both a shaper and a policer are configured in the parent of a class with a percent-based policer, the percent-based policer is based on the lower rate configured (shaper or policer).

The following hierarchical policy-map illustrates these considerations:

```
policy-map grandchild
  class AF11
    police cir percent 60
```

```
  class AF12
    police cir percent 40
!
policy-map child
  class AF1
    bandwidth percent 50
    service-policy grandchild
!
policy-map parent
  class class-default
    shape average 50000000
    service-policy child
```

The policers in the grandchild policy are percent-based policers. They defer to their parent class (class AF1 in the parent policy) for rate-limiting features. Because none exist here, the policers "step up" to the parent class (**class class-default**, in the parent policy).

There, they find a shaper that limits the throughput to 50Mbps. So, the policer in class AF11 would be configured with a CIR of 30 Mbps (60% of 50Mbps); the policer in class AF12, a CIR of 20 Mbps.

Verifying the Configuration and Operation of the Policing Feature

As with all MQC QoS features, you have three ways to verify the configuration and performance of the policing feature:

- **show policy-map** *policy-name*

  Displays the user-entered configuration. Analogous to contents of the running configuration on the router but displays default values and actions not explicitly called out in the configuration.

- **show policy-map interface** *interface-name*

  Displays statistics for all features within that policy-map. Primary means of verifying that a QoS policy is operating as expected..

- **show platform hardware qfp active feature qos interface** *interface-name*

  Displays real-time information from the dataplane. Shows the exact rates and burst sizes that are programmed in hardware.

Example 1: show policy-map *policy-name* Command

If we configure the policy-map simple_policer as follows:

```
policy-map simple_policer
  class AF1
    police cir 20000000
```

show policy-map command output looks like this:

```
show policy-map simple_policer

Policy Map simple_policer
  Class AF1
```

```
police cir 20000000 bc 625000
  conform-action transmit
  exceed-action drop
```

Besides the explicit conforming burst and conform (or exceed) actions, notice the lack of statistics or interface information. We merely define a policy, an action applicable to multiple interfaces.

Example 2: show policy-map interface *interface-name* Command

Here is a sample output of the **show policy-map interface** command for an instance of a policy-map attached to a particular interface:

```
show policy-map interface GigabitEthernet1/0/0

GigabitEthernet1/0/0

 Service-policy input: simpler_policer

  Class-map: AF1 (match-any)                              --+
   1000 packets, 1496000 bytes                              |Classification
   5 minute offered rate 0000bps, drop rate 0000bps         |Section
   Match: :dscp af11 (10) af12 (12) af13 (14)               |
   police:                                                 --+
     cir 20000000 bps, bc 625000 bytes                      |
    conformed 447 packets, 668712 bytes; actions:           |Policing
     transmit                                               |Section
    exceeded 553 packets, 827288 bytes; actions:            |
     drop                                                   |
    conformed 0000 bps, exceeded 0000 bps                  --+

  Class-map: class-default (match-any)
    0 packets, 0 bytes
    5 minute offered rate 0000 bps, drop rate 0000 bps
    Match: any
```

The organization of the output reflects the policy-map definition combined with a hierarchical output that represents the policy-map and class hierarchy. Within each class, a classification section displays the classification counters (statistics of packets that were determined to belong to this class) and the classification criteria (a summary of the class-map that defines what packets belong to this class). Following the classification section, you observe a block that represents each QoS action configured within that class. Because policing is the sole action in this example, only a block of policing statistics displays.

The following table summarizes the meaning of different fields in the **show** command output.

SUMMARY STEPS

1. As you will observe, the output provides a summary of the configuration along with statistics. A router uses the statistics (over time) to calculate rates and display them. Rates in these formulations represent a *decayed average (rate)*. The frequency (default, 300 seconds) of the calculation hinges on the load-interval for that interface. Statistics in the **show policy-map interface** output persist until you issue a **clear counters** command. The dataplane updates the statistics every 10 seconds.
2. Notice that Classification and Policer Action statistics arise from different entities in the dataplane. Consequently, they might update at slightly different times (briefly, the action counters might exceed the classification counters).

DETAILED STEPS

	Command or Action	Purpose
Step 1	As you will observe, the output provides a summary of the configuration along with statistics. A router uses the statistics (over time) to calculate rates and display them. Rates in these formulations represent a *decayed average (rate)*. The frequency (default, 300 seconds) of the calculation hinges on the load-interval for that interface. Statistics in the **show policy-map interface** output persist until you issue a **clear counters** command. The dataplane updates the statistics every 10 seconds.	
Step 2	Notice that Classification and Policer Action statistics arise from different entities in the dataplane. Consequently, they might update at slightly different times (briefly, the action counters might exceed the classification counters).	

Example 3: show platform hardware qfp active feature qos interface Command

This command should only be necessary if you believe the router is configured correctly but is behaving unexpectedly. Viewing information directly from the dataplane can help you assess whether any quantization of rates or burst parameters were necessary to accommodate the hardware.

The following example corresponds to the **show policy-map interface** output from the previous example:

```
show platform hardware qfp active feature qos interface g1/0/0

Interface: GigabitEthernet1/0/0, QFP interface: 9
  Direction: Input
    Hierarchy level: 0
    Policy name: simple_policer
      Class name: AF1, Policy name: simple_policer
        Police:
            cir:  20000000 bps,  bc: 638976 bytes
            pir: 0 bps,  be: 638976 bytes
            rate mode: Single Rate Mode
            conformed: 447 packets, 668712 bytes; actions:
              transmit
            exceeded: 427 packets, 638792 bytes; actions:
              drop
            violated: 126 packets, 188496 bytes; actions:
              drop
            color aware: No
            green_qos_group: 0, yellow_qos_group: 0
      Class name: class-default, Policy name: simple_policer
```

If you understand the output of the previous two commands (Example 1 and Example 2), this output should be pretty self-explanatory. However, you should be aware of the following points related to using this command:

Although we configured a single-rate two-color policer, the output of the dataplane command corresponds to a single-rate, three-color policer. The hardware always operates in a three-color mode. To achieve two-color functionality it simply matches the Violate and Exceed actions. When we push statistics to the control plane, IOS aggregates the Exceed and Violate statistics to generate the expected appearance of a two-color policer.

Statistics in the dataplane are transitory. Every 10 seconds the dataplane pushes statistics to IOS and then clears its local counters. Essentially, all statistics observed through the dataplane command are counts of what transpired since the last push. This means that dataplane commands help you view hardware behavior in real time. For meaningful (persistent) statistics, however, you should always use the regular IOS **show policy-map interface** command.

Configuration Examples for QoS Packet Policing

Example 1: Simple Network Admission Control

In its simplest form a policer can be used to rate-limit all traffic entering an interface (and thereby a network). We assume that the network sending the traffic will "shape" what exits its egress interface and only send traffic that will conform to the contracted rate. We can use *egress scheduling* on the senders' network to apportion the contracted rate to different classes of traffic.

With this simplest example of policing no classification is required as the policer is intended to cap all traffic. We will consider all traffic as belonging to class-default in the absence of any user-defined classes.

In the following example, we have a GigabitEthernet connection but the customer has only contracted for a 100 Mbps service rate. The configuration could look something like this:

```
policy-map ingress_cap_all_100m
  class class-default
    police cir 100000000
!
interface GigabitEthernet1/0/0
  service-policy ingress_cap_all_100m
```

Example 2: Network Admission Control - Hierarchical Policers

In Example 1: Simple Network Admission Control, on page 611 we policed all traffic to a contracted service-rate, assuming that the sender would apportion bandwidth within that contracted rate. However, we may not always trust the sender to limit the traffic within an individual class. For example, say we offer a priority service (traffic guaranteed low latency through the network) but charge the user for different levels of priority access. By simply applying the simple policer in Example 1 we could not guarantee that the sender doesn't forward us more priority traffic than contracted. We can expand the example to enforce also a cap on an individual class of traffic.

In the following example, we limit the total admission to 100 Mbps AND ensure that voice traffic caps at 5 Mbps of traffic:

```
class-map match-all voice
  match dscp ef
!
! child policy to enforce 5Mbps Voice Traffic
!
policy-map ingress_police_child
  class voice
    police cir percent 5 bc 5 ms
!
policy-map police_ingress_parent
  class class-default
    police cir 100000000
  service-policy ingress_police_child
```

```
!
interface GigabitEthernet1/0/0
  service-policy in police_ingress_parent
```

Example 3: Network Admission Control - Color-Aware Policer

In Example 2: Network Admission Control - Hierarchical Policers, on page 611, we introduced the scheme of capping a particular class of traffic within the contracted service-rate. This scheme hinges on customer shaping of traffic to the service-rate.

If we received traffic at a rate exceeding the parent policer (the contracted service-rate), no guarantee exists that it would not drop some of the voice traffic admitted by the child policer. To ensure that any traffic admitted by the child policer is also admitted by the parent, you could employ a color-aware policer for the parent policer.

The following example shows how complex outcomes can be achieved with combinations of policers. Here, we mark all voice traffic admitted by the child policer as Green (qos-group1) and all traffic other than voice as Yellow (qos-group2). The parent policer is configured with a CIR that ensures that we forward all the Green traffic and a PIR that ensures that we enforce the contracted service-rate:

```
class-map match-all voice
  match dscp ef
!
!child policy to enforce 5Mbps Voice Traffic
!
policy-map ingress_police_child
  class voice
    !conforming voice marked Green, Excess Dropped
    police cir 5m bc 3125 conform-action set-qos-transmit 1
  class class-default
   !all traffic other than voice marked Yellow
   set qos-group2
!
class maps needed for color-aware policer
!
class-map policer-green
  match qos-group1
class-map policer-yellow
  match qos-group2
!
!parent policy to enforce 100Mbps service rate
!
policy-map ingress_police_parent
  class class-default
    police cir 5m bc 3125 pir 100m be 625000
      conform-action transmit
      exceed-action transmit
      violate-action drop
      conform-color policer-green exceed-color policer-yellow
    service-policy ingress_police_child
!
interface GigabitEthernet1/0/0
  service-policy in ingress_police_parent
```

Command Reference

police

As discussed within chapter there are three variants of the police command that achieve the same result, namely:

[no] police *cir* [**bc** *conform-burst*] [**pir** *pir*] [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]]

[no police cir *cir* [**bc** *conform-burst*] [**pir** *pir*] [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]]

[no] police rate*cir* [**bc** *conform-burst*] [**pir** *pir*] [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]]

Henceforth we shall denote this as:

[no] police [cir | rate]*cir* [**bc** *conform-burst*] [**pir** *pir*] [**be** *peak-burst*] [**conform-action** *action* [**exceed-action** *action* [**violate-action** *action*]]]

We have already seen how the same command may be used to configure different types of policers.

Rather than present a single CLI which would be confusing and option combinations which might not be correct, we will present a subset of the options depending on the policer type you wish to configure.

Single-Rate, Two-Color Policer

This policer type can be expressed on a single line if you require only one action per designated conformance level:

[no] police [cir | rate]*cir***[percent** *percent***][[bc]** *conform-burst* **[ms]**] [**account** *options*]**[conform-action** *action* **[exceed-action** *action***]]**

Multiple lines (using sub-modes) are necessary if we requiremore than one action:

[no] police [cir | rate]*cir***[percent** *percent***][[bc]** *conform-burst* **[ms]**][**account** *options*] **<return>**
[no] conform-action *action* **<return>**
[no] conform-action *action* **<return>**
[no] exceed-action *action* **<return>**
[no] exceed-action *action* **<return>**

Single-Rate, Three-Color Policer

This policer type can be expressed on a single line if you require only one action per designated conformance level:

[no] police [cir | rate]*cir***[percent** *percent***][[bc]** *conform-burst* **[ms]**] [**[be]** *exceed-burst* **[ms]**][**account** *options*]**conform-action** *action* **exceed-action** *action* **[violate-action** *action***]**

Multiple lines (using sub-modes) are necessary if we require more than one action:

[no] police [cir | rate]*cir***[percent** *percent***][[bc]** *conform-burst* **[ms]**] [**[be]** *exceed-burst* **[ms]**][**account** *options*] **<return>**
[no] conform-action *action* **<return>**
[no] conform-action *action* **<return>**
[no] exceed-action *action* **<return>**
[no] exceed-action *action* **<return>**
[no] violate-action *action* **<return>**
[no] violate-action *action* **<return>**

Dual-Rate, Three Color Policer

This policer type can be expressed on a single line if you require only one action per designated conformance level:

[no] police [cir | rate]*cir***[percent** *percent***][[bc]** *conform-burst* **[ms]**] **[pir]** *peak-rate* **[ms][[be]** *exceed-burst* **[ms]**][**account** *options*]**conform-action** *action* **exceed-action** *action* **[violate-action** *action***]**

Multiple lines (using sub-modes) are necessary if we require more than one action:

[no] police [cir | rate]*cir***[percent** *percent***][[bc]** *conform-burst* **[ms]**] [**[pir]** *peak-rate* **[ms]**][**[be]** *exceed-burst* **[ms]**][**account** *options*] **<return>**
[no] conform-action *action* **<return>**
[no] conform-action *action* **<return>**
[no] exceed-action *action* **<return>**
[no] exceed-action *action* **<return>**
[no] violate-action *action* **<return>**
[no] violate-action *action* **<return>**

Single-Rate, Three-Color, Color-Aware Policer

[no] police [cir | rate]*cir***[percent** *percent***][[bc]** *conform-burst* **[ms]**] [**[be]** *exceed-burst* **[ms]**][**account** *options*] **<return>**
[no] conform-action *action* **<return>**
[no] conform-action *action* **<return>**
[no] exceed-action *action* **<return>**
[no] exceed-action *action* **<return>**
[no] violate-action *action* **<return>**
[no] conform-color *conform-color* **exceed-color** *exceed-color***<return>**

Dual-Rate, Three-Color, Color-Aware Policer

[no] police [cir | rate]*cir***[percent** *percent***][[bc]** *conform-burst* **[ms]**][**[pir]** *peak-rate* **[ms]**] [**[be]** *exceed-burst* **[ms]**][**account** *options*] **<return>**
[no] conform-action *action* **<return>**
[no] conform-action *action* **<return>**
[no] exceed-action *action* **<return>**
[no] exceed-action *action* **<return>**
[no] violate-action *action* **<return>**
[no] conform-color *conform-color* **exceed-color** *exceed-color***<return>**

police Command Default and Modes; Keyword/Argument Descriptions

Command Default

Disabled

Command Modes

Policy-map class configuration (config-pmap-c) when specifying a single action to be applied to a marked packet

Policy-map class police configuration (config-pmap-c-police) submode when specifying multiple actions to be applied to a marked packet

Syntax Description

The following table list the keywords/arguments for the **police** command and their purpose.

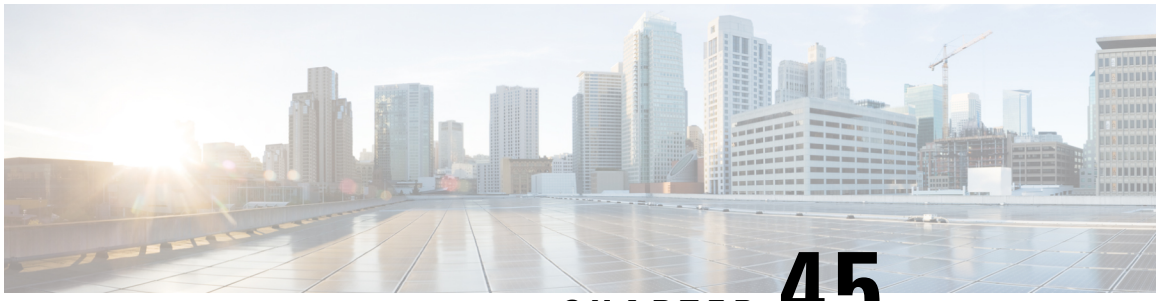
Keyword/Argument	Definition
bc	Specifies the Conforming Burst Size(Bc).
be	Specifies the Exceeding Burst Size(Be).
cir	Specifies the Committed Information Rate(CIR).
Conform-Action	Specifies the action to take on traffic that is determined to be "conforming."
Exceed-Action	Specifies the action to take on traffic that is determined to be "exceeding."
pir	Specifies the Peak Information Rate (PIR).
Violate-Action	Specifies the action to take on traffic that is determined to be "violating."

The following table lists the options for the Account keyword.

Table 51: Account keyword options

Option	Purpose
qinq	Specifies queue-in-queue encapsulation as the BRAS-DSLAM encapsulation type
dot1q	Specifies IEEE 802.1Q VLAN encapsulation as the BRAS-DSLAM encapsulation type
aal5	Specifies the ATM Adaptation Layer 5 that supports connection-oriented variable bit rate (VBR) services
aal3	Specifies the ATM Adaptation Layer 3 that supports both connectionless and connection-oriented links

Option	Purpose
subscriber-encapsulation	Specifies the encapsulation type at the subscriber line
user-defined	Indicates that the router is to use the offset value that you specify when adjusting policing length
offset	Specifies the number of bytes that the router is to use when calculating overhead. Valid values are from -63 to 63 bytes
atm	Applies the ATM cell tax in the ATM overhead calculation



CHAPTER 45

Queue Limits and WRED

About

On Cisco IOS XE devices, we dedicate memory to store packets that are queued in egress interface or QoS queues. The memory is treated as a global pool available to all interfaces rather than as carved or owned by individual interfaces.

A *queue-limit* caps the depth of a particular queue and serves two purposes. First, they constrain how much of the available packet memory an individual queue may consume. This ensures that other interfaces or queues also have fair access to this shared resource. Second, they constrain how much data we store if a queue is congested, thereby capping the latency applications in that queue will experience.

When a packet is ready for enqueuing we check the current depth of that queue and the configured queue-limit. If the former has already achieved the latter then the packet is dropped (tail drop).

Queue Limits

The packet memory of an ASR 1000 Series Aggregation Services Router (heretofore the ASR 1000 Series Router) is a shared resource. We do not allocate individual interfaces and queues a share of this memory. Rather they represent a global-pool available to all queues on a first come, first serve basis.

To control how much data an individual queue may store in the shared packet memory, we use *queue-limit*, a per-queue configurable value. It serves two purposes. First, it limits the latency for a packet arriving to a nearly full queue - at some point it is better to drop than to deliver packets so slowly that they are useless at the receiver. Second, it ensures that a single interface can’t put so many packets into the shared memory that it starves other interfaces.

We manage the shared memory very efficiently: Instead of carving pools of buffers into predetermined sizes, the hardware manages blocks of memory (32 byte blocks on the original QFP) and assigns the minimum number of blocks needed to store a packet.

The following table shows how the amount of packet memory and the maximum configurable number of queues vary by platform:

ESP (Embedded Services Processors) Router Hardware	Packet Memory	Maximum Queues
ASR1001	64 MB	16,000
ASR1001-X	512 MB	16,000
ASR1002-F	64 MB	64,000
ASR1002-X	512 MB	116,000
ESP5	64 MB	64,000
ESP10	128 MB	128,000
ESP20	256 MB	128,000
ESP40	256 MB	128,000
ESP100	1 GB (two 512-MB)	232,000*
ESP200	2 GB (four 512-MB)	464,000*

For ESP100 and ESP200, physical ports are associated with a particular QFP (Quantum Flow Processor) complex on the ESP card. To maximally-use all queues, you must distributed them among different slots and SPAs in the chassis.

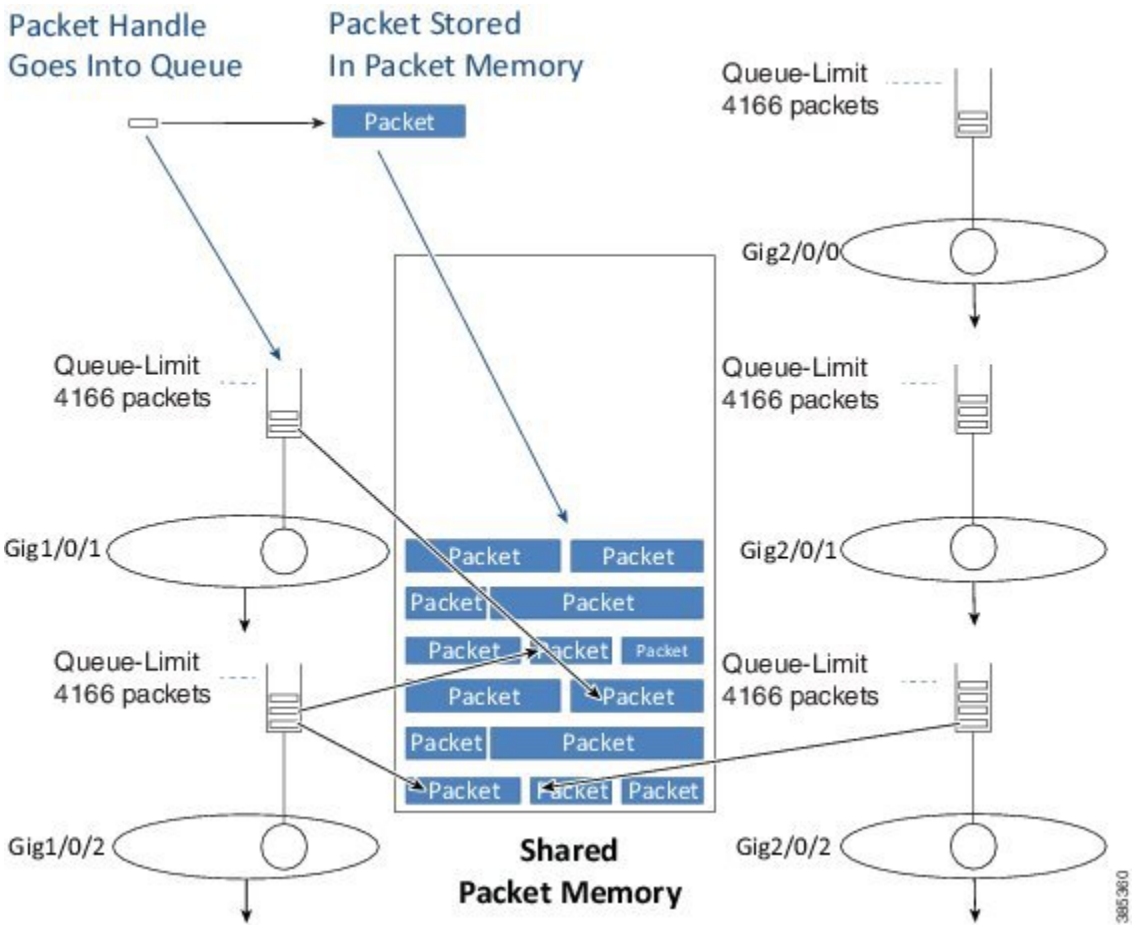
The amount of packet memory in an ASR 1000 Series Router is driven by a number of factors: cost, technology availability and what makes sense. When QFP was first released few choices for memory technologies were available to handle the 10s of gigabits per second of reads (and writes) required for packet memory. Even when memory could handle the speed requirements, options for size were limited and module cost was extremely high; we could have designed a system with more memory but it would have been prohibitively expensive with no real upside.

Beyond simply the number of queues supported, you must also consider the rate at which packets can ingress and egress the system. For example, looking at the ESP10, you could say that 128MB and 128,000 queues translate into 1KB of memory per queue. This is pretty meaningless if you never have all 128K queues congested simultaneously.

Let's view the size in another way: An ESP10 can transmit or receive data at a max rate of 10Gbps. At this speed, 128 MB of memory provides over 100mS of buffering which is quite reasonable.

From above it should now be evident that we expect to oversubscribe the sum of all queue-limits in the system.

Figure 83: Queue limits

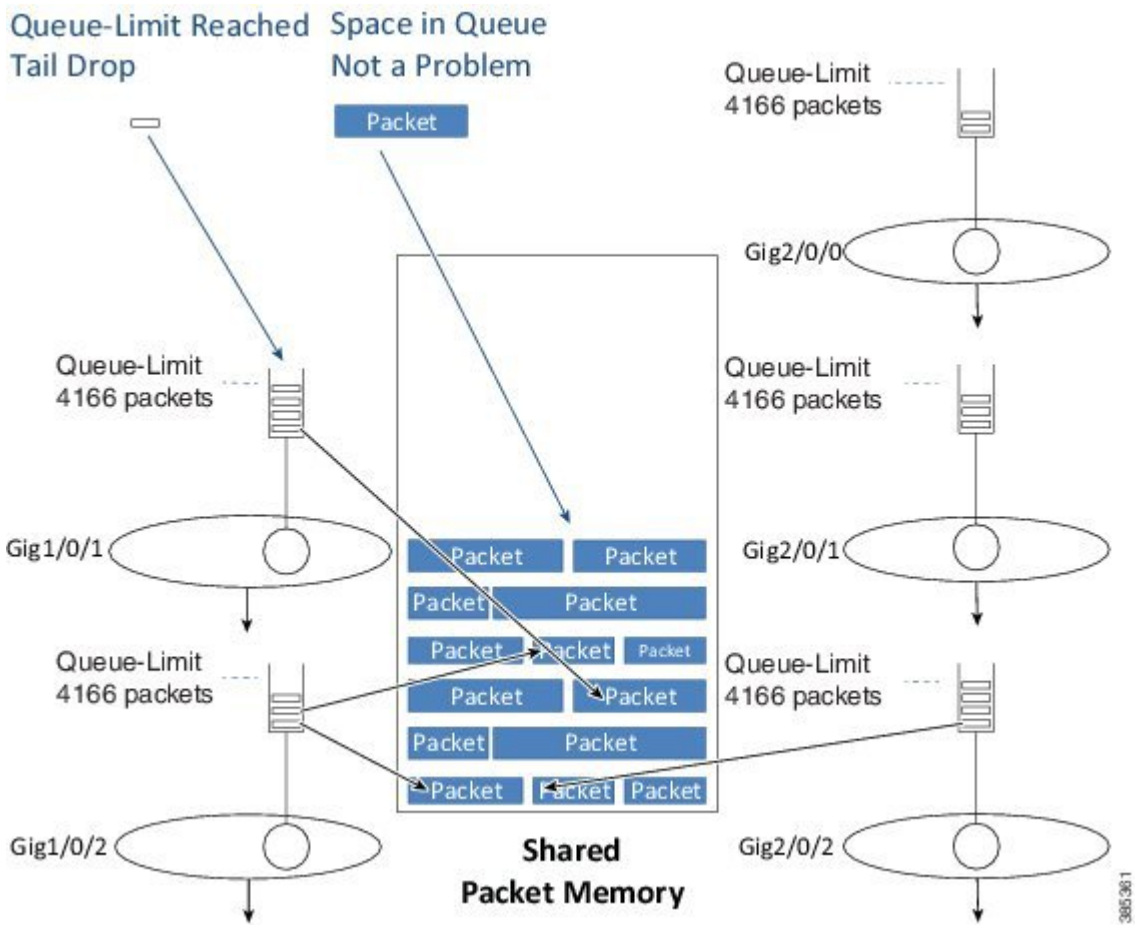


After we determine the egress interface for a packet, we will know the queue information for that interface. Into that queue we place a small packet handle, which contains the scheduling length for the packet, and a pointer to where the packet is stored in the shared packet memory. We store the actual packet itself in shared packet memory.

Tail Drop

When we enqueue a packet we first examine the configured queue-limit as well as how much data that interface currently has buffered (the *instantaneous queue depth*).

Figure 84: Tail Drop



If the queue depth is already at the preconfigured limit, we will drop the packet and record a **tail drop**.

If QoS is not configured, you can view the drop in the output of the **show interface** command.

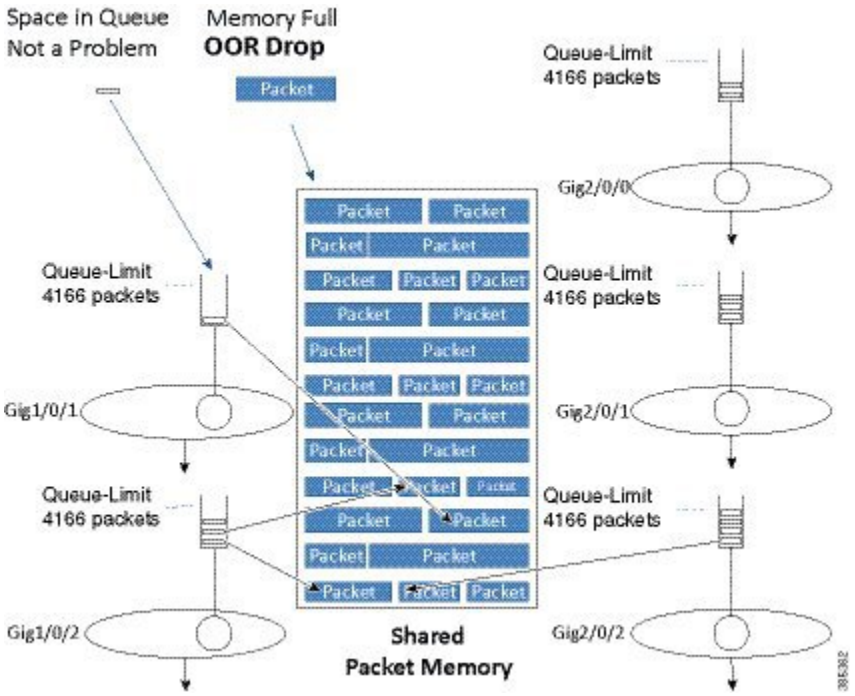
If QoS is configured, you can view the drop in the class output of the **show policy-map interface**.

As the diagram illustrates, a tail drop does not mean that no memory exists to store the packet rather it means that a queue has already reached its individual limit on how much data it can store.

Out of Resources Drop

Another scenario is possible on enqueue if the queue has not yet reached its individual queue-limit but the shared-packet memory may be full. If so and no place exists to store the packet, we must drop it. This drop would be recorded as a No Buffer drop and reported to the syslog as an *Out Of Resources (OOR) condition*.

Figure 85: ORR Drop

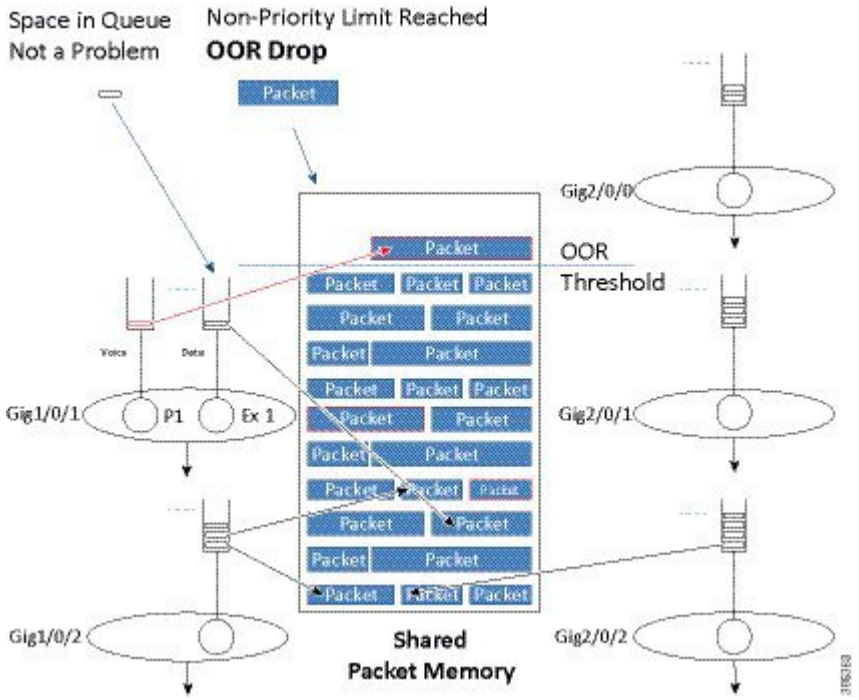


If OOR drops are only seen very occasionally you can ignore them. However, if this is a regular condition then you should review queue-limits to see whether you are allowing an individual queue or interface to consume too much memory. To avoid this situation, you might need to lower the queue-limit for one or more queues.

Memory Reserved for Priority Packets

The description of packet memory being 100% full is not really accurate. We know that some packets (those from priority classes and pak_priority packets) are more important than others and we want to ensure that we always have space in memory to store these important packets so. To do this, we limit packets from normal data queues to 85% of the total packet memory.

Figure 86: Memory Reserved for Priority Packets



The diagram above shows how we treat priority packets and data packets differently. In this scenario, 85% of the packet memory has been consumed. If a normal data packet arrives it is dropped because the OOR threshold has been reached. However, if a priority packet were to arrive it would still be enqueued as there is physical space available.

Please note that we are not restricting priority packets to a small portion of the memory. Instead, we are dropping non-priority packets when memory is nearly full.

Vital Threshold

We also provide a second level of protection that will drop all user traffic, including priority packets, when the memory utilization exceeds 98%. We term this the *vital threshold* and it ensures that we can enqueue internal control packets, which are inband packets that may need to travel between different control processors in the system. As priority packets are usually forwarded when enqueued, exceeding a 98% threshold is unexpected.

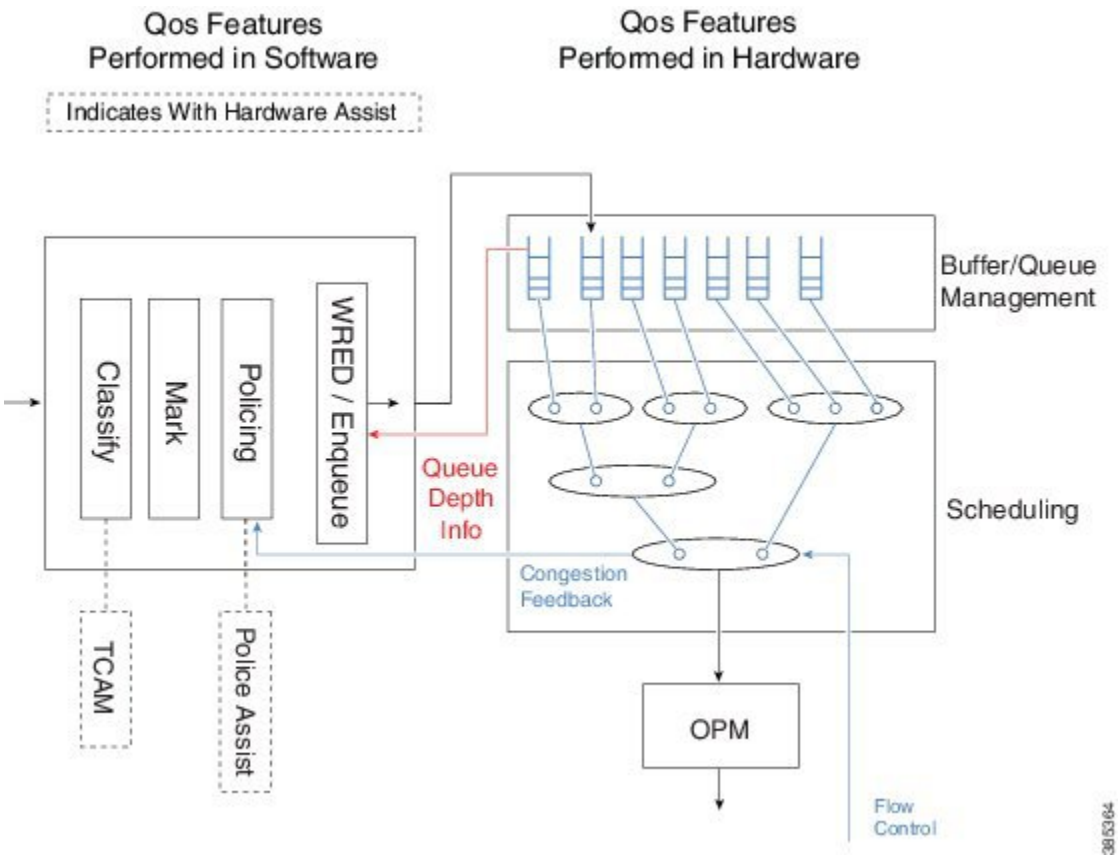
You can see the amount of memory in a system and the realtime-utilization of that memory using the **show platform hardware qfp active bqs 0 packet-buffer utilization** command.

```
show platform hardware qfp active bqs 0 packet-buffer utilization
Packet buffer memory utilization details:
  Total:     256.00 MB
  Used :    2003.00 KB
  Free :     254.04 MB

  Utilization:    0 %

  Threshold Values:
    Out of Memory (OOM)      :     255.96 MB, Status: False
    Vital (> 98%)            :     253.44 MB, Status: False
```

```
Out of Resource (OOR)    :     217.60 MB, Status: False
```

On the ASR 1000 Series Aggregation Services Router, all queuing, scheduling and packet memory management is performed by dedicated hardware. When we enqueue a packet we are passing control from software to hardware. As the hardware, specifically the BQS (Buffering, Queuing and Scheduling) subsystem, manages the memory it monitors how much data each queue is currently storing in packet memory. When we are ready to enqueue a packet we query the hardware for current status. The hardware will report an instantaneous and an average queue depth for that queue. Software will then determine whether to continue with the queue or drop the packet (and report it). Tail drop decisions are made using the instantaneous queue depth reported by hardware. Instead, WRED uses the average queue depth.

Figure 87: Vital Threshold



Packet Mode vs Byte Mode

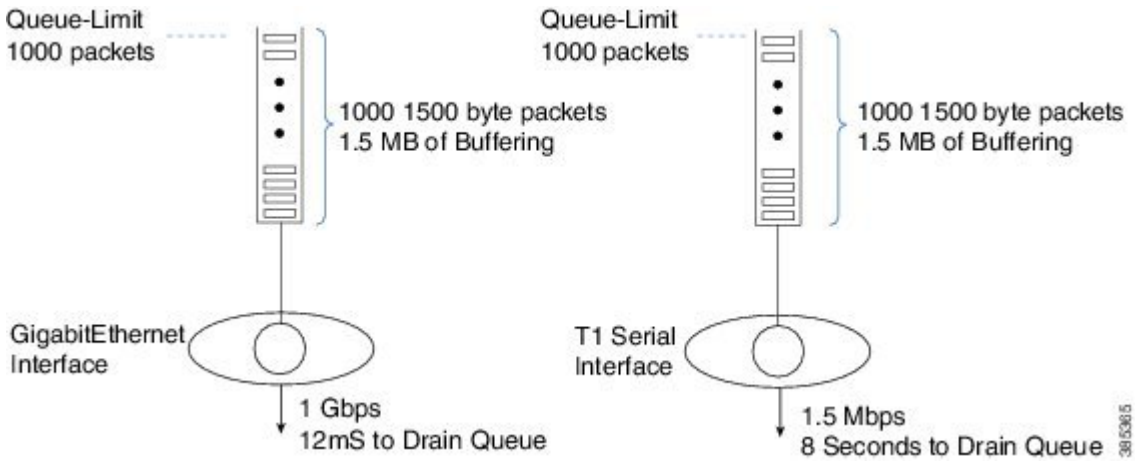
The hardware may operate in one of two modes; packet mode or byte mode. When reporting the instantaneous and average queue depth it will report those values in packets or in bytes but not in both. At the time a queue is created the mode is set and can't be changed unless you remove and reattach the policy-map.

The diagram above shows how some QoS features are performed in software and others in hardware. The enqueue is really on the boundary of the two. Software will receive Queue Depth Information from the hardware and then decide whether to drop the packet or to move it to packet memory and add a packet handle to the queue. WRED is a more advanced form of drop decision and will be covered later in the chapter.

Default Queue-Limits

The following diagram shows the need for *variable queue limits*.

Figure 88: Variable Queue Limits



The queue on the left is serviced at 1 Gbps. If 1000 1,500 byte packets were waiting transmission it would take 12 mS to drain the queue. This means a packet arriving to an almost full queue could be delayed by 12 mS while waiting its turn to be forwarded.

The schedule on the right represents a T1 interface, a considerably slower interface operating at approx. 1.5 Mbps. If the same 1000 packets were waiting transmission through a T1 interface it would take 8 seconds to drain the queue. Obviously most users (and applications) would be disappointed with such a delay.

The diagram highlights the second role of queue-limits mentioned above - constraining the latency for applications in a queue.

How we determine the default queue mode and queue-limit will vary depending on whether or not QoS is configured.

Note that we select default queue-limits to be appropriate for as many users as possible but that does not mean they are always the best choice. We cannot know how many physical and logical interfaces will be configured in a system, how bursty traffic will be in any queue, the latency requirements of applications in a queue, etc. The defaults are a good starting point but it is not unusual to tune queue-limits further.

When Qos is not Configured

In the scheduling chapter we have seen that when no QoS is configured all packets go through a single FIFO that we refer to as the Interface Default Queue. The queue-limit for the interface default queue is configured in bytes and is calculated as 50mS worth of buffering based on the interface speed (ESP-40 is an exception where 25mS is used).

As an example consider a GigabitEthernet interface. The interface speed is 1 Gbps but with internal overdrive we send at 1.05 Gbps:

50mS worth of buffering in bytes would be: 1.05 Gbps / 8 bits per byte * .05 seconds = 6,562,500 bytes

You can use the **show platform hardware qfp active infrastructure bqs queue output default interface gig1/0/0 | inc qlimit** command to view the queue-limit for an interface default queue.

When QoS is Configured

Note The default mode for any queue created using the MQC CLI is packet. (This is an historical artifact rather than an admission that packet mode is superior.)

Calculating queue-limit depends on a number of factors:

If the queue is a *priority queue* the default queue-limit is 512 packets. Yes. This is a large limit but we assume that these values are meaningless. Because queue admission control ensures that packets are enqueued at a rate lower than they will be transmitted, a priority queue should always be nearly empty. Thus, we can set the queue-limit arbitrarily large and use it across all interface speeds.

For a *bandwidth queue* we target a maximum of 50mS worth of data buffered but make an exception for low speed queues where this might represent a very small amount of data. To calculate how much data would be transmitted (in 50mS) we need to know the service speed. For an interface default queue (recall, the only game in town for scenarios without QoS) this is simple - a single queue 'owns' the entire bandwidth of the interface. When QoS is configured, the picture gets murky.

First, we need to introduce the concept of *visible bandwidth* , a value ascertained from the configuration that captures the service rate of a queue without accounting for the offered load. The table below shows how the visible bandwidth depends on the commands used:

Table 52: Representation of Visible Bandwidth Depends on the Commands used

Commands	Visible Bandwidth
shape	shape rate
bandwidth	bandwidth rate
shape and bandwidth	bandwidth rate
bandwidth remaining	Inherited directly from the parent. • If the policy-map is attached to a physical interface the value inherited would be the interface speed. • If the policy is a child policy with a parent shaper the visible bandwidth would be the parent shape rate.

Second, we need the Maximum Transmission Unit (MTU) for the interface where the policy is attached. As we are configuring a queue-limit in packets (recall that this is the default) and want to limit the potential latency, we look at a worst case scenario where a queue is full of MTU-size packets (view the MTU in the output of the **show interface** command).

Given the visible bandwidth, the MTU, and a maximum of 50mS worth of buffered data, we can calculate a queue-limit as follows:

queue-limit = (visible bandwidth / 8 bits) * 50mS / MTU

Let's consider a queue shaped to 100 Mbps on a GigabitEthernet Interface. The visible bandwidth would be the shape rate (100 Mbps) and the MTU would be 1500 bytes (what you expect on an Ethernet type interface):

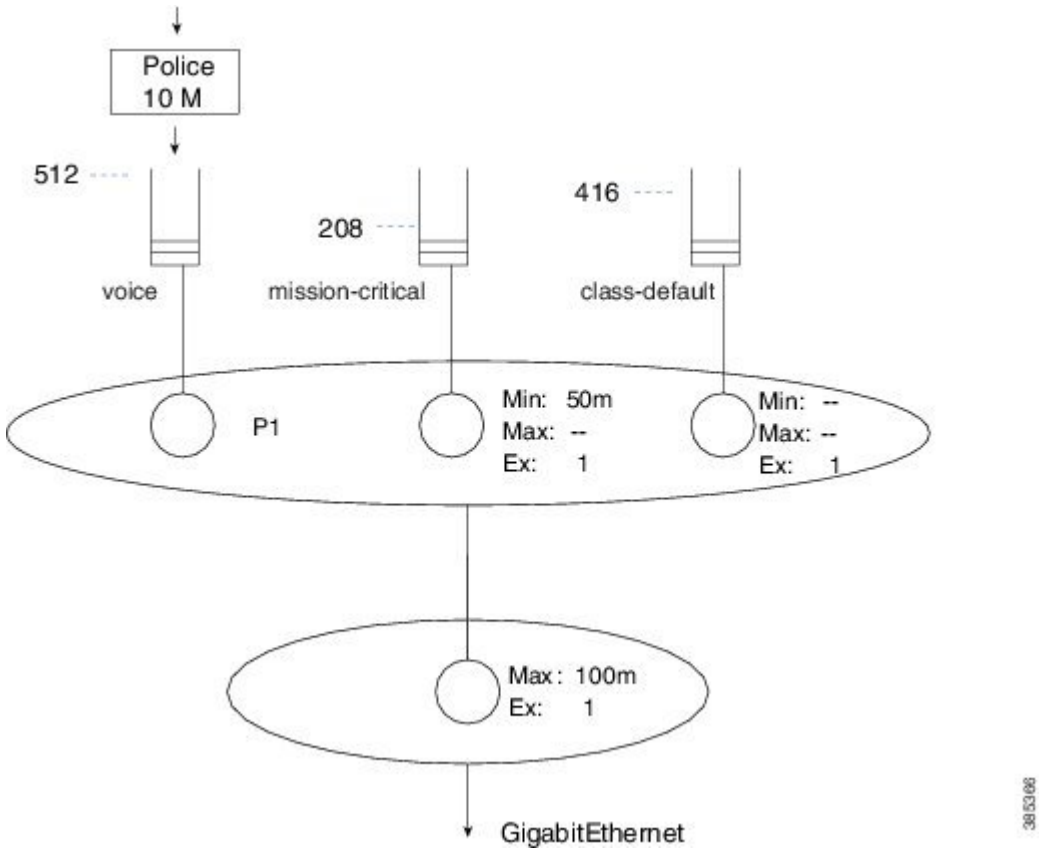
queue-limit = 100 Mbps / 8 bits * .05 sec / 1500 bytes = 416 packets

As mentioned, we make an exception for low speed queues. If the calculated queue-limit is less than 64 packets we use 64 packets as the queue-limit.

Let’s consider a more comprehensive example of how to calculate default queue-limits. Consider the following hierarchical policy-map attached to a GigabitEthernet Interface:

```
policy-map child
  class voice
    priority
    police cir 10m
  class mission-critical
    bandwidth 50000
policy-map parent
  class class-default
    shape average 100m
    service-policy child
interface GigabitEthernet1/0/0
  service-policy out parent
```

For completeness, the scheduling hierarchy for this policy-map would look as follows:



The child policy-map has three queuing classes: voice, mission-critical, and class-default. Let's examine each in turn:

The voice queue is a priority queue so queue-limit will default to 512 packets.

The mission-critical queue has the **bandwidth** command configured with a rate of 50 Mbps so the visible bandwidth will be 50 Mbps (refer to the table above). As this is an Ethernet-type interface the MTU is 1500 bytes:

queue-limit = 50 Mbps / 8 bits * .05 sec / 1500 bytes = 208 packets

Although the implicit class-default has no queuing command configured, the implicit excess weight is equivalent to configuring **bandwidth remaining ratio 1**. This means that class-default will inherit its visible bandwidth from the parent (refer to the table above). At the parent, notice the shape configured with a value of 100 Mbps. The visible bandwidth for class-default in the child is therefore 100 Mbps and as before the MTU for the interface type is 1500 bytes:

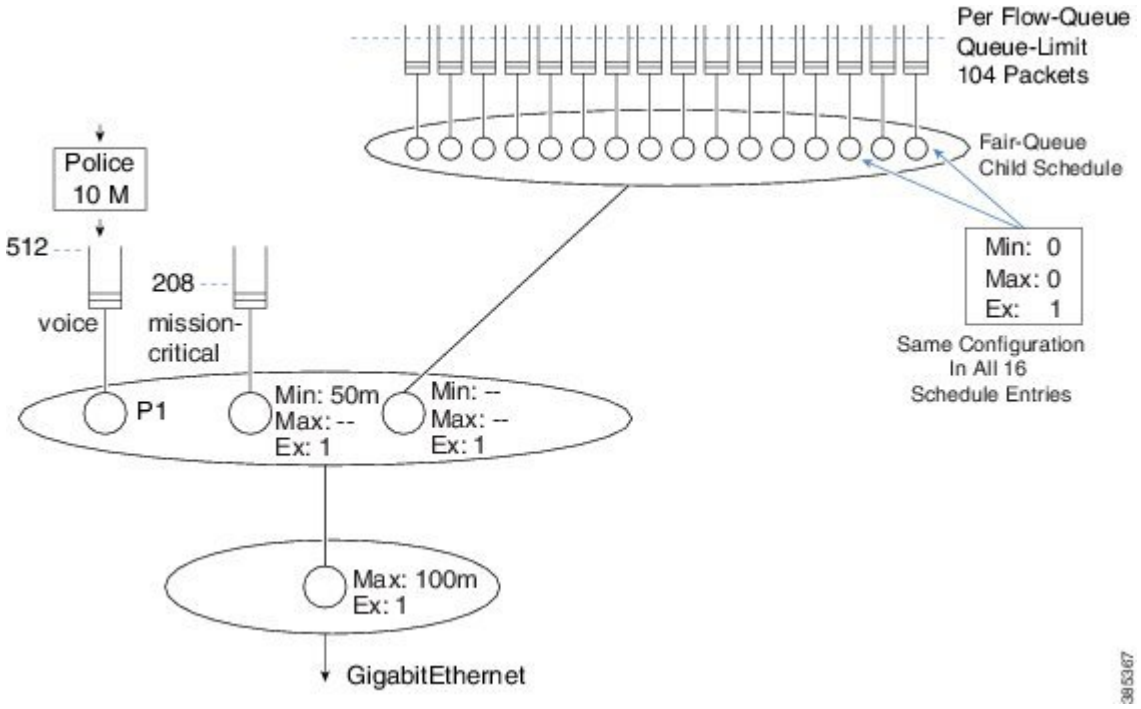
queue-limit = 100 Mbps / 8 bits * .05 sec / 1500 bytes = 416 packets

When Fair-Queue is Configured

In flow-based fair queuing we introduced flow-based fair queuing, where we configure 16 individual flow queues for a class and each flow-queue is configured with the same queue-limit. By default, this limit is ¼ of what is calculated based on the visible bandwidth of the class where the fair-queue feature is configured.

As an example, let's add fair-queue to class-default in the previous configuration example (see the asterisks):

```
policy-map child
  class voice
    priority
    police cir 10m
  class mission-critical
    bandwidth 50000
  class class-default                          ****
    fair-queue                                 ****
policy-map parent
  class class-default
    shape average 100m
    service-policy child
interface GigabitEthernet1/0/0
  service-policy out parent
```



Previously we had calculated queue-limit for class-default to be 416 packets based on the visible bandwidth inherited from the shaper in the parent.

Because flow-based fair-queuing is configured, we create 16 flow queues for that one class. The queue-limit for each individual flow queue is set as 104 packets – ¼ of the 416 packets we calculated.

Changing Queue-Limits

As stated previously, the default queue-limits set by the platform should apply to the majority of users but occasionally tuning them might be required.

Why and When to Change Queue-Limits

Three general situations necessitate tuning queue-limits: OOR drops, bursty traffic leading to tail drops, and latency issues.

When you observe OOR drops, you might need to reduce queue-limits to avoid the situation. As we anticipate that each *bandwidth remaining queue* will inherit its visible bandwidth from the parent, OOR drops may occur when many such queues are created. Additionally, changing queue-limits to byte mode might grant more granular control over how much packet memory a given queue may consume.

Occasionally, we observe that the rate of a stream over time is less than the minimum service rate of a queue. Yet, packets are still tail dropped. You can experiment by dramatically increasing the queue-limit. If systemic oversubscription is the cause, you will tail drops no matter how large you make the queue-limit. If burstiness is causing the drops you should no longer see packet loss. A good starting point is to double the queue-limit – if drops are gone then try reduce to 1.5 times the original queue-limit. You want to find a point where drops are no longer seen but not use unreasonably large queue-limits that may in turn lead to OOR issues. Note that

schedule burstiness caused by mixing very low rates with high rates in the same schedule could also be a cause.

Finally, you might need to adjust queue-limits to avoid unreasonable latency if a queue were to become congested. If you have queues with a visible bandwidth of less than roughly 15Mbps they will be assigned the default minimum queue-limit of 64 packets. If you add multiple queues to low speed interfaces, the minimum guaranteed service rates for those queues can become particularly low. Changing queue-limits to byte mode can be a good choice here.

For QoS Queue

You can use the **queue-limit** command to modify queue limits in any class containing a queuing action (bandwidth, bandwidth remaining, priority or shape). The queue limit may be specified in packets (default), bytes, or time. (We will review an example of each.) Here is an example of setting the limit in packet mode:

```
policy-map packet-mode-example
  class critical-data
    bandwidth percent 50
    queue-limit 2000
```

When you use the **queue-limit** command with the byte option, the second option, you are changing the queue's mode from packet to byte (as discussed previously). For the change to execute, you will need to remove and reattach the policy-map (or save configuration and reload the router). If you want to specify WRED thresholds in bytes you must first use the **queue-limit** command to change the mode of the queue to bytes:

```
policy-map byte-mode-example
  class critical-data
    bandwidth percent 50
    queue-limit 5000 bytes
```

Note If you attempt to change the queue-limit mode while a policy is attached to an interface, you will see an error message:

```
queue-limit 5000 bytes
Runtime changing queue-limit unit is not supported,please remove service-policy first
```

The third option is to specify the queue limit in time (milliseconds). Actually, the hardware only supports units in either packets or bytes. When you specify the unit in milliseconds the router will convert this to bytes; you are effectively changing the mode to byte. The router will use the visible bandwidth of the class.

```
policy-map time-mode-example
  class critical-data
    shape average 20m
    queue-limit 50 ms
```

In this example the visible bandwidth of the queue is 20 Mbits/sec (2.5 Mbytes/sec). In 50mS at a rate of 2.5 Mbytes per/sec, you generate 125000 bytes of data (0.05s*2.5 Mbps). Therefore, in this example, we would set queue-limit at 125000 bytes. You can verify the value calculated in the output of the **show policy-map interface** command.

For Interface Default Queue

You cannot directly change the queue-limit for an interface that does not have an attached QoS policy. In IOS classic, the **hold-queue** command achieved this. In IOS XE, the hold queue exists within the IOSd daemon but is meaningless in the regular packet forwarding path. However, adjusting the hold-queue still has meaning for packets punted to IOSd, provided you have a topology with a very large number of routing peers and require more buffering within IOSd to handle simultaneous updates from all those peers.

To change the queue limit for the interface default queue, you can attach a simple policy-map with just class-default:

```
policy-map modify-interface-queue
  class class-default
    queue-limit 100 ms
!
interface gigabitethernet1/0/0
  service-policy out modify-interface-queue
```

WRED

WRED is a feature that monitors queue utilization. Under congestion and to alleviate further congestion, it randomly drops packets signaling endpoints to reduce their transmission rate.

Relience on Elasticity of IP Flows

WRED relies on the *elasticity of many IP flows*, where *elastic* describes flows that increase and reduce their send rate when the receiver detects missing packets. A very good example of elasticity is TCP. It starts slowly then increases the sender's *congestion window* (amount of outstanding unacknowledged traffic allowed) until either it reaches the maximum *receiver's receive window size* or it loses packets in the network. If the latter, it switches to a *congestion avoidance algorithm*, attempting to settle at the maximum congestion window size achievable without losing packets (see RFC 5681 for further details).

Another good example of elastic traffic is video (consider your favorite video streaming application). After starting the video, you typically observe that video quality improves as the rate increases. The rate continues to increase until the application discerns the capacity of the network. When it detects drops in the network it will recede, delivering the highest quality possible given the prevailing network conditions.

The How of WRED

With the diagram above, we visualize how WRED operates.

Senders behind our router send traffic to receivers elsewhere in the network. WRED is configured on the link (interface) connecting the router to the network. When the sum of the send rates of all the flows exceeds the link capacity we observe packets backing up in the queue configured for that interface.

In the section Tail Drop, on page 619, we described a tail-drops threshold for a queue. WRED uses a lower (minimum) threshold to determine when congestion is occurring. When the queue-depth reaches this threshold, we randomly drop packets rather than enqueue them, despite the queue spaces that are still available. The random nature of the drops ensures that we only drop packets from a small number of flows.

Let's say that you initially drop a single packet from Flow 1. TCP (or whatever elastic transport mechanism) will detect that drop and reduce the send rate of that flow. If by doing so, the link rate now exceeds the

aggregate send rate, then the queue depth will start to fall. If the queue depth falls below the WRED minimum threshold then WRED will cease dropping packets.

If the aggregate send rate still exceeds the link rate then the queue depth will continue to increase and WRED will continue to randomly drop packets. What if we now drop a packet from Flow 4, both Flows 1 and 4 are now backed off. This process continues until enough streams back off to alleviate the congestion.

The random element of WRED ensures that not all flows back off at the same time. If they did, they would likely try to increase their send rates again at the same time, resulting in a saw tooth effect where in synchrony, all senders reduce and increase their send rates. By randomly selecting packets to drop we randomly signal that different flows should back off at different times.

Average Queue Depth

In the previous discussion of WRED, we described random drops occurring when the queue depth crossed a predetermined threshold. Actually, we use a dampened average of the queue depth rather than the *instantaneous queue-depth*, which we use for the tail drop check and *average queue depth* for WRED.

Surely, this is no surprise: internet traffic is bursty. If we used instantaneous queue depth to monitor congestion we might drop packets in haste and so respond to normal bursts in traffic rather than to real congestion.

To determine how changes in average queue depth are dampened, we use the *WRED Exponential Weighting Constant*. The router will remember the current value of average queue depth. Whenever a packet reaches the enqueue stage, we examine the instantaneous queue depth and recalculate the average queue depth. The formula to calculate the new value of average queue depth is as follows:

Avg = OldAvg + (Instantaneous-OldAvg)/2^exp-weighting-constant

where Avg is the average queue depth calculated at the current enqueue time; Instantaneous, the current queue depth; and OldAvg, the previously calculated average that we remembered since the last enqueue.

For example, if the OldAvg is 12.0 packets, the Instantaneous is 14 packets (observed upon enqueuing a packet), and exp-weighting-constant is 6 (the default for packet mode WRED on the ASR 1000 Router), the Avg would be:

Avg = 12 + (14 – 12) / 2^6 = 12 + .03125 = **12.03125**

Note exp-weighting-constant = 9 if the queue is run in byte mode.

Later, we enqueue another packet. If one packet was transmitted from the head of the queue in the interim, the instantaneous queue depth would remain 14. Now, the calculation of AVG yields:

Avg = 12.03125 + (14 – 12.03125)/2^6 = 12.03125 + 0.0308 = **12.06201**

The example shows that the average queue depth is dampened. The instantaneous queue depth can grow considerably beyond the average. Consequently, the WRED max threshold is always considerably less than the queue limit. The example also illustrates the time necessary for the average to converge on the instantaneous, even if the queue depth stays consistent for some time. This dampening of average queue depth is how WRED avoids reacting to regular microbursts in traffic.

Only with a PHd in voodoo mathematics, should you consider changing the value of EWC. It is a "true geek knob" that should be avoided. For completeness only and not to encourage, here is the code change the EWC:

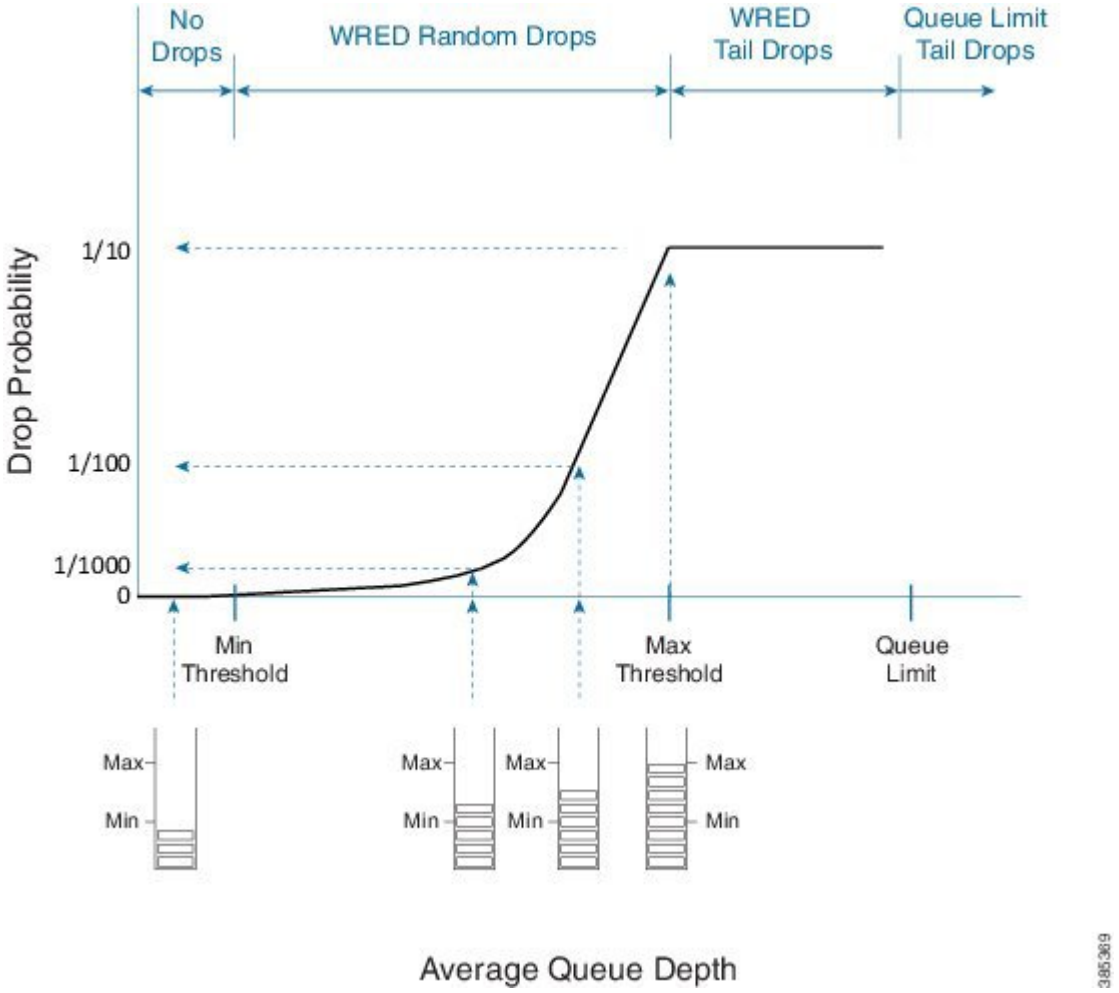
```
policy-map ewc-example
  class class-default
    random-detect
    random-detect exponential-weighting-constant 5
```

WRED Threshholds and Drop Curves

WRED drop decisions are driven by the average queue depth calculated upon enqueue.

When configuring WRED, we set a Minimum threshold, a Maximum threshold, and a Drop Probability for each precedence value (or DSCP, discard-class, etc.).

The following diagram shows the drop curve for a sample precedence value.



If the average queue depth is calculated to a value less than the WRED minimum threshold we are in a range where WRED will not drop any packets.

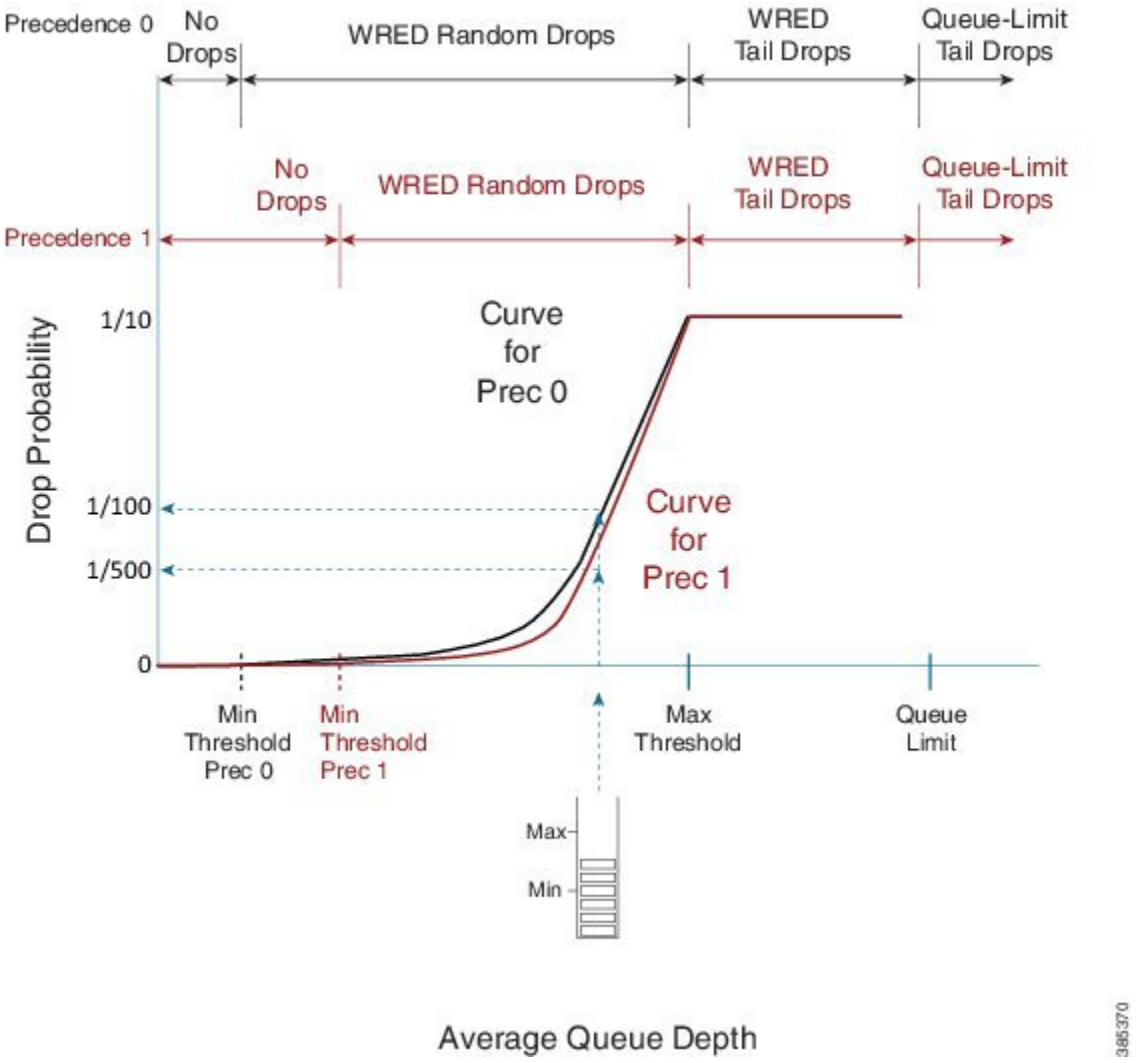
Between the Min and Max thresholds we are in the WRED random drop zone. Observe the exponential rise in drop probability from zero at the Min threshold to the configured WRED drop probability (defaults to 1 in 10 packets) at the Max threshold. The exponential nature of the curve means WRED will drop a very small number of packets when the average queue depth approximates the minimum threshold. As the illustration

as reflects, the average queue depth and drop probability increase in tandem. Knowing the average queue depth, we know the associated drop probability. Then, we decide whether to drop the packet or enqueue it.

If the WRED average queue depth is calculated to exceed the maximum threshold then the packet will experience a *WRED tail drop*. This differs slightly from the queue-limit tail drop - it is driven by average rather than instantaneous queue depth. If the instantaneous queue depth reaches the class's queue limit then the drop would be recorded as a queue-limit tail drop as opposed to a WRED tail drop.

Please note that the 'W' in WRED stands for *weighted* (some traffic may be dropped more aggressively than others).

Here, we show how to use multiple drop curves:



If a packet with IP precedence 0 arrives, the router will apply the black curve to calculate the drop probability. In the example the drop probability is calculated as 1 in 100 packets.

If a packet with IP precedence 1 arrives, we apply the mauve-colored curve. For the same average queue depth we would now see a drop probability of just 1 in 500.

Notice how the default maximum threshold is the same for each precedence value. The difference in WRED minimum threshold for each precedence value means that we will start dropping precedence 0 traffic before we drop any other traffic. Moreover, we will be more aggressive in dropping that traffic for a given queue depth in the random drop zone.

Note When you configure WRED, for each drop curve, the router will pick appropriate values for Min threshold, Max threshold and Drop Probability. Those values depend on the configured queue limit thereby accounting for the interface speed. We strongly recommend that you use the default values unless you fully understand the implications of any change.

WRED - Changing Drop Curves

Regardless of the WRED mode, you can tune any individual drop curve. Using the same command you may change the Minimum Threshold, the Maximum Threshold or the Drop Probability at Maximum Threshold for that drop curve. With the minimum and maximum thresholds and the drop probability, a router can construct the exponential curve it needs to determine drop probability for any average queue depth. Tuning WRED parameters is not typical; do not attempt unless you have a thorough understanding of how tuning will impact applications in that class. The default values should suffice for the vast majority of use cases.

If you decide to tune WRED drop curves, you have the option to specify thresholds in packets (default), bytes or time. The queue-limit must be configured in the chosen unit before you add WRED configuration to the class and only when the queue is already running in the desired mode can you change thresholds in that unit. Moreover, you can only change the curve for a particular DSCP, precedence or discard-class value provided WRED is operating in that mode.

Recall that the drop probability is an integer number. If the average queue limit is at the maximum threshold, a packet has a *1 in that integer value chance* of being dropped. For example, if the drop probability is 20, a 1 in 20 (5%) chance exists for a packet to dropped by WRED.

The command to change a drop curve is **random-detect [dscp|precedence|discard-class]** *value min-threshold max-threshold drop-probability*, as illustrated here:

```
policy-map tuneprecedence
  class bulk-data
    bandwidth remaining percent 30
    random-detect
    random-detect precedence 1 1301 2083 10
```

Running the queue in packet mode (the default) and WRED in precedence mode (also the default), I decide against differentiation in the minimum threshold for precedence 1 and 2. I change the curve for precedence 1, setting the minimum threshold to 1301, the maximum threshold to 2083 and the drop probability at max threshold to 1 in 10 packets:

random-detect precedence 1 1301 2083 10

As always, we can verify the configuration with the **show policy-map interface** command:

```
show policy-map interface g1/0/0
 GigabitEthernet1/0/0

  Service-policy output: tuneprecedence

    Class-map: bulk-data (match-all)
```

```
  0 packets, 0 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match: access-group name bulkdata
  Queueing
  queue limit 4166 packets
  (queue depth/total drops/no-buffer drops) 0/0/0
  (pkts output/bytes output) 0/0
  bandwidth remaining 30%
    Exp-weight-constant: 4 (1/16)
    Mean queue depth: 1086 packets
    class  Transmitted   Random drop   Tail drop   Minimum   Maximum   Mark
           pkts/bytes    pkts/bytes    pkts/bytes  thresh    thresh    prob

    0         0/0           0/0           0/0       1041      2083      1/10
    1        0/0           0/0           0/0       1301      2083      1/10
    2         0/0           0/0           0/0       1301      2083      1/10
    3         0/0           0/0           0/0       1431      2083      1/10
    4         0/0           0/0           0/0       1561      2083      1/10
    5         0/0           0/0           0/0       1691      2083      1/10
    6         0/0           0/0           0/0       1821      2083      1/10
    7         0/0           0/0           0/0       1951      2083      1/10
```

Notice the new values we set for precedence 1.

What if we change the thresholds for a queue that is running in time-based mode where WRED is running in DSCP mode? In particular, we want the minimum threshold of af21 to exceed that of af11. The configuration would appear as follows:

```
policy-map tunedscp
  class bulk-data
    bandwidth remaining percent 30
    queue-limit 50 ms
    random-detect dscp-based
    random-detect dscp af21 22 ms 25 ms 10
```

Looking at the output of **show policy-map interface** we verify the configuration:

```
show policy-map interface g1/0/0
 GigabitEthernet1/0/0

  Service-policy output: tunedscp

    Class-map: bulk-data (match-all)
      148826 packets, 223239000 bytes
      5 minute offered rate 2358000 bps, drop rate 0000 bps
      Match: access-group name bulkdata
      Queueing
      queue limit 50 ms/ 6250000 bytes
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 148826/223239000
      bandwidth remaining 30%

        Exp-weight-constant: 9 (1/512)
        Mean queue depth: 0 ms/ 992 bytes
        dscp    Transmitted     Random drop  Tail drop   Minimum    Maximum     Mark
                pkts/bytes      pkts/bytes   pkts/bytes  thresh     thresh      prob
                                                         ms/bytes   ms/bytes
        af11    96498/144747000   0/0          0/0       21/2734375 25/3125000  1/10
        af21    52328/78492000    0/0          0/0       22/2750000 25/3125000  1/10
```

With DSCP-based WRED we will only show curve statistics for DSCP values that have been observed within that class (refer to Mode: Precedence, DSCP, and Discard-Class, on page 638).

WRED Max Thresholds for Priority Enqueue

In the WRED Threshholds and Drop Curves, on page 632, we showed how to tune the minimum threshold of WRED curves. Another option is to modify the maximum threshold. When you do so with different thresholds for different DSCP values you can effectively claim that under congestion we always drop one type of traffic.

Let's use af11 to designate in-contract bulk data traffic and af12 to designate out-of-contract bulk data traffic? Under congestion, we want to always provide preferential treatment to af11 over af12. If we specify a lower WRED maximum threshold for af12 we could drop this traffic while still enqueuing af11.

In the following configuration, we change the maximum threshold for af12 from the default of 624 packets (for this bandwidth) to 580 packets:

```
policy-map maxthreshold
  class bulk-data
    bandwidth percent 30
    random-detect dscp-based
    random-detect dscp af12 468 580 10
```

Let's verify the configuration:

```
show policy-map interface g1/0/0
 GigabitEthernet1/0/0

  Service-policy output: maxthreshold

    Class-map: bulk-data (match-all)
      359826 packets, 539739000 bytes
      5 minute offered rate 7208000 bps, drop rate 0000 bps
      Match: access-group name bulkdata
      Queueing
      queue limit 1249 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 359826/539739000
      bandwidth 30% (300000 kbps)
        Exp-weight-constant: 4 (1/16)
        Mean queue depth: 0 packets
        dscp   Transmitted     Random drop   Tail drop    Minimum   Maximum   Mark
               pkts/bytes      pkts/bytes    pkts/bytes   thresh    thresh    prob

        af11  154689/232033500    0/0           0/0        546       624      1/10
        af12  205137/307705500    0/0           0/0        468       580      1/10
```

Looking at the configuration you can see that if the average queue depth exceeds 580 packets, all af12 packets would be *WRED tail dropped* but we would still enqueue af11 packets.

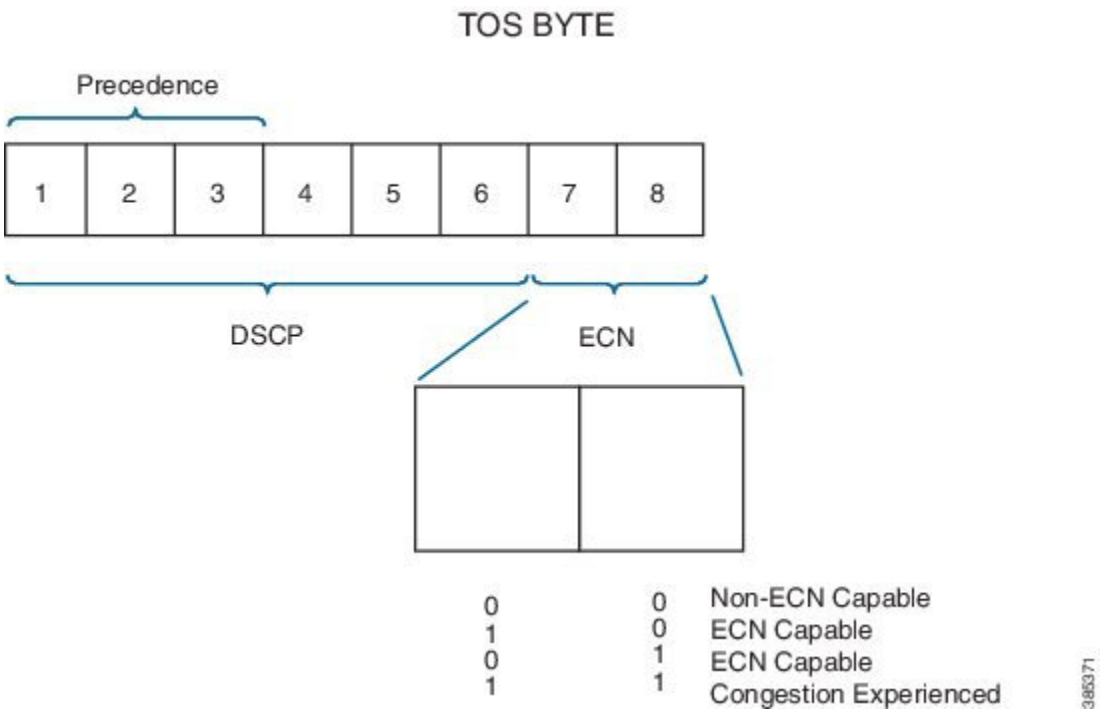
Be alert when modifying maximum thresholds to ensure that behavior is as expected. Here, if congestion persists and the average queue depth remains above 580 packets, then we would totally starve af12 traffic of any service during persistant congestion.

ECN - Explicit Congestion Notification

Explicit Congestion Notification (ECN) is an extension to the IP protocol where the network can mark a packet to signal congestion to the endpoints rather than drop packets early to signal congestion. Upon receiving such a packet, the endpoint echoes that congestion notification back to the sender.

Note ECN mode must be explicitly enabled in WRED.

To understand ECN you should first grasp the TOS byte in the IP header. Originally it was used to carry IP precedence bits in the 3 most significant bits, and more recently, to carry the DSCP codepoint in the 6 most significant bits of this byte. We define the remaining 2 bits in RFC3168 as ECN bits.



When WRED is configured in ECN mode it will look at the ECN bits before dropping a packet. If these bits are both set to zero, the router assumes the endpoints are ECN incapable and WRED will drop the packet to signal congestion is occurring.

If either of the ECN bits is set to 1, the router assumes that the endpoint is ECN capable and can mark congestion experienced rather than dropping the packet by setting both ECN bits to 1. The endpoints must signal a transport is ECN capable only if the upper layer protocol is elastic in nature.

Note The router will only look at the ECN bits when determining whether to mark or drop a packet.

The following is an example of configuring WRED in ECN mode:

```
policy-map ecn-example
  class bulk-data
    bandwidth remaining percent 30
```

```
random-detect dscp-based
random-detect ecn
```

Mode: Precedence, DSCP, and Discard-Class

WRED Precedence Mode

In WRED Threshholds and Drop Curves, on page 632 we describe drop curves.

When you enable WRED the default is to run in *precedence mode* and create 8 distinct drop curves, one for each valid precedence value. The default minimum threshold increases with the precedence value. The impact is that precedence 0 will start dropping earlier and more aggressively than precedence 1, precedence 1 earlier and more aggressively then precedence 2, etc. The same default maximum threshold and drop probability is configured for each curve.

When a packet arrives the IP precedence bits determine which curve we use to find the appropriate drop probability. If the packet is not "IP," we use the precedence 0 drop curve. If the packet is MPLS encapsulated then the EXP bits are treated as precedence bits and determine the appropriate drop curve.

In the following example we enable WRED in precedence mode. Observe that WRED must reside in a class that has a queuing action (including class-default):

```
policy-map wred-precedence-example
  class bulk-data
     bandwidth remaining percent 30
     random-detect
     random-detect precedence-based
```

In this example, we use **bandwidth remaining** command as the queuing action. The **random-detect** command enables WRED in the class bulk-data and the **random-detect precedence-mode** command tells WRED to operate in precedence mode.

Note The **random-detect precedence-mode** command is optional as the default mode for WRED is precedence-based.

As with all QoS features the **show policy-map interface** command is the primary means to verify your configuration:

```
show policy-map int g1/0/0
 GigabitEthernet1/0/0

  Service-policy output: wred-precedence-example

   Class-map: bulk-data (match-all)
    6468334 packets, 9702501000 bytes
    5 minute offered rate 204108000 bps, drop rate 0000 bps
    Match: access-group name bulkdata
    Queueing
    queue limit 4166 packets
    (queue depth/total drops/no-buffer drops) 1308/0/0
    (pkts output/bytes output) 6468335/9702502500
    bandwidth remaining 30%
     Exp-weight-constant: 4 (1/16)
     Mean queue depth: 1308 packets
     class     Transmitted       Random drop    Tail drop      Minimum    Maximum     Mark
```

```
            pkts/bytes          pkts/bytes      pkts/bytes      thresh      thresh      prob

  0             0/0               0/0             0/0           1041        2083        1/10

  1             0/0               0/0             0/0           1171        2083        1/10

  2             0/0               0/0             0/0           1301        2083        1/10

  3             0/0               0/0             0/0           1431        2083        1/10

  4       6468335/9702502500      0/0             0/0           1561        2083        1/10

  5             0/0               0/0             0/0           1691        2083        1/10

  6             0/0               0/0             0/0           1821        2083        1/10

  7             0/0               0/0             0/0           1951        2083        1/10
```

Notice how statistics and curve configuration values are displayed for each of the 8 drop curves that are created in precedence mode. The average queue-depth is less than the minimum threshold so no random drops are reported.

WRED DSCP Mode

The second option for configuring WRED is DSCP mode, where we create 64 unique curves.

Similar to precedence mode, any non-IP traffic will use the default (DSCP 0) curve. If MPLS traffic is seen, we treat the MPLS EXP bits as precedence values and select the curve accordingly (EXP 1 treated as DSCP CS1, EXP 2 as CS2, etc.).

Here is an example of configuring WRED in DSCP mode:

```
policy-map wred-dscp-example
  class bulk-data
    bandwidth remaining percent 30
    random-detect dscp-based
```

Here, we verify the configuration with the **show policy-map interface** command:

```
show policy-map int
 GigabitEthernet1/0/0

  Service-policy output: wred-dscp-example

    Class-map: bulk-data (match-all)
      5655668 packets, 8483502000 bytes
      5 minute offered rate 204245000 bps, drop rate 0000 bps
      Match: access-group name bulkdata
      Queueing
      queue limit 4166 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 5655669/8483503500
      bandwidth remaining 30%
       Exp-weight-constant: 4 (1/16)
       Mean queue depth: 1 packets
       dscp   Transmitted      Random drop   Tail drop    Minimum   Maximum   Mark
              pkts/bytes       pkts/bytes  pkts/bytes    thresh     thresh    prob
       af11 1205734/1808601000     0/0         0/0         1821      2083      1/10
```

```
  cs4  5270109/7905163500     0/0         0/0         1561      2083      1/10
```

Notice that we only display statistics and drop curve information for 2 DSCP values (af11 and cs4). In DSCP mode, 64 unique drop curves are configured and IOS will maintain statistics for all. However, it will only display information for drop curves that have actually observed traffic. In this example, we have observed only display traffic with DSCP af11 and cs4, hence the display.

WRED Discard-Class

Discard-class is an internal marking very similar in concept to qos-group. We can mark discard-class on ingress (and not on egress) as well as employ to select a WRED drop curve on egress.

Occasionally, the precedence or DSCP marking in a packet is unavailable for classification on an egress interface. A use-case is an MPLS-encapsulating router where we receive IP packets on the ingress interface and forward MPLS-encapsulated packets on the egress interface.

DSCP must be mapped into a smaller number of EXP values (6 bits in the DiffServ field vs 3-bit field in MPLS header) so some granularity is lost. Let's say af11 is used for in-contract and af12 for out-of-contract bulk data. On the egress interface the DSCP visibility is lost; af11 and af12 would probably be mapped into the same EXP. Now, what if we want to provide preferential treatment to af11 over af12 on the egress interface?

We could use WRED discard-class mode to achieve this. To do so, you will need to mark discard-class on ingress interfaces, as in the following sample policy:

```
policy-map mark-in-contract
  class bulk-data
    police cir 50000000 pir 100000000
    conform-action set-dscp-transmit af11
    conform-action set-mpls-exp-imposition-transmit 1
    conform-action set-discard-class-transmit 2
    exceed-action set-dscp-transmit af12
    exceed-action set-mpls-exp-imposition-transmit 1
    exceed-action set-discard-class-transmit 1
    violate-action drop
```

In this policy traffic adhering to the CIR is marked as in-contract:

```
    conform-action set-dscp-transmit af11
    conform-action set-mpls-exp-imposition-transmit 1
    conform-action set-discard-class-transmit 2           ****
```

Traffic between the CIR and PIR is marked as out-of-contract:

```
    exceed-action set-dscp-transmit af12
    exceed-action set-mpls-exp-imposition-transmit 1
    exceed-action set-discard-class-transmit 1            ****
```

Violating traffic is dropped.

Notice how the same EXP value will be set for conforming and exceeding traffic – it is all bulk data traffic and will use the same per-hop-behavior in the MPLS network. However, for in-contract and out-of-contract traffic we also mark distinct discard-classes (see the asterisks), which we use use on the egress interface to provide preferential treatment.

On the egress interface you would configure WRED in discard-class-based mode, as follows:

```
policy-map wred-discard-class-example
  class bulk-data
    bandwidth remaining percent 30
    random-detect discard-class-based
```

Looking at the output of **show policy-map interface** command you will see something like:

```
show policy-map int g1/0/0
 GigabitEthernet1/0/0

  Service-policy output: wred-discard-class-example

    Class-map: bulk-data (match-all)
      1500 packets, 1040000 bytes
      5 minute offered rate 51955000 bps, drop rate 0000 bps
      Match: access-group name bulkdata
      Queueing
      queue limit 4166 packets
      (queue depth/total drops/no-buffer drops) 943/0/0
      (pkts output/bytes output) 1500/1040000
      bandwidth remaining 30%
        Exp-weight-constant: 4 (1/16)
        Mean queue depth: 943 packets
        discard-class  Transmitted  Random drop   Tail drop   Minimum   Maximum  Mark
                       pkts/bytes   pkts/bytes    pkts/bytes  thresh    thresh   prob

        0                 0/0          0/0          0/0        1041      2083    1/10
        1               500/4000       0/0          0/0        1171      2083    1/10
        2              1000/1000000    0/0          0/0        1301      2083    1/10
        3                 0/0          0/0          0/0        1431      2083    1/10
        4                 0/0          0/0          0/0        1561      2083    1/10
        5                 0/0          0/0          0/0        1691      2083    1/10
        6                 0/0          0/0          0/0        1821      2083    1/10
        7                 0/0          0/0          0/0        1951      2083    1/10
```

Looking at the output you can see that 8 drop curves are created when you run WRED in discard-class mode. Referring to the configuration above, in-contract traffic is marked with discard-class 2 and out-of-contract traffic is marked with discard-class 1.

You can also see that the WRED curve for discard-class 1 has a lower minimum threshold. This means that under congestion, out-of-contract traffic will start dropping earlier and more aggressively than in-contract traffic.

Any traffic devoid of an explicitly-set discard-class is assumed to that does not have a discard-class explicitly set will be assumed to be discard-class 0.

Command Reference - random detect

Use the **random-detect** *options* command to enable and control operation of WRED, applying different options as below.

To enable WRED – use one of the following:
random-detect

Enable WRED in precedence mode.

random-detect precedence-based

Enable WRED in precedence mode.

random-detect dscp-based

Enable WRED in DSCP mode.

random-detect discard-class-based

Enable WRED in discard-class mode.

To tune WRED Drop Curve – use one of the following

random-detect precedence *value min-threshold max-threshold drop-probability*

Modify the drop curve for a particular precedence value

random-detect dscp *value min-threshold max-threshold drop-probability*

Modify the drop curve for a particular DSCP value

random-detect precedence *value min-threshold max-threshold drop-probability*

Modify the drop curve for a particular discard-class value. Note the min-threshold and max-threshold may be configured in packets (default), bytes or time. To use the units of bytes or time the queue must first be configured for that mode using the **queue-limit** command.

To change the WRED Exponential Weighting Constant

random-detect exponential-weighting-constant *value*

To enable Explicit Congestion Notification Support

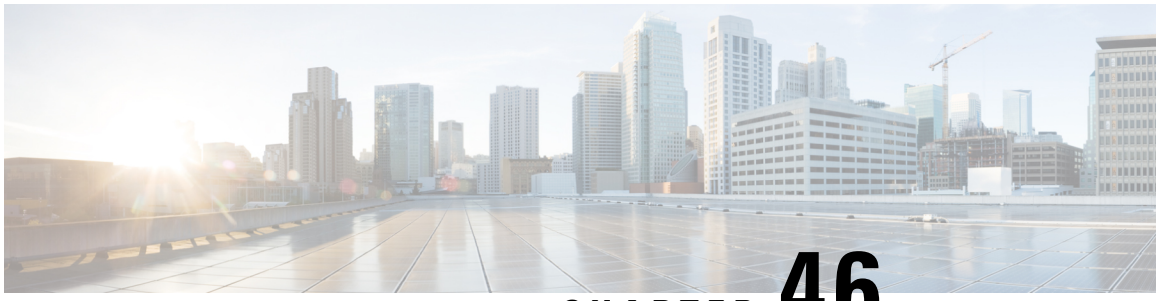
random-detect ecn

Usage:

The **random-detect** command may be used in any queuing class configured with the **bandwidth**, **bandwidth remaining** or **shape**commands. This includes class-default which has an implicit bandwidth remaining value.

The ASR 1000 Series Aggregation Services Router has no queues in parent or grandparent levels of a scheduling hierarchy. So, the **random-detect** command is not supported in any class that contains a child queuing policy.

The default values for WRED minimum and maximum thresholds are proportional to the queue-limit for a class and therefore proportional to the expected service rate of the queue. Modifying WRED drop curves should not be undertaken unless you have a deep understanding on how changes will affect applications in that class.



CHAPTER 46

Information About QoS for Etherchannels

Etherchannel with QoS Feature Evolution

An Etherchannel is a port-channel architecture that allows grouping of several physical links to create one logical Ethernet link for the purpose of providing fault tolerance, and high-speed links between switches, routers, and servers. An Etherchannel can be created from between two and eight active Fast, Gigabit, or 10-Gigabit Ethernet ports, with an additional one to eight inactive (failover) ports, which become active as the other active ports fail.

QoS for Etherchannel interfaces has evolved over several Cisco IOS XE releases. It is important to understand what level of support is allowed for your current level of Cisco IOS XE software and underlying Etherchannel configuration. Various combinations of QoS are supported based on how Etherchannel is configured. There are three different modes in which Etherchannel can be configured:

- Etherchannel VLAN-based load balancing via port-channel subinterface encapsulation CLI
- Etherchannel Active/Standby with LACP (no Etherchannel load balancing)
- Etherchannel with LACP with load balancing

Each of these models has specific restrictions regarding which levels of Cisco IOS XE software include support and the possible QoS configurations with each.

The following summarizes the various Etherchannel and QoS configuration combinations that are supported. Example configurations will be provided later in this document. Unless specifically mentioned together, the

combination of service policies in different logical and physical interfaces for a given Etherchannel configuration is not supported.

Etherchannel VLAN-Based Load Balancing via Port-Channel Subinterface Encapsulation CLI

Supported in Cisco IOS XE Release 2.1 or later:

- Egress MQC Queuing Configuration on Port-Channel Subinterface
- Egress MQC Queuing Configuration on Port-Channel Member Link
- QoS Policies Aggregation—Egress MQC Queuing at Subinterface
- Ingress Policing and Marking on Port-Channel Subinterface
- Egress Policing and Marking on Port-Channel Member Link

Supported in Cisco IOS XE Release 2.6 or later:

- QoS Policies Aggregation—MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface

Etherchannel Active/Standby with LACP (No Etherchannel Load Balancing)

Supported in Cisco IOS XE 2.4 or later:

- Egress MQC Queuing on Port-Channel Member Link—No Etherchannel Load Balancing

Etherchannel with LACP and Load Balancing

Supported in Cisco IOS XE 2.5 or later:

- Egress MQC Queuing Configuration on Port-Channel Member Link—Etherchannel Load Balancing

Supported in Cisco IOS XE 3.12 or later:

- General MQC QoS support on Port-channel main-interface

  We recommend that as a best practice for QoS, that you use port-channel aggregation—see the "Aggregate EtherChannel Quality of Service" chapter.

Supported in Cisco IOS XE 3.16.3 or later and in Cisco IOS XE Fuji 16.3 or later:

- General MQC QoS support on Port-channel sub-interface

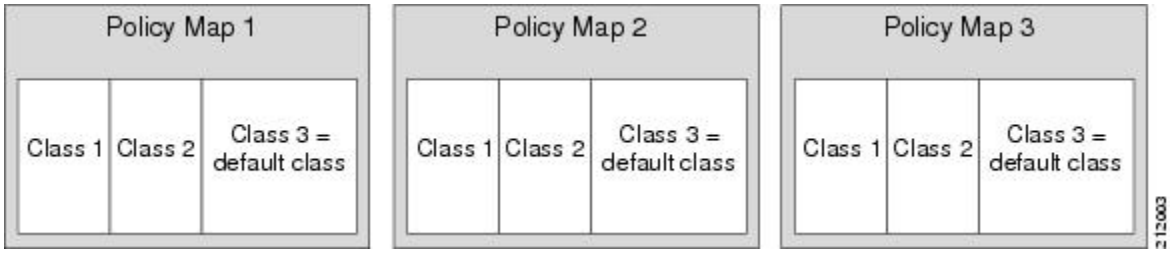
  We recommend that as a best practice for QoS, that you use port-channel aggregation—see the "Aggregate EtherChannel Quality of Service" chapter.

Understanding Fragments in Class Definition Statements

The QoS Policies Aggregation feature introduces the idea of fragments in class definition statements. A default traffic class definition statement can be marked as a fragment within a policy-map. Other policy-maps on the same interface can also define their default traffic class statements as fragments, if desired. A separate policy-map can then be created with a service fragment class definition statement that will be used to apply QoS to all of the fragments as a single group.

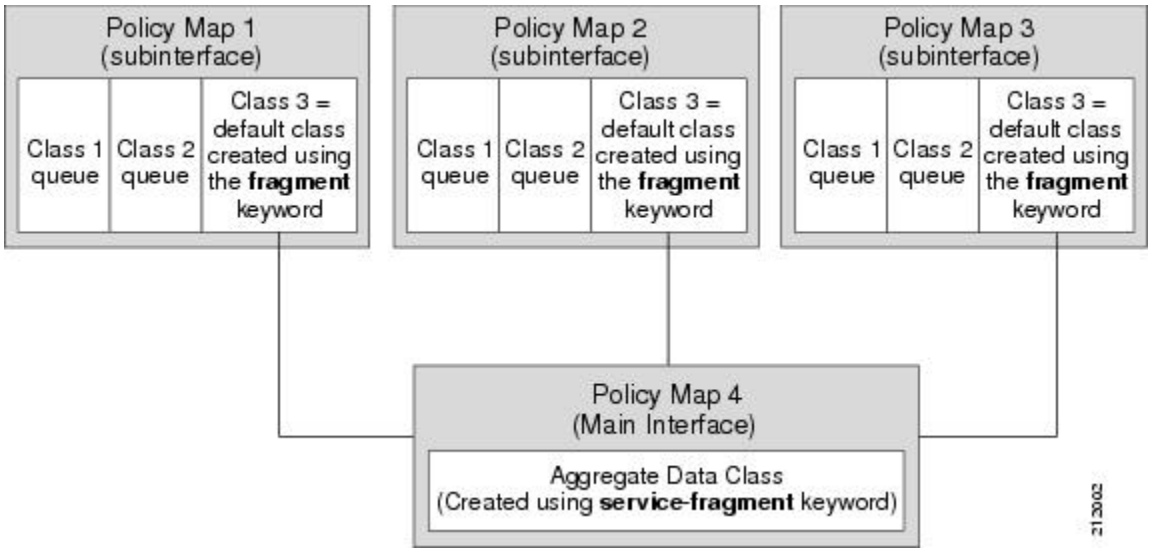
The figure below provides an example of one physical interface with three attached policy-maps that is not using fragments. Note that each policy-map has a default traffic class that can classify traffic only for the default traffic within its own policy-map.

Figure 89: Physical Interface with Policy-Maps—Not Using Fragments



The figure below shows the same configuration configured with fragments, and adds a fourth policy-map with a class definition statement that classifies the fragments collectively. The default traffic classes are now classified as one service fragment group rather than three separate default traffic classes within the individual policy-maps.

Figure 90: Physical Interface with Policy-Maps—Using Fragments



Fragments for Gigabit Etherchannel Bundles

When fragments are configured for Gigabit Etherchannel bundles, the policy-maps that have a default traffic class configured using the **fragment** keyword are attached to the member subinterface links, and the policy-maps that have a traffic class configured with the **service-fragment** keyword to collectively classify the fragments is attached to the physical interface.

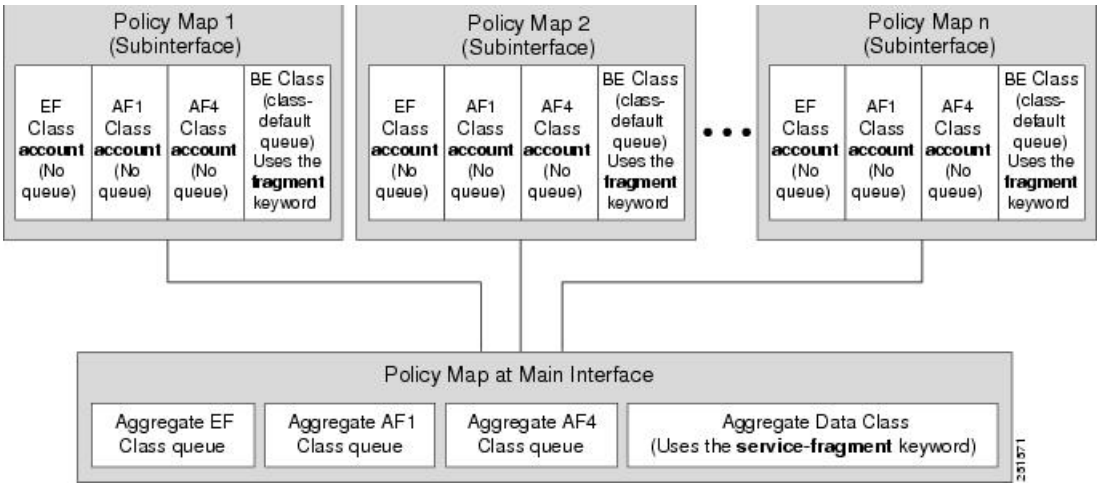
All port-channel subinterfaces configured with fragments that are currently active on a given port-channel member link will use the aggregate service fragment class on that member link. If a member link goes down, the port-channel subinterfaces that must switch to the secondary member link will then use the aggregate service fragment on the new interface.

QoS: Policies Aggregation MQC

The QoS: Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface feature extends the previous support of aggregation of class-default traffic using the **fragment** and **service-fragment** configurations, to other user-defined traffic classes in a subinterface policy-map, such as DSCP-based traffic classes, that are aggregated at the main-interface policy-map as shown in the figure below.

When no queueing is configured on a traffic class in the subinterface policy-map, the **account** command can be used to track queueing drops that occur at the aggregate level for these classes, and can be displayed using the **show policy-map interface** command.

Figure 91: Policy-Map Overview for the MQC Support for Multiple Queue Aggregation at Main Interface Feature



Differences Between the Original Feature and the MQC Support for Multiple Queue AggregationDifferences Between Policy Aggregation—Egress MQC Queuing at Subinterface and the MQC Support for Multiple Queue Aggregation at Main Interface

Although some of the configuration between the “Policy Aggregation – Egress MQC Queuing at Subinterface” scenario and the “MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface” scenario appear similar, there are some important differences in the queuing behavior and the internal data handling. See the figure in the “Understanding the QoS: Policies Aggregation MQC” section.

For example, both configurations share and require the use of the **fragment** keyword for the **class class-default** command in the subscriber policy-map, as well as configuration of the **service-fragment** keyword for a user-defined class in the main-interface policy-map to achieve common policy treatment for aggregate traffic. However, the use of this configuration results in different behavior between the original and enhanced QoS policies aggregation implementation:

- In the original implementation using the fragment and service-fragment architecture, all default class traffic and any traffic for classes without defined queueing features at the subinterface goes to the

class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy-map. Subinterface traffic aggregation (for example, from multiple subscribers on the same physical interface) ultimately occurs only for a single class, which is the default class.

- In the enhanced implementation of the MQC Support for Multiple Queue Aggregation at Main Interface feature also using the fragment and service-fragment architecture, all default class traffic also goes to the class-default queue and is aggregated into a common user-defined queue and policy defined at the main policy-map. However, other classes, such as DSCP-based subscriber traffic classes, are also supported for an aggregate policy. These traffic classes do not support any queues or queueing features other than **account** at the subscriber policy-map. The use of the fragment and service-fragment architecture enables these other subscriber traffic classes (from multiple subscribers on the same physical interface) to achieve common policy treatment for aggregate traffic that is defined for those same classes at the main policy-map.

How to Configure QoS for Etherchannels

Configuring Egress MQC Queuing on Port-Channel Subinterface

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The port-channel subinterface should have been previously configured with the appropriate encapsulation subcommand to match the select primary and secondary physical interfaces on the Etherchannel. Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command, or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-number* **.** *subinterface-number*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface port-channel** *port-channel-number* **.** *subinterface-number* **Example:** `Device(config)# interface port-channel 1.200`	Specifies the port-channel subinterface that receives the service policy configuration.

Step 4	**service-policy output** *policy-map-name*
	Example:
	`Device(config-subif)#`
	`service-policy output WAN-GEC-sub-Out`
		Specifies the name of the service policy that is applied to output traffic.

Step 5	**end** **Example:** `Device(config-subif)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring Egress MQC queuing on Port-Channel Member Links

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map that uses queuing features should be configured using previously defined class maps. The Etherchannel member link interface should already be configured to be part of the channel group (Etherchannel group). No policy-maps that contain queuing commands should be configured on any port-channel subinterfaces. Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command, or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service policy configuration.

Step 4	**service-policy output** *policy-map-name*
	Example:
	`Device(config-if)#`
	`service-policy output WAN-GEC-sub-Out`
		Specifies the name of the service policy that is applied to output traffic for this physical interface that is part of the Etherchannel.

Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring QoS Policies Aggregation—Egress MQC Queuing at Subinterface

Before you begin

Default class traffic from multiple Port-channel subinterfaces can be aggregated into a common policy-map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and the **service-fragment** configuration at the main interface class. Queuing occurs at the subinterface for other traffic classes that are defined with queuing features in the subinterface policy-map.

This feature is configured using Modular QoS CLI (MQC). It is most useful in QoS configurations where several policy-maps attached to the same physical interface want aggregated treatment of multiple default traffic classes from multiple port-channel sub-interfaces. Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command, or the port-channel main-interface must have the **load-balancing vlan** command. It is assumed that these commands have already been executed.

Note This feature is supported when policy-maps are attached to multiple port-channel subinterfaces and the port-channel member link interfaces. This feature cannot be used to collectively classify default traffic classes of policy-maps on different physical interfaces. It can collectively classify all traffic directed toward a given port-channel member link when designated by the **primary** or **secondary** directives on the subinterface **encapsulation** command. All subinterface traffic classes should have queues. However, when a traffic class in the subinterface policy-map is not configured with any queuing feature (commands such as **priority**, **shape**, **bandwidth**, **queue-limit**, **fair-queue**, or **random-detect**), the traffic is assigned to the class-default queue. No classification occurs or is supported at the main interface policy-map for any subinterface traffic classes that do not use the **fragment** and **service-fragment** configuration.

A multistep process is involved with the complete configuration of the QoS Policies Aggregation feature. The following sections detail those steps.

Note the following about attaching and removing a policy-map:

- To configure QoS Policies Aggregation, you must attach the policy-map that contains the **service-fragment** keyword to the main interface first, and then you must attach the policy-map that contains the **fragment** keyword to the subinterface.
- To disable QoS Policies Aggregation, you must remove the policy-map that contains the fragment keyword from the subinterface first, and then you must remove the policy-map that contains the service-fragment keyword from the main interface.

Configuring a Fragment Traffic Class in a Policy-Map

Before you begin

This procedure shows only how to configure the default traffic class as a fragment within a policy-map. It does not include steps on configuring other classes within the policy-map, or other policy-maps on the device.

Example

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a fragment named BestEffort is created in policy-map subscriber1 and policy-map subscriber 2. In this example, queuing features for other traffic classes are supported at the subinterface policy-map.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example also shows how to configure a fragment named BestEffort for the default class in a policy-map on a subinterface using the QoS Policies Aggregation MQC Support for Multiple Queue Aggregation at Main Interface implementation. In this example, notice that queuing features are not supported for the other classes in the policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

After configuring default class statements as fragments in multiple subinterface policy-maps, a separate policy-map with a class statement using the service-fragment keyword must be configured to apply QoS to the class stratements configured as fragments.

What to Do Next

After configuring multiple default class statements as fragments in a policy-map, a separate policy-map with a class statement using the **service-fragment** keyword must be configured to apply QoS to the class statements configured as fragments.

This process is documented in the “Configuring a Service Fragment Traffic Class” section.

Configuring a Service Fragment Traffic Class

Before you begin

This task describes how to configure a service fragment traffic class statement within a policy-map. A service fragment traffic class is used to apply QoS to a collection of default class statements that have been configured previously in other policy-maps as fragments.

This procedure assumes that fragment default traffic classes were already created. The procedure for creating fragment default traffic classes is documented in the “Configuring a Fragment Traffic Class in a Policy-Map” section.

Like any policy-map, the configuration does not manage network traffic until it has been attached to an interface. This procedure does not cover the process of attaching a policy-map to an interface.

Note A service fragment can be used to collectively classify fragments only from the same physical interface. Fragments from different interfaces cannot be classified using the same service fragment.

Only queueing features are allowed in classes where the **service-fragment** keyword is entered, and at least one queueing feature must be entered in classes when the **service-fragment** keyword is used.

A policy-map with a class using the **service-fragment** keyword can be applied only to traffic leaving the interface (policy-maps attached to interfaces using the **service-policy output** command).

A class configured using the **service-fragment** keyword cannot be removed when it is being used to collectively apply QoS to fragments that are still configured on the interface. If you wish to remove a class configured using the **service-fragment** keyword, remove the fragment traffic classes before removing the service fragment.

The **service-fragment** keyword cannot be entered in a child policy-map.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name* **service-fragment** *fragment-class-name*
5. **shape average percent** *percent*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map BestEffortFragments`	Specifies the name of the traffic policy to configure and enters policy-map configuration mode.
Step 4	**class** *class-name* **service-fragment** *fragment-class-name* **Example:** `Device(config-pmap)# class data service-fragment BestEffort`	Specifies a class of traffic that is the composite of all fragments matching the *fragment-class-name*. The *fragment-class-name* when defining the fragments in other policy-maps must match the *fragment-class-name* in this command line to properly configure the service fragment class.

	Command or Action	Purpose
Step 5	**shape average percent** *percent* **Example:** `Device(config-pmap-c)# shape average percent 50`	Enters a QoS configuration command. Only queueing features are supported in default traffic classes configured as fragments. The queueing features that are supported are **bandwidth**, **shape**, and **random-detect exponential-weighting-constant**. Multiple QoS queueing commands can be entered.
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	Exits policy-map class configuration mode and returns to privileged EXEC mode.

Examples

Note This example shows a sample configuration that is supported in releases prior to Cisco IOS XE Release 2.6.

In the following example, a policy-map is created to apply QoS to all fragments named BestEffort.

```
policy-map main-interface
 class data service-fragment BestEffort
  shape average 400000000
```

In the following example, two fragments are created and then classified collectively using a service fragment.

```
policy-map subscriber1
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber 2
 class voice
  set cos 5
  priority level 1
 class video
  set cos 4
  priority level 2
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
```

Note This example shows a sample configuration that is supported in Cisco IOS XE Release 2.6 and later releases.

The following example shows the creation of two fragments called BestEffort in the subinterface policy-maps, followed by a sample configuration for the **service-fragment** called BestEffort to aggregate the queues at the main interface policy-map:

```
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map subscriber2
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
policy-map main-interface
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
```

Troubleshooting Tips

Ensure that all class statements that should be part of the same service fragment share the same *fragment-class-name*.

What to Do Next

Attach the service fragment traffic classes to the main physical interfaces.

Attach the fragment traffic classes to the member-link subinterfaces.

Configuring Service Fragments on a Physical Interface Supporting a Gigabit Etherchannel Bundle

Before you begin

This procedure assumes that a service fragment traffic class has already been created. A service fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy-Map" section. The procedure for creating a service fragment traffic classes is documented in the "Configuring a Service Fragment Traffic Class" section.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions document only the procedure for attaching a policy-map that already has a fragment traffic class to a member link subinterface.

Note For proper behavior, when a port-channel member link goes down, all member links should have the same policy-map applied.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *service-fragment-class-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service-policy configuration.

	Command or Action	Purpose
Step 4	**service-policy output** *service-fragment-class-name* **Example:** `Device(config-if)# service-policy output aggregate-member-link`	Attaches a service policy that contains a service fragment default traffic class to the physical Gigabit Ethernet interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Examples

In the following example, the policy-map aggregate-member-link is attached to the physical interface.

```
interface GigabitEthernet1/1/1
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 service-policy output aggregate-member-link
```

What to do next

Ensure that the fragment class name is consistent across service-fragment and fragment class definitions. Continue to the "Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces" section.

Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces

Before you begin

This procedure assumes that a service fragment traffic class has already been created. A service fragment traffic class cannot be configured without configuring a fragment class. The procedure for creating a fragment class is documented in the "Configuring a Fragment Traffic Class in a Policy-Map" section. The procedure for creating a service fragment traffic class is documented in the "Configuring a Service Fragment Traffic Class" section.

These instructions do not provide any details about the options that can be configured for Gigabit Etherchannel member link subinterfaces. These instructions only document the procedure for attaching a policy-map that already has a fragment traffic class to a member link subinterface.

Fragments cannot be used for traffic on two or more physical interfaces.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-interface-number* **.** *port-channel-subinterface-number*
4. **service-policy output** *fragment-class-name*

5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-interface-number* **.** *port-channel-subinterface-number* **Example:** `Device(config)# interface port-channel 1.100`	Enters subinterface configuration mode to configure an Etherchannel member link subinterface.
Step 4	**service-policy output** *fragment-class-name* **Example:** `Device(config-subif)# service-policy output subscriber`	Attaches a service policy that contains a fragment default traffic class to the Etherchannel member link subinterface
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named subscriber has a fragment default traffic class and is attached to the port-channel subinterface of an Etherchannel bundle.

```
interface port-channel 1.100
 service-policy output subscriber
```

Configuring Ingress Policing and Marking on Port-Channel Subinterface

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The Etherchannel member link interface should

already be configured to be part of the channel group (Etherchannel group). Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-number* **.** *port-channel-interface-number* **.** *sub-interface-number*
4. **service-policy input** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-number . port-channel-interface-number . sub-interface-number* **Example:** `Device(config)# interface port-channel 1.100.100`	Enters subinterface configuration mode to configure an Etherchannel member link subinterface.
Step 4	**service-policy input** *policy-map-name* **Example:** `Device(config-subif)# service-policy input sub-intf-input`	Specifies the name of the service policy that is applied to input traffic for the port-channel subinterface previously specified.
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named sub-intf-input is defined and attached to the port-channel subinterface in the input direction.

```
policy-map sub-intf-input
  class voice
    set precedence 5
  class video
    set precedence 6
  class class-default
    set precedence 3
!
interface Port-channel 1.100
 service-policy input sub-intf-input
```

Configuring Egress Policing and Marking on Port-Channel Member Links

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The Etherchannel member link interface should already be configured to be part of the channel group (Etherchannel group). Cisco IOS XE Release 2.1 or later software is required. The global configuration must contain the **port-channel load-balancing vlan-manual** command or the port-channel main-interface configuration must contain the **load-balancing vlan** command. It is assumed that these commands have already been executed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface port-channel** *port-channel-number . port-channel-interface-number . sub-interface-number*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface port-channel** *port-channel-number . port-channel-interface-number . sub-interface-number* **Example:** `Device(config)# interface port-channel 1.100.100`	Enters subinterface configuration mode to configure an Etherchannel member link subinterface.

	Command or Action	Purpose
Step 4	**service-policy output** *policy-map-name* **Example:** `Device(config-subif)# service-policy output WAN-GEC-member-Out-police`	Specifies the name of the service policy that is applied to output traffic for the Etherchannel member link subinterface specified in the previous step.
Step 5	**end** **Example:** `Device(config-subif)# end`	Exits subinterface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named WAN-GEC-member-Out-police is defined and attached to the port-channel subinterface in the output direction.

```
policy-map WAN-GEC-member-Out-police
  class voice
    set precedence 5
  class video
    set precedence 6
  class class-default
    set precedence 3
!
interface port-channel 1.100
 service-policy output WAN-GEC-member-Out-police
```

Configuring Policies Aggregation—MQC Support for Multiple Queue Aggregation at Main Interface

Before you begin

This feature is configured using the MQC. It is most useful in QoS configurations where several policy-maps attached to the same physical interface want aggregated treatment of multiple user-defined traffic classes from multiple port-channel subinterfaces. Cisco IOS XE Release 2.6 or later software is required. The global configuration must contain the following command: **port-channel load-balancing vlan-manual** or the main interface of the port-channel being configured must have the following command: **port-channel load-balancing vlan**. It is assumed that these commands have already been executed.

This feature is supported when policy-maps are attached to multiple port-channel subinterfaces and the port-channel member link interfaces. This feature cannot be used to collectively classify default traffic classes of policy-maps on different physical interfaces. It can collectively classify all traffic directed towards a given Port-channel member-link when designated by the **primary** or **secondary** directives on the sub-interface **encapsulation** command. The following items describe the behavior and restrictions on configuring this type of QoS Policy Aggregation with Etherchannel:

- Subinterface traffic classes without configured queuing features do not have queues at the subscriber level

- Default class traffic from multiple subinterfaces can be aggregated into a common policy-map at the main interface when you use the **fragment** keyword at the subinterface **class class-default** configuration, and **service-fragment** configuration at the main interface class
- This configuration additionally enables support for other subinterface traffic classes (such as DSCP-based classes) to be aggregated into a common policy-map at the main interface.
- This feature is enabled by using the **fragment** keyword in the subinterface **class-default** class, and **service-fragment** configuration in the main interface class (this also enables aggregation of the default class.
- Queuing features are not configured at the subinterface policy-map for the other traffic classes.
- Queuing occurs at the main interface policy-map for other subinterface traffic classes as an aggregate.
- Optional tracking of statistics is supported using the **account** command for other traffic classes in the subinterface policy-map.

A multistep process is involved with the complete configuration of QoS multiple queue aggregation at a main interface feature, as follows:

1. Configure default class statements as fragments in multiple subinterface policy-maps as described in the "Configuring a Fragment Traffic Class in a Policy-Map" section.
2. Configure a separate policy-map with a class statement using the **service-fragment** keyword in order to apply QoS to the class statements configured as fragments as described in the "Configuring a Service Fragment Traffic Class" section.
3. Configure service fragment traffic classes and attach them to the main physical interfaces as described in the "Configuring Service Fragments on a Physical Interface Supporting a Gigabit Etherchannel Bundle" section.
4. Configure fragment traffic classes and attach them to the member link subinterfaces as described in the "Configuring Fragments on Gigabit Etherchannel Member Link Subinterfaces" section.

Configuring MQC Queuing on Port-Channel Member Link—No Etherchannel Load Balancing

Before you begin

Traffic classes must be configured using the **class-map** command. A one or two level hierarchical policy-map should be configured using previously defined class maps.

Cisco IOS XE Release 2.4 or later software is required.

The port-channel main interface should also contain the following commands that create an active/standby scenario. Such a configuration will allow only a single interface to be active and forwarding traffic at any time.

- **interface Port-channel1**
- **lacp fast-switchover**
- **lacp max-bundle 1**

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service policy configuration.

Step 4	**service-policy output** *policy-map-name*
	Example:
	`Device(config-if)#`
	`service-policy output WAN-GEC-member-Out`
		Specifies the name of the service policy that is applied to output traffic.

Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named main-intf is defined and attached to the port-channel member links in the output direction.

```
interface Port-channel 1
 lcap fast-switchover
 lacp max-bundle 1
!
policy-map main-intf
 class voice
  priority
  police cir 10000000
```

```
 class video
  bandwidth remaining ratio 10
 class class-default
  bandwidth remaining ratio 3
!
interface GigabitEthernet0/0/0
 channel-group 1 mode active
 service-policy output main-intf
!
interface GigabitEthernet0/0/1
 channel-group 1 mode active
 service-policy output main-intf
```

Configuring MQC Queuing Configuration on Port-Channel Member Link—Etherchannel Load Balancing

Before you begin

Traffic classes must be configured using the **class-map** command. A one- or two-level hierarchical policy-map should be configured using previously defined class maps. The port-channel subinterface should have been previously configured with the appropriate encapsulation subcommand to match the select primary and secondary physical interfaces on the Etherchannel. Cisco IOS XE Release 2.5 or later software is required.

The Etherchannel setup may have multiple active interfaces with flow-based load balancing enabled.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface GigabitEthernet** *card/bay/port*
4. **service-policy output** *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface GigabitEthernet** *card/bay/port* **Example:** `Device(config)# interface GigabitEthernet 0/1/0`	Specifies the member link physical interface that receives the service policy configuration.

	Command or Action	Purpose
Step 4	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output WAN-GEC-member-Out`	Specifies the name of the service policy that is applied to output traffic.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Example

In the following example, the service policy named main-intf is defined and attached to the port-channel member links in the output direction.

```
  class voice
   priority
   police cir 10000000
  class video
   bandwidth remaining ratio 10
  class class-default
   bandwidth remaining ratio 3
!
interface GigabitEthernet0/0/0
 channel-group 1 mode active
 service-policy output main-intf
!
interface GigabitEthernet0/0/1
 channel-group 1 mode active
 service-policy output main-intf
```

Configuration Examples for QoS for Etherchannels

Example: Configuring QoS Policies Aggregation—Egress MQC Queuing at Subinterface

```
port-channel load-balancing vlan-manual
!
class-map match-all BestEffort
!
class-map match-all video
 match precedence 4
!
class-map match-all voice
 match precedence 5
!
```

```
policy-map subscriber
 class voice
  priority level 1
 class video
  priority level 2
 class class-default fragment BE
  shape average 100000000
  bandwidth remaining ratios 80

policy-map aggregate-member-link
 class BestEffort service-fragment BE
 shape average 100000000
!
interface Port-channel1
 ip address 209.165.200.225 255.255.0.0
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 209.165.200.226 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.200
 encapsulation dot1Q 200
 ip address 209.165.200.227 255.255.255.0
 service-policy output subscriber
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 209.165.200.228 255.255.255.0
 service-policy output subscriber
!
interface GigabitEthernet1/1/1
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
!
interface GigabitEthernet1/1/2
 no ip address
 channel-group 1 mode on
 service-policy output aggregate-member-link
```

Example: Configuring QoS Policies Aggregation—MQC Support for Multiple Queue Aggregation at Main Interface

```
port-channel load-balancing vlan-manual
!
policy-map subscriber1
 class voice
  set cos 5
  account
 class video
  set cos 4
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
!
policy-map subscriber2
```

```
 class voice
  set cos 2
  account
 class video
  set cos 3
  account
 class AF1
  account
 class class-default fragment BestEffort
  shape average 200000000
  bandwidth remaining ratio 10
!
policy-map main-interface-out
 class voice
  priority level 1
 class video
  priority level 2
 class AF1
  bandwidth remaining ratio 90
 class data service-fragment BestEffort
  shape average 400000000
  bandwidth remaining ratio 1
!
interface GigabitEthernet1/1/1
 no ip address
 channel-group 1 mode on
 service-policy output main-interface-out
!
interface GigabitEthernet1/1/2
 no ip address
 channel-group 1 mode on
 service-policy output main-interface-out
!
interface Port-channel1.100
 encapsulation dot1Q 100
 ip address 10.0.0.1 255.255.255.0
 service-policy output subscriber1
!
interface Port-channel1.200
 encapsulation dot1Q 200
 ip address 10.0.0.2 255.255.255.0
 service-policy output subscriber2
!
interface Port-channel1.300
 encapsulation dot1Q 300
 ip address 10.0.0.4 255.255.255.0
 service-policy output subscriber2
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular Quality of Service Command-Line Interface	"Applying QoS Features Using the MQC" module
Configuring RADIUS-based policing	*Intelligent Services Gateway Configuration Guide*
CISCO ASR 1000 Series software configuration	*Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Quality of Service for Etherchannel Interfaces

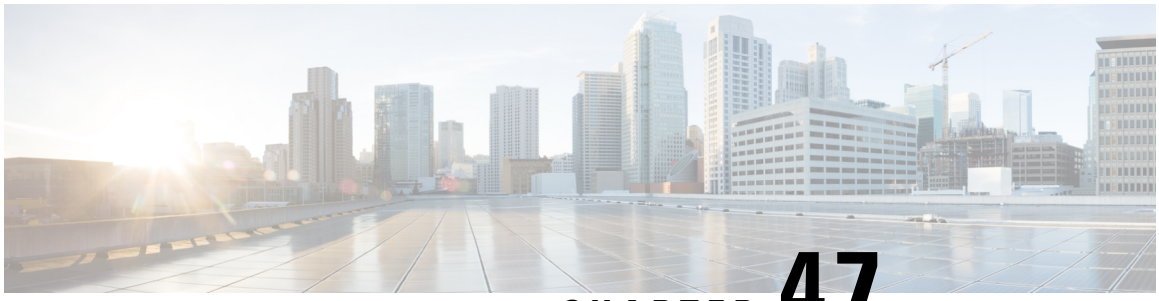
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to https://cfnng.cisco.com/. An account on Cisco.com is not required.

Table 53: Feature Information for Quality of Service for Etherchannel Interfaces

Feature Name	Releases	Feature Information
Egress MQC Queuing Configuration on Port-Channel Subinterface	Cisco IOS XE Release 2.1	This feature supports the configuration of Egress MQC queuing on port-channel subinterface. This feature was introduced on Cisco ASR 1000 Series Routers.

Feature Name	Releases	Feature Information
Egress MQC Queuing Configuration on Port-Channel Member Link	Cisco IOS XE Release 2.1	This feature supports the configuration of Egress MQC queuing on port-channel member link. This feature was introduced on Cisco ASR 1000 Series Routers.
QoS Policies Aggregation—Egress MQC Queuing at Subinterface	Cisco IOS XE Release 2.1	This feature supports the configuration of QoS Policies Aggregation - Egress MQC queuing at subinterface. This feature was introduced on Cisco ASR 1000 Series Routers.
Ingress Policing and Marking on Port-Channel Subinterface	Cisco IOS XE Release 2.1	This feature supports the configuration of Ingress Policing and Marking on port-channel subinterface. This feature was introduced on Cisco ASR 1000 Series Routers.
Egress Policing and Marking on Port-Channel Member Link	Cisco IOS XE Release 2.1	This feature supports the configuration of Egress policing and marking on port-channel member link. This feature was introduced on Cisco ASR 1000 Series Routers.
Egress MQC Queuing Configuration on Port-Channel Member Link - No Etherchannel Load Balancing	Cisco IOS XE Release 2.4	This feature supports the configuration of Egress MQC Queuing on Port-Channel Member Link - no Etherchannel Load Balancing. This feature was introduced on Cisco ASR 1000 Series Routers.
Egress MQC Queuing Configuration Supported on Port-Channel Member Link - Etherchannel Load Balancing	Cisco IOS XE Release 2.5	This feature supports the configuration of Egress MQC Queuing on Port-Channel Member Link - Etherchannel Load Balancing. This feature was introduced on Cisco ASR 1000 Series Routers.
QoS Policies Aggregation - MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface	Cisco IOS XE Release 2.6	This feature supports the configuration of QoS Policies Aggregation - MQC Support for Multiple Queue Aggregation at Main Interface - Egress MQC Queuing at Main Interface. This feature was introduced on Cisco ASR 1000 Series Routers.



CHAPTER 47

Applying QoS Features Using the MQC

About

This chapter provide an overview of Modular QoS CLI (MQC), which is how all QoS features are configured on the Cisco ASR 1000 Series Aggregation Services Router. MQC is a standardized approach to enabling QoS on Cisco routing and switching platforms.

We intend this chapter as an overview of configuration tasks required for any QoS configuration. Individual features are covered in appropriate modules.

Cisco Modular QoS CLI

With MQC, you perform 4 simple steps to enable and verify QoS. Examples are shown for each step. (Refer to individual chapters for feature explanations.)

1. **Create class-maps** - Classify your traffic (applications) into classes that you will work on.

```
class-map voice
  match dscp ef
class-map video
  match dscp AF41 AF42
```

2. **Create policy-map** - Define the treatment each class should receive.

```
policy-map simple-example
  class voice
    priority
    police cir percent 10
  class video
    bandwidth remaining percent 30
```

3. **Attach the policy-map** - Bind the policy to a physical or logical interface, identifying the traffic on which your policy should operate. You must specify whether the policy will apply to traffic that will enter the router via that interface (ingress) or to traffic that will exit the router via that interface (egress).

```
interface gigabitethernet1/0/0
  service-policy out simple-example
```

4. **Verify operation of the QoS policy** - Issue the `show policy-map interface` command to verify operation of all QoS features configured with the MQC.

```
show policy-map interface gigabitethernet1/0/0
```

Create Class Maps

When you create a class-map you are defining a group of applications that should receive similar treatment. You will specify a name for the group and subsequently use that name when defining the treatment they should receive.

You will need to define one or more filters (classification rules), establishing that a particular packet (application) belongs to the group you specified. When you create a class-map, you can decide whether a packet must match just one filter (*match-any*) or all filters (*match-all*) to be considered part of that group.

Create a class-map as follows:

```
class-map [match-all|match-any] <traffic-class-name>
  match….    Filter1
  match…..   Filter2
```

The following example illustrates a class where a packet need only match a single filter. If either the packet has the DSCP value ef **or** Cisco NBAR recognizes that the packet carries the skype application, then we consider the packet *as belonging to the voice class*. We use the name voice in a policy-map to define treatment for any packet classified as belonging to this class:

```
class-map match-any voice
  match dscp ef
  match protocol skype
```

In the following example, we employ the match-all semantic: a packet must match all filters to belong to a class. We mandate that traffic must be recognized as MAPI (using Cisco NBAR) and also be to or from the address specified in the access list:

```
ip access-list extended mail-server-addr
  permit ip any host 10.10.10.1
  permit ip host 10.10.10.1 any
!
class-map match-all work-email
  match protocol mapi
  match access-group name mail-server-addr
```

The previous examples illustrate the flexibility of filter definitions on the ASR 1000 series platform. Filters can be based on marks in the packet header (precedence, DSCP, Exp or COS), access-lists, Cisco NBAR (match protocol *xxx*) or internal markings like qos-group. (Refer to the classification chapter for a more complete description of supported filters - when available.)

For convenience, you can also include other class-maps as filters in a class-map:

```
class-map broadcast-video
  match dscp cs5
class-map multimedia-streaming
  match dscp af31 af32 af33
class-map multimedia-conferencing
  match dscp af41 af42 af43
class-map realtime-interactive
  match dscp cs4
!
class-map match-any all-video
  match class broadcast-video
  match class multimedia-streaming
  match class multimedia conferencing
  match class realtime-interactive
!
class-map match-any interactive-video
  match class multimedia conferencing
  match class realtime-interactive
```

In this example we use *nested class-maps* in the definition of classes all-video and interactive-video.

By definition, a particular packet might match the classification criteria of multiple classes in a class-map. If so, the order in which classes are defined in a policy-map determines which class the packet belongs to; a packet belongs to the first class it matches.

Create Policy-Maps

A policy-map is how you specify what actions should apply to each class of traffic you create.

Let's re-examine the simple example above:

```
policy-map simple-example
  class voice
    priority
    police cir percent 10
  class video
    bandwidth remaining percent 30
```

The policy-map name is simple-example – this is the name we use when we subsequently attach the policy to one or more interfaces. The policy itself is quite readable – we have defined two classes of traffic: voice and video. Voice traffic should receive priority (low latency) scheduling but throughput of that class is limited to 10% of the interface bandwidth. For video traffic, we have a dedicated queue and a guarantee of 30% of what remains after voice is serviced.

The above policy-map has a 3rd implicit class; class-default is the last class in a policy, whether explicitly configured or not. It is a catch-all into which falls any traffic that does not match one of the user-defined classes. In egress policies class-default will have its own queue and an implicit bandwidth remaining ratio of 1. If bandwidth values are specified in percentage, class-default will receive any unassigned percent (see asterisks). Knowing this the above policy-map would actually look as follows:

```
class-map class-default
  match any
!
policy-map simple-example
  class voice
    priority
    police cir percent 10
```

```
  class video
    bandwidth remaining percent 30
  class class-default
    bandwidth remaining percent 70                ****
```

Note You never need to create a class-map for class-default. We visualize it here to provide a better understanding of how the policy works. If a packet does not match the voice class or the video class it will always match class-default.

Examples of actions in the policy above include the **priority**, **police**, and **bandwidth** commands. Actions function as control knobs to differentiate how one class of traffic will be treated vs. another.

One very important differentiation when looking at actions is queuing vs. non-queuing. What if we now add one more class to the simple-example policy-map:

```
class-map youtube
  match protocol youtube
  !
policy-map simple-example
  class voice
    priority
    police cir percent 10
  class youtube
    police cir percent 5
  class video
    bandwidth remaining percent 30
```

We have added a third user-defined class named youtube that is rate-limiting YouTube traffic such that it can never exceed 5% of link capacity. As this class has no queuing action configured, no queue is created (see below the list of actions that create a queue). Packets that match this class (those with protocol youtube) will traverse the policer and then be enqueued in the class-default queue.

Did you notice that we placed the youtube class before the video class in our policy definition? We want to ensure that youtube traffic is always part of this class rather than our video class. By defining this class earlier in the policy-map we will check for a match to this class before we check the video class criteria.

The specific actions that will create a queue are **priority**, **bandwidth**, **bandwidth remaining** and **shape**. Other actions like **fair-queue**, **queue-limit** and **random-detect** may only be used in a class already containing one of the actions that creates a queue. The actions **police** and **set** will not create a queue, although you can use the **police** command for queue admission control.

One key reason to differentiate between queuing actions and non-queuing actions is that a policy-map that will be applied to ingress traffic may not contain any queuing actions on the ASR 1000 Series Router. Let's summarize which actions are queuing and which are not:

Queuing and Non-Queuing Actions		**Action**
Queuing Actions		
	Scheduling	
		priority
		bandwidth

Queuing and Non-Queuing Actions		Action
		bandwidth remaining
		shape
		fair-queue
	Queue Management / Congestion Avoidance	
		queue-limit
		random-detect
Non-Queuing Actions		
	Rate-Limiting / Admission Control	
		police
	Marking	
		set

Hierarchical policy-maps can be created by embedding a policy-map within a class of another policy-map:

```
policy-map child
  class voice
    priority
    police cir percent 10
  class video
    bandwidth remaining percent 30
!
policy-map parent-vlan
  class class-default
    shape average 100m
    service-policy child
```

A common use is to create a *shape on parent* / *queue on child* policy that can be attached to a logical interface such as a VLAN or a tunnel.

From a classification perspective, a packet must adhere to the classification criteria of the child as well as the parent class to be considered a member of a particular child class. In this example the parent class is class-default and by definition any traffic will match this class.

When defining hierarchical polices we can re-use policy-maps for convenience.

In the following example, we use the policy-map named child in both parent-vlan100 and parent-vlan200. When instantiated (attached to an interface) the voice class in parent-vlan100 will be limited to 10 Mbps (10% of 100m parent shaper) while the voice class in parent-vlan200 will be limited to 5 Mbps (10% of 50m parent shaper):

```
policy-map child
  class voice
    priority
    police cir percent 10
```

```
  class video
    bandwidth remaining percent 30
!
policy-map parent-vlan100
  class class-default
    shape average 100m
    service-policy child
!
policy-map parent-vlan200
  class class-default
    shape average 50m
    service-policy child
!
int gigabitethernet1/0/0.100
  service-policy out parent-vlan100
int gigabitethernet1/0/0.200
  service-policy out parent-vlan200
```

This example shows that although the definition may be shared the instances of the policy on different interfaces are truly unique.

You can also create hierarchical policies with policy-maps used in user-defined classes.

The following example illustrates a 3-level hierarchical policy, the max currently supported on the ASR 1000 Series Router. For a packet to match a class at the application level it must now match 3 requirements: the voice or video classifier at the child, the vlan classifier in the vlan-sharing policy-map, and the class-default (anything) in the physical level policy-map:

```
class-map vlan100
  match vlan 100
class-map vlan200
  match vlan 200
!
policy-map child
  class voice
    priority
    police cir percent 10
  class video
    bandwidth remaining percent 30
!
policy-map vlan-sharing
  class vlan100
    shape average 100m
    service-policy child
  class vlan200
    shape average 50m
    service-policy child
!
policy-map physical-policy
  class class-default
    shape average 500m
    service-policy vlan-sharing
!
interface gigabitethernet1/0/0
  service-policy out physical-policy
```

When you create a policy-map, IOS will perform some error checking on the policy. For example, if I create a policy with an unconstrained priority queue and then guarantee bandwidth to another queue, IOS will recognize the disconnect; if the unconstrained priority queue can comsume the entire interface bandwidth then clearly you cannot guarantee any of that bandwidth to another queue:

```
policy-map create-error-example
  class unconstrained-priority
    priority
  class bandwidth-guarantee
    bandwidth percent 50
```

If IOS detects an error in the policy during creation, it will reject the configuration and display an error at that time.

Attach the Policy-Map

The third step in using the Cisco MQC is to *instantiate the policy-map* (ie., to attach the policy to an interface and thus initiate control of traffic). We use the **service-policy** command to attach the policy and also to specify whether it is acting on traffic ingressing that interface or egressing that interface:

```
interface gigabitethernet1/0/0
  service-policy out simple-example
```

We have already mentioned that queuing policies are only supported for egress traffic (**service-policy out** *policy-name*) but policies that contain only non-queuing actions may be attached for ingress (**service-policy in** *policy-name*) or egress traffic.

We term the interface to where we apply the **service-policy** command the *attach point*. This point could be a physical interface (such as an Ethernet interface or a T1 interface) or it could be a logical interface such as a VLAN sub-interface or a tunnel interface.

When a policy-map contains queuing actions but no hierarchical policies we refer to the policy as a *flat policy*. A flat policy may only be attached to a physical interface.

To attach a queuing policy to a logical interface, you must use a hierarchical shape on parent/queue on child style policy.

As you recall, error checking occurs when you create a policy. A second round of error checking occurs when you attach the policy to an interface. For example, I might create a policy with bandwidth guarantees that can't be realized on a particular type of interface. The policy-map may be valid when defined but when combined with information about the attachment interface, IOS can recognize the error as in the following example:

```
policy-map attach-error-example
  class bulk-data
    bandwidth 200000
```

This policy dictates that 200 Mbps should be reserved for bulk-data. If I attach this policy to a GigabitEthernet interface it should work fine. However, if I attach this policy to a POS OC3 interface it will be rejected at attach time. An OC3 interface has a nominal bandwidth of 155 Mbps. 200 Mbps could never be reserved for a particular class of traffic.

Verify Operation of the QoS Policy

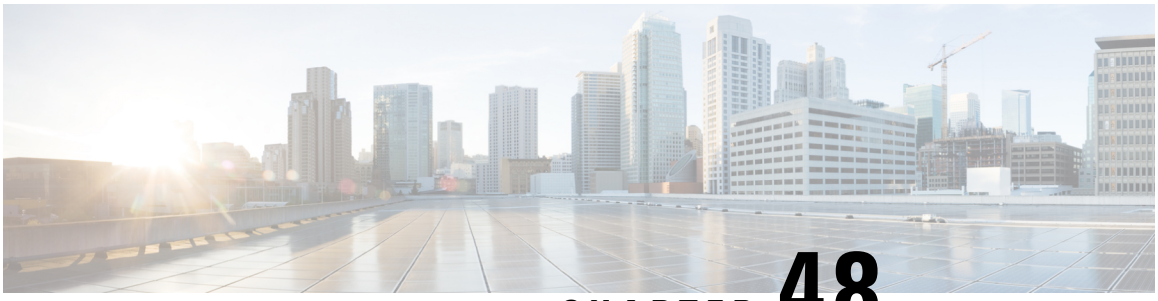
One command is always available to verify the operation of any QoS policy:

show policy-map interface *interface-name*

The output of this command displays a section for each class in a policy-map. It also shows statistics for packets and bytes classified as belonging to that class as well as for each action configured in the class.

Note The statistics available from this command are also available via SNMP in the CISCO-CLASS-BASED-QOS-MIB.

If the QoS policy is attached to a multipoint interface such as DMVPN, we use the **show policy-map multipoint tunnel** *tunnel-number* variant of the command. Similarly if the policy is attached to a broadband session, we would use the **show policy-map session uid** *session-number* variant of the command.



CHAPTER 48

Classifying Network Traffic Using NBAR

Network-Based Application Recognition (NBAR) is a classification engine that recognizes and classifies a wide variety of protocols and applications. When NBAR recognizes and classifies a protocol or an application, you can configure the network to apply the appropriate quality of service (QoS) for that application or traffic with the classified protocol.

This module contains an overview of classifying network traffic using NBAR.

Restrictions for Classifying Network Traffic Using NBAR

NBAR does not support the following applications:

- Non-IP traffic.
- Multiprotocol Label Switching (MPLS)-labeled packets. NBAR classifies only IP packets. You can, however, use NBAR to classify IP traffic before the traffic is handed over to MPLS. Use the modular QoS CLI (MQC) to set the IP differentiated services code point (DSCP) field on NBAR-classified packets and make MPLS map the DSCP setting to the MPLS experimental (EXP) setting inside the MPLS header.
- NBAR processing. By design, NBAR processing is temporarily disabled during the In-Service Software Upgrade (ISSU). The following syslog message indicates the restart of the NBAR classification once ISSU is complete: “%NBAR_HA-5-NBAR_INFO: NBAR sync DONE!.”
- Multicast packet classification.
- Asymmetric flows with stateful protocols.
- Packets that originate from or destined to a device running NBAR.

Note In the NBAR context, asymmetric flows are flows in which different packets go through different devices, for reasons such as load balancing implementation or asymmetric routing, where packets flow through different routes in different directions.

NBAR is not supported on the following logical interfaces:

- Dialer interfaces
- Dynamic tunnels such as Dynamic Virtual Tunnel Interface (DVTI)
- Fast Etherchannels
- IPv6 tunnels that terminate on the device
- MPLS
- Overlay Transport Virtualization (OTV) overlay interfaces

Note In cases where encapsulation is not supported by NBAR on some links, you can apply NBAR on other interfaces of the device to perform input classification. For example, you can configure NBAR on LAN interfaces to classify output traffic on the WAN link.

The following virtual interfaces are supported depending on the image of your Cisco IOS:

- Generic routing encapsulation (GRE)
- IPsec IPv4 tunnel (including tunneled IPv6) in protocol discovery mode and MQC mode
- IPsec IPv6 tunnel in protocol discovery mode but not in MQC mode
- Multipoint GRE/Dynamic Multipoint VPN (DMVPN) in protocol discovery mode

Note NBAR requires more CPU power when NBAR is enabled on tunneled interfaces.

Note From Cisco IOS 15.5(3)M, NBAR functionality will not be supported on IOS, if you are impacted by this change, we recommend that you consider IOS XE as an alternative solution.

If protocol discovery is enabled on both the tunnel interface and the physical interface on which the tunnel interface is configured, the packets that are designated to the tunnel interface are counted on both interfaces. On the physical interface, the packets are classified and are counted based on the encapsulation. On the tunnel interface, packets are classified and are counted based on the Layer 7 protocol.

For all protocols, only 16 combinations of subclassification per protocol can be configured. You can define a combination for subclassification using the **match protocol** *protocol-name variable-field-name value* command.

NBAR and Sub-classification of Modbus Protocol

Modbus is a serial communications protocol (messaging structure) originally published by Modicon in 1979 for using with its programmable logic controllers (PLCs). This is a standard communication protocol which is commonly used for connecting industrial electronic devices. MODBUS is currently implemented using few modes and one of them is TCP/IP over Ethernet (Supported in NBAR).

The following are the modbus sub-classification options:

Table 54: Modbus Sub-classification Options

Sub-classification Field	Description
encapsulated-transport	Sub-classify Modbus function encapsulated transport
exception-response	Sub-classify Modbus function exception
mask-write-register	Sub-classify Modbus function mask_write_register
read-FIFO-Queue	Sub-classify Modbus function read FIFO Queue
read-coils	Sub-classify Modbus function read coils
read-discrete-input	Sub-classify Modbus function read discrete input
read-exception-status	Sub-classify Modbus function read exception status
read-file-record	Sub-classify Modbus function read file record
read-holding-registers	Sub-classify Modbus function read holding registers
read-or-write-registers	Sub-classify Modbus function read or write registers
write-file-record	Sub-classify Modbus function write file record
write-multiple-coils	Sub-classify Modbus function write multiple coils
write-multiple-registers	Sub-classify Modbus function write multiple registers
write-single-coil	Sub-classify Modbus function write single coil
write-single-register	Sub-classify Modbus function write single register

NBAR Functionality

NBAR is a classification engine that recognizes and classifies a wide variety of protocols and applications, including web-based and other difficult-to-classify applications and protocols that use dynamic TCP/UDP port assignments.

When NBAR recognizes and classifies a protocol or an application, the network can be configured to apply the appropriate QoS for that application or traffic with that protocol. The QoS is applied using the MQC.

Note For more information about the MQC, see the "Applying QoS Features Using the MQC" module.

NBAR introduces several classification features that identify applications and protocols from Layer 4 through Layer 7. These classification features are as follows:

- Statically assigned TCP and UDP port numbers.
- Non-TCP and non-UDP IP protocols.
- Dynamically assigned TCP and UDP port numbers. This kind of classification requires stateful inspection, that is, the ability to inspect a protocol across multiple packets during packet classification.
- Subport classification or classification based on deep packet inspection, that is, classification for inspecting packets.

Note Access Control Lists (ACLs) can also be used for classifying static port protocols. However, NBAR is easier to configure and can provide classification statistics that are not available when ACLs are used.

NBAR includes a Protocol Discovery feature that provides an easy way to discover application protocols that are operating on an interface. For more information about Protocol Discovery, see the "Enabling Protocol Discovery" module.

Note NBAR classifies network traffic by application or protocol. Network traffic can be classified without using NBAR. For information about classifying network traffic without using NBAR, see the "Classifying Network Traffic" module.

NBAR includes the Protocol Pack feature that provides an easy way to load protocols and helps NBAR recognize additional protocols for network traffic classification. A protocol pack is set a of protocols developed and packed together. A new protocol pack can be loaded on the device to replace the default IOS protocol pack that is already present in the device.

NBAR Benefits

Identifying and classifying network traffic is an important first step in implementing QoS. A network administrator can more effectively implement QoS in a networking environment after identifying the number and types of applications and protocols that are running on a network.

NBAR gives network administrators the ability to see the different types of protocols and the amount of traffic generated by each protocol. After NBAR gathers this information, users can organize traffic into classes. These classes can then be used to provide different levels of service for network traffic, thereby allowing better network management by providing the appropriate level of network resources for the network traffic.

NBAR is also used in Cisco Application Visibility and Control (AVC). With AVC, NBAR provides better application performance through better QoS and policing, and provides finer visibility about the network that is being used.

With AVC license, the following NBAR features are supported:

- Classification inside transient IPv6 tunnels
- Custom protocols
- Customization of protocol attributes
- Field extraction
- Protocol pack updates

NBAR and Classification of HTTP Traffic

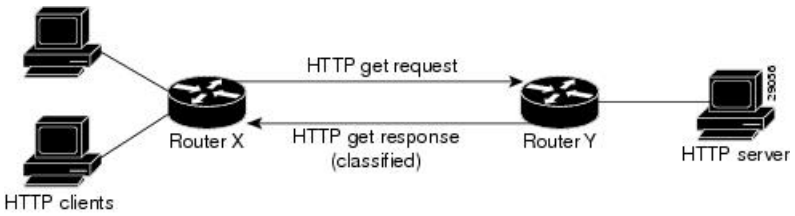
Classification of HTTP Traffic by a URL Host or MIME

NBAR can classify application traffic by looking beyond the TCP/UDP port numbers of a packet. This is called subport classification. NBAR looks into the TCP/UDP payload itself and classifies packets based on content, such as the transaction identifier, message type, or other similar data, within the payload.

Classification of HTTP traffic by a URL, a host, or a Multipurpose Internet Mail Extension (MIME) type is an example of subport classification. NBAR classifies HTTP traffic by the text within the URL or host fields of a request by using regular expression matching. HTTP client request matching in NBAR supports most HTTP request methods such as GET, PUT, HEAD, POST, DELETE, OPTIONS, CONNECT, and TRACE. The NBAR engine then converts the specified match string into a regular expression.

The figure below illustrates a network topology with NBAR in which Device Y is the NBAR-enabled device.

Figure 92: Network Topology with an NBAR-enabled Device



When specifying a URL for classification, include only the portion of the URL that follows the www.*hostname.domain* in the **match** statement. For example, for the URL www.cisco.com/latest/whatsnew.html, include only /latest/whatsnew.html with the **match** statement (for instance, **match protocol http url /latest/whatsnew.html**).

Host specifications are identical to URL specifications. NBAR performs a regular expression match on the host field contents inside an HTTP packet and classifies all packets from that host. For example, for the URL www.cisco.com/latest/whatsnew.html, include only www.cisco.com.

For MIME type matching, the MIME type can contain any user-specified text string. A list of the Internet Assigned Numbers Authority (IANA) supported MIME types can be found at the following URL:

http://www.iana.org/assignments/media-types/

When matching by MIME type, NBAR matches a packet containing the MIME type and all subsequent packets until the next HTTP transaction.

NBAR supports URL and host classification in the presence of persistent HTTP. NBAR does not classify packets that are part of a pipelined request. With pipelined requests, multiple requests are pipelined to the server before previous requests are serviced. Pipelined requests are not supported with subclassification and tunneled protocols that use HTTP as the transport protocol.

The NBAR Extended Inspection for HTTP Traffic feature allows NBAR to scan TCP ports that are not well known and to identify HTTP traffic that traverses these ports. HTTP traffic classification is no longer limited to the well-known and defined TCP ports.

Depending on your release, the Enable NBAR URI Extraction for HTTP Transactions for Persistent Connections feature supports extraction and export of the URL field per transaction, and not only the URL of the first transaction as supported in earlier releases. To enable multi-transaction, a protocol pack with 'Enhanced Web Classification' has to be installed. When an Enhanced Web Classification protocol pack is installed, the **match connection transaction-id** command configuration in flexible netflow tracks multiple HTTP transactions. For more information on tracking HTTP transactions, refer to *Cisco IOS Flexible NetFlow Configuration Guide*.

Note NBAR performs significant additional tasks for classification and export per transaction. These tasks impact performance and may cause increased export rate.

Classification of HTTP Traffic by Using HTTP Header Fields

NBAR introduces expanded ability for users to classify HTTP traffic by using information in the HTTP header fields.

HTTP works using a client/server model. HTTP clients open connections by sending a request message to an HTTP server. The HTTP server then returns a response message to the HTTP client (this response message is typically the resource requested in the request message from the HTTP client). After delivering the response, the HTTP server closes the connection and the transaction is complete.

HTTP header fields are used to provide information about HTTP request and response messages. HTTP has numerous header fields. For additional information on HTTP headers, see section 14 of RFC 2616: *Hypertext Transfer Protocol—HTTP/1.1*. This RFC can be found at the following URL:

http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html

NBAR is able to classify the following HTTP header fields:

- For request messages (client-to-server), the following HTTP header fields can be identified using NBAR:
  - User-Agent
  - Referrer
  - From
- For response messages (server to client), the following HTTP header fields can be identified using NBAR:
  - Server
  - Location
  - Content-Base
  - Content-Encoding

Within NBAR, the **match protocol http c-header-field** command is used to specify that NBAR identify request messages (the “c” in the **c-header-field** portion of the command is for client). The **match protocol http s-header-field** command is used to specify response messages (the “s” in the **s-header-field** portion of the command is for server).

Note The **c-header-field** and **s-header-field** keywords and associated arguments in the **match protocol http** command are no longer available. The same functionality is achieved by using the individual keywords and arguments. For more information, see the syntax of the **match protocol http** command in the *Quality of Service Solutions Command Reference*.

Note The c-header-field performs subclassifications based on a single value in the user-agent, the referrer, or from-header field values. The s-header-field performs subclassifications based on a single value in the server, location, content-encoding, or content-base header field values. These header field values are not related to each other. Hence, the c-header and s-header fields are replaced by the user-agent, referrer, from, server, content-base, content-encoding, and location parameters as per the intent and need of HTTP subclassification.

Combinations of Classification of HTTP Headers and URL Host or MIME Type to Identify HTTP Traffic

Note that combinations of URL, Host, MIME type, and HTTP headers can be used during NBAR configuration. These combinations provide customers with more flexibility to classify specific HTTP traffic based on their network requirements.

NBAR and Classification of Citrix ICA Traffic

NBAR can classify Citrix Independent Computing Architecture (ICA) traffic and perform subport classification of Citrix traffic based on the published application name or ICA tag number.

Classification of Citrix ICA Traffic by Published Application Name

NBAR can monitor Citrix ICA client requests for a published application that is destined to a Citrix ICA Master browser. After the client requests the published application, the Citrix ICA master browser directs the client to the server with the most available memory. The Citrix ICA client then connects to this Citrix ICA server for the application.

Note For Citrix to monitor and classify traffic by the published application name, use Server Browser Mode on the master browser.

In server browser mode, NBAR statefully tracks and monitors traffic and performs a regular expression search on the packet contents for the published application name specified by the **match protocol citrix** command. The published application name is specified by using the **app** keyword and the *application-name-string* argument of the **match protocol citrix** command. For more information about the **match protocol citrix** command, see the Quality of Service Solutions Command Reference.

The Citrix ICA session triggered to carry the specified application is cached, and traffic is classified appropriately for the published application name.

Citrix ICA Client Modes

Citrix ICA clients can be configured in various modes. NBAR cannot distinguish among Citrix applications in all modes of operation. Therefore, network administrators might need to collaborate with Citrix administrators to ensure that NBAR properly classifies Citrix traffic.

A Citrix administrator can configure Citrix to publish Citrix applications individually or in Published Desktop Mode. In the Published Desktop Mode of operation, all applications within the published desktop of a client use the same TCP session. Therefore, differentiation among applications is impossible, and NBAR can be used to classify Citrix applications only as aggregates (by looking at port 1494).

The Published Application Mode for Citrix ICA clients is recommended when you use NBAR. In Published Application Mode, a Citrix administrator can configure a Citrix client in either Seamless or Nonseamless (windows) modes of operation. In Nonseamless Mode, each Citrix application uses a separate TCP connection, and NBAR can be used to provide interapplication differentiation based on the name of the published application.

Seamless Mode clients can operate in one of two submodes: session sharing or nonsession sharing. In seamless session sharing mode, all clients share the same TCP connection, and NBAR is not able to differentiate among applications. Seamless sharing mode is enabled by default in some software releases. In seamless nonsession sharing mode, each application for each client uses a separate TCP connection. NBAR can provide interapplication differentiation in seamless nonsession sharing mode.

Note NBAR operates properly in Citrix ICA secure mode. Pipelined Citrix ICA client requests are not supported.

NBAR and Sub-classification of Modbus Protocol

Modbus is a serial communications protocol (messaging structure) originally published by Modicon in 1979 for using with its programmable logic controllers (PLCs). This is a standard communication protocol which is commonly used for connecting industrial electronic devices. MODBUS is currently implemented using few modes and one of them is TCP/IP over Ethernet (Supported in NBAR).

The following are the modbus sub-classification options:

Table 55: Modbus Sub-classification Options

Sub-classification Field	Description
encapsulated-transport	Sub-classify Modbus function encapsulated transport
exception-response	Sub-classify Modbus function exception
mask-write-register	Sub-classify Modbus function mask_write_register
read-FIFO-Queue	Sub-classify Modbus function read FIFO Queue
read-coils	Sub-classify Modbus function read coils
read-discrete-input	Sub-classify Modbus function read discrete input
read-exception-status	Sub-classify Modbus function read exception status
read-file-record	Sub-classify Modbus function read file record

Sub-classification Field	Description
read-holding-registers	Sub-classify Modbus function read holding registers
read-or-write-registers	Sub-classify Modbus function read or write registers
write-file-record	Sub-classify Modbus function write file record
write-multiple-coils	Sub-classify Modbus function write multiple coils
write-multiple-registers	Sub-classify Modbus function write multiple registers
write-single-coil	Sub-classify Modbus function write single coil
write-single-register	Sub-classify Modbus function write single register

Classification of Citrix ICA Traffic by ICA Tag Number

Citrix uses a TCP session each time an application is opened. In the TCP session, a variety of Citrix traffic may be intermingled in the same session. For example, print traffic may be intermingled with interactive traffic, causing interruption and delay for a particular application.

Most users would prefer printing to be handled as a background process that does not interfere with the processing of higher-priority traffic. To accommodate this printing preference, the Citrix ICA protocol includes the ability to identify Citrix ICA traffic based on the ICA tag number of the packet. The ability to identify, tag, and prioritize Citrix ICA traffic is referred to as ICA Priority Packet Tagging. With ICA Priority Packet Tagging, Citrix ICA traffic is categorized as high, medium, low, and background, depending on the ICA tag of the packet.

When ICA traffic priority tag numbers are used, and the priority of the traffic is determined, QoS features can be implemented to determine how the traffic will be handled. For example, QoS traffic policing can be configured to transmit or drop packets with a specific priority.

Citrix ICA Packet Tagging

The Citrix ICA tag is included in the first two bytes of the Citrix ICA packet, after the initial negotiations are completed between the Citrix client and server.

The first two bytes of the packet (byte 1 and byte 2) contain the byte count and the ICA priority tag number. Byte 1 contains the low-order byte count, and the first two bits of byte 2 contain the priority tags. The other six bits contain the high-order byte count.

The ICA priority tag value can be a number from 0 to 3. The number indicates the packet priority, with 0 being the highest priority and 3 being the lowest priority.

To prioritize Citrix traffic by the ICA tag number of the packet, you must specify the tag number using the **ica-tag** keyword and the *ica-tag-value* argument of the **match protocol citrix** command. For more information about the **match protocol citrix** command, see the Quality of Service Solutions Command Reference.

The table below contains information about different Citrix traffic and the respective priority tags.

Table 56: Citrix ICA Packet Tagging

Priority	ICA Bits (decimal)	Sample Virtual Channels
High	0	Video, mouse, and keyboard screen updates

Priority	ICA Bits (decimal)	Sample Virtual Channels
Medium	1	Program neighborhood, clipboard, audio mapping, and license management
Low	2	Client common equipment (COM) port mapping and client drive mapping
Background	3	Auto client update, client printer mapping, and original equipment manufacturers (OEM) channels

NBAR and RTP Payload Type Classification

Real-time Transport Protocol (RTP) is a packet format for multimedia data streams. It can be used for media-on-demand and for interactive services such as Internet telephony. RTP consists of a data part and a control part. The control part is called Real-Time Transport Control Protocol (RTCP). RTCP is a separate protocol that is supported by NBAR. It is important to note that the NBAR RTP Payload Type Classification feature does not identify RTCP packets and that RTCP packets run on odd-numbered ports and RTP packets run on even-numbered ports.

The data part of RTP is a thin protocol that provides support for applications with real-time properties such as continuous media (audio and video), which includes timing reconstruction, loss detection, and security and content identification. RTP is discussed in RFC 1889 (*A Transport Protocol for Real-Time Applications*) and RFC 1890 (*RTP Profile for Audio and Video Conferences with Minimal Control*).

The RTP payload type is the data transported by RTP in a packet, for example, audio samples or compressed video data.

The NBAR RTP Payload Type Classification feature not only allows real-time audio and video traffic to be statefully identified, but can also differentiate on the basis of audio and video codecs to provide more granular QoS. The RTP Payload Type Classification feature, therefore, does a deep-packet inspection into the RTP header to classify RTP packets.

For more information on the classification of RTP with NBAR, see NBAR RTP Payload Classification.

NBAR and Classification of Custom Protocols and Applications

NBAR supports the use of custom protocols to identify custom applications. Custom protocols support static port-based protocols and applications that NBAR does not currently support. You can add to the set of protocols and application types that NBAR recognizes by creating custom protocols.

Custom protocols extend the capability of NBAR Protocol Discovery to classify and monitor additional static port applications and allow NBAR to classify nonsupported static port traffic.

Once the custom protocols are defined, you can then use them with the help of NBAR Protocol Discovery and the MQC to classify the traffic.

With NBAR supporting the use of custom protocols, NBAR can map static TCP and UDP port numbers to the custom protocols.

There are two types of custom protocols:

- Predefined custom protocols
- User-defined custom protocols

NBAR includes the following characteristics related to predefined custom protocols and applications:

- Custom protocols have to be named custom-xx, with xx being a number.
- Ten custom applications can be assigned using NBAR, and each custom application can have up to 16 TCP and 16 UDP ports each mapped to the individual custom protocol. The real-time statistics of each custom protocol can be monitored using Protocol Discovery.
- After creating a variable when creating a custom protocol, you can use the **match protocol** commandto classify traffic on the basis of a specific value in the custom protocol.

NBAR includes the following characteristics related to user-defined custom protocols and applications:

- The ability to inspect the payload for certain matching string patterns at a specific offset.
- The ability to allow users to define the names of their custom protocol applications. The user-named protocol can then be used by Protocol Discovery, the Protocol Discovery MIB, and the **match protocol**command as an NBAR-supported protocol.
- The ability of NBAR to inspect custom protocols specified by traffic direction (that is, traffic heading toward a source or destination rather than traffic in both directions), if desired by the user.
- CLI support that allows a user configuring a custom application to specify a range of ports rather than to specify each port individually.
- The **variable**keyword, the *field-name*argument,and the *field-length* argument were added to the **ip nbar custom**command.
- The **http** keyword group that lets you add custom host and URL signatures.

This additional keyword and two additional arguments allow for creation of more than one custom protocol based on the same port numbers.

Note Defining a user-defined custom protocol restarts the NBAR feature, whereas defining predefined custom protocol does not restart the NBAR feature.

NBAR DNS-based Classification

NBAR can improve traffic classification by using DNS transaction information exchanged when a user initiates a connection with an application server. This method offers the significant advantage of classifying flows from the first packet in the flow.

To illustrate, when a web-based application is opened in a browser, the browser first communicates with a DNS server to request the IP address of the relevant server for the application. The DNS transaction consists of a request and response; the response contains the IP address of the server for the web-based application.

Using information from this transaction, NBAR can correctly associate the web-based application with the relevant server IP address. NBAR can then identify future traffic involving that IP address from the first packet of the flow.

Supported Platforms

This feature is supported on platforms operating Cisco IOS XE, beginning with Cisco IOS XE release 3.17S, and including IOS XE Denali 16.x.

Advantages

NBAR applies multiple methods to classifying traffic, including in some cases, classifying traffic from the first packet, such as by socket-cache. The DNS-based classification feature operates with other NBAR methods to improve traffic classification. It is especially helpful for certain specific types of traffic, including asymmetric server-to-client flows, as well as some types of encrypted traffic.

Complementarity with Other NBAR Classification Methods

In general, the NBAR engine uses numerous strategies together to provide the most granular possible classification of traffic. First-packet classification may occur by multiple methods, including DNS-based classification and socket-cache. Additional classification methods may then add greater granularity to the classification.

Limitations

- Identification by DNS transaction information is insufficient in some situations. In these cases, NBAR relies on other methods to classify the traffic, where possible. For example, this method does not function well with generic hosts or service aggregation. (In the case of generic hosts or service aggregation, numerous services are hosted through a single server IP address, either using the same host name or different host names.)
- In some cases, NBAR may not be have access to the DNS transaction data for some traffic. For example, a network topology might include a local DNS server accessed through a connection not monitored by NBAR. DNS-based classification is not possible in these cases.

Limiting or Disabling DNS-based Classification

DNS-based classification may be disabled (see Enabling and Disabling DNS-based Classification, on page 696).

Typically, it is recommended to leave the DNS Guard feature in its default enabled state, which limits DNS-based Categorization to operating only when the complete DNS transaction (request, response) is available, but in special cases, it can be disabled (see Enabling and Disabling DNS Guard for DNS-based Categorization, on page 696).

Related Functionality

In addition to the DNS-based classification feature, NBAR has other methods that can, in some cases, provide first packet classification of traffic.

Customized server specification. This feature operates on all platforms that support NBAR, including those that do not support the DNS-based classification method. This feature is more limited than the DNS transaction method in its functionality. Customized server specification requires user configuration of the specific domains to identify using the DNS transaction information.

Use of customized server specification overrides other NBAR classification methods for the specified domain, and should only be used when specifically required. For information about this feature, including configuration commands, see: NBAR Custom Applications Based on DNS Name.

NBAR and Classification with Dynamic PDLMs

Dynamic Packet Description Language Modules (PDLMs) allow new protocol support or enhance existing protocol support for NBAR without the requirement of a specific Cisco release upgrade and device reload. If

the support is for enhancing protocols for NBAR, the module version of the PDLMs should be greater than the existing version of the PDLMs. Subsequent Cisco releases incorporate support for these new protocols.

Note PDLMs must be loaded on both Route Processors (RPs) when using the ASR 1006 redundant hardware setup.

Dynamic PDLMs are platform-specific and have a Software Family Identifier (SFI) embedded in them. Dynamic PDLMs of other platforms cannot be loaded on Cisco ASR 1000 Series Aggregation Services Routers.

NBAR-Supported Protocols

The **match protocol**(NBAR) command is used to classify traffic on the basis of protocols supported by NBAR. NBAR is can classify the following types of protocols:

- Non-UDP and non-TCP IP protocols
- TCP and UDP protocols that use statically assigned port numbers
- TCP and UDP protocols that use statically assigned port numbers but still require stateful inspection
- TCP and UDP protocols that dynamically assign port numbers and therefore require stateful inspection

To view the list of protocols supported in a protocol pack, see NBAR Protocol Library.

NBAR2 Protocol Pack

The NBAR2 Protocol Pack provides an easy way to update protocols supported by NBAR2 without replacing the base IOS image that is already present in the device. A Protocol Pack is a set of protocols developed and packaged together. To view the list of protocols supported in a Protocol Pack, see NBAR2 Protocol Library.

NBAR and Classification of Peer-to-Peer File-Sharing Applications

The following applications are the most common peer-to-peer file-sharing applications supported by NBAR:

- BitTorrent
- DirectConnect
- eDonkey
- eMule
- FastTrack
- KazaA (and KazaA Lite and KazaA Lite Resurrection)
- Win MX
- POCO

DirectConnect and eDonkey P2P protocols support the following subclassifications depending on your release:

- eDonkey supports the following subclassification options:

- file-transfer
- search-file-name
- text-chat

- KazaA, FastTrack, and Gnuetella support the file-transfer subclassification.

The Gnutella file sharing became classifiable using NBAR in Cisco IOS XE Release 2.5.

Applications that use the Gnutella protocol are Bearshare, Gnewtellium, Gnucleus, Gtk-Gnutella, Limewire, Mutella, Phex, Qtella, Swapper, and Xolo. The traffic from the applications that use the Gnutella protocol will be classified as Gnutella and not as the respective application.

NBAR Multi stage Classification

NBAR supports a wide range of stateful network protocols such as HTTP classification by URL, Host and MIME type, FTP, TFTP, and so on. NBAR classifies static-port protocols such as those classifiable with access control lists (ACLs).

Multi stage classification reports the underlying protocol as a temporary classification instead of an unknown classification. For example, in earlier releases, to support cases like Video-over-HTTP, where the signature is found on the HTTP response packet, recursive classification over HTTP was allowed causing the first packet of HTTP flows to be reported as unknown, which in turn impacted the following:

- Protocol discovery—reduced classification.
- Packet-based flexible NetFlow (FNF)—reduced classification.
- QoS—delayed classification.
- Performance—because more packets were being processed.
- Aging short flows that are in the middle of a classification process stops without any classification results, although they were partially classified.

Prior to NBAR multi stage classification, NBAR reported an unknown classification result until a final classification decision was reached. NBAR multi stage classification returns the most up-to-date classification decision. It modifies the data path to expose the underlying protocols from media partitioning (MP) recursive classification path—instead of returning “unknown” until a final classification is available, it returns the current (temporary) classification decision.

NBAR multi stage classification has the following characteristics:

Backward incompatibility

If a system has a policy that matches a protocol like SOCKet Secure (SOCKS), which is an underlying protocol for AOL Instant Messenger (AIM) and Bittorrent, when all other protocols have failed (when other protocols are also enabled, either through protocol discovery or through FNF or explicitly through modular QoS CLI [MQC]), this policy would match the first packets of AIM or Bittorrent flows as SOCKS. Blocking the underlying protocol while allowing non underlying protocols is not possible with multi stage classification.

Traffic Reordering

When a user configures different priorities for each classification on the traffic flow, the flow might be directed to different output queues. With multi stage classification more than one classification decision for a single traffic flow may occur. When the traffic is based on prioritized classification, we recommend that the underlying protocols get a higher priority (for example, HTTP get a higher priority than Video-over-HTTP).

Performance Routing (PfR)

When PfR checks the classification from NBAR to make a routing decision, it takes into account if this is a final classification or not. If it is not the final classification, no routing decision is made as it may split the traffic flow to many paths resulting in an “unknown” classification.

NBAR clients let the users know if the classification is temporary or not.

NBAR Scalability

Interface Scalability

Depending on your release there is no limit to the number of interfaces on which protocol discovery can be enabled.

The following table provides details of the protocol discovery supported interface and the release number.

Table 57: Release and Protocol Discovery Interface Support

Release	Number of Interfaces Supported with Protocol Discovery
Cisco IOS XE Release 2.5	128
Cisco IOS XE Release 2.6	256
Cisco IOS XE Release 2.7	256
Cisco IOS XE Release 3.2S and later releases	256

Flow Scalability

The number of bidirectional flows and the platforms supported are same for all releases. A method to reduce the number of active flows based on quick aging is available.

Quick aging occurs under the following conditions:

- TCP flows that do not reach the established state.
- UDP flows with fewer than five packets that are not classified within the specified quick aging timeout.
- Flows that are not classified within the specified quick aging timeout.

The quick aging method reduces the number of flows required for NBAR operation up to three times or more depending on the network behavior.

The Cisco Cloud Services Router 1000V Series devices exhibit the same behavior as that of ESP5 with respect to flow scalability.

Flow Table Sizing

The **ip nbar resources flow max-sessions** command provides the option to override the default maximum flow sessions that are allowed in a flow table. The performance of the device with the NBAR feature depends on the memory size and the number of flows configured for the flow table. The flexibility to change the

number of flows helps in increasing the performance of the system depending on the capacity of the device. To verify the NBAR flow statistics, use the **show ip nbar resources flow** command.

The following table provides the details of the platform and the flow size limits:

Table 58: Platform and Flow Size Details

Platform	Maximum Number of Flows	Default Number of Flows	Memory Upper Limit (70% of Platform Memory)
ESP5/ASR1001/CSR	750,000	500,000	179 MB
ESP10	1,650,000	1,000,000	358 MB
ESP20/ESP40/ASR1002-X	3,500,000	1,000,000	716 MB
ESP100	10,000,000	3,000,000	2.1 GB

To reduce the memory impact, the recommended number of flows is 50,000, where such a configuration is sufficient.

Note The total number of flow entries does not increase when the overall system memory usage is at or above 90%.

NBAR Protocol Discovery

NBAR includes a feature called Protocol Discovery. Protocol discovery provides an easy way to discover protocol packets passing through an interface. For more information about Protocol Discovery, see the "Enabling Protocol Discovery" module.

NBAR Protocol Discovery MIB

The NBAR Protocol Discovery MIB expands the capabilities of NBAR Protocol Discovery by providing the following new functionalities through the Simple Network Management Protocol (SNMP):

- Enable or disable Protocol Discovery per interface.
- Display Protocol Discovery statistics.
- Configure and display multiple top-n tables that list protocols by bandwidth usage.
- Configure thresholds based on the traffic of particular NBAR-supported protocols or applications that report breaches and send notifications when these thresholds are exceeded.

For more information about the NBAR Protocol Discovery MIB, see the "Network-Based Application Recognition Protocol Discovery Management Information Base" module.

NBAR and Multipacket Classification

Depending on your release, NBAR provides the ability to simultaneously search large number of multipacket signatures. This new technique is supported for many of the new protocols. This technique also provides

improved performance and accuracy for other protocols. Along with the support for new signatures, the multipacket classification capabilities change NBAR behavior in the following ways:

1. NBAR classification requires anywhere between 1 and 15 payload packets in a flow depending on the protocol. Retransmitted packets are not counted in this calculation.
2. NBAR will neither classify flows without any payload packets nor classify any TCP payload packet with a wrong sequence number even if there are 15 payload packets for classification.
3. TCP retransmitted packets are not counted as valid packets for classification in the Multipacket Engine module. These type of packets can delay the classification until a sufficient number of valid payload packets are accumulated.
4. Payload packets with only static signatures in NBAR are classified after the single-packet and multipacket protocols are processed and failed. Therefore, a maximum of 15 payload packets can be classified as unknown until the final (static) classification decision is taken.
5. Due to the above-mentioned restrictions, custom protocols can be used to force the classification of the first packet, ignoring the existence of payload or correct sequence numbers in the port-based classification.

NBAR on VRF Interfaces

Depending on your release, the NBAR IPv4 and IPv6 classification on VRF interfaces is supported.

Note Classification for Citrix protocol with “app” subclassification is not guaranteed on VRF interfaces when NBAR is enabled on VRF interfaces.

NBAR and IPv6

Depending on your release, the following types of classification are supported:

- NBAR provides static port-based classification and IP protocol-based classification for IPv6 packets.
- NBAR supports IPv6 classification in protocol discovery mode, but not in MQC mode.
- NBAR always reads the next header field in the fixed IPv6 header to determine the transport layer protocol used by the packet’s payload for IPv6 packets. If an IPv6 packet contains one or more extension headers, NBAR will not skip to the last IPv6 extension header to read the actual protocol type; instead, NBAR classifies the packet as an IPv6 extension header packet.

NBAR Support for IPv6

Depending on your release, NBAR supports the following types of classification:

- Native IPv6 classification.
- Classification of IPv6 traffic flows inside tunneled IPv6 over IPv4 and teredo.
- IPv6 classification in protocol discovery mode and in MQC mode.
- Static and stateful classification.

- Flexible NetFlow with NBAR based fields on IPv6.

NBAR supports IPv6 in IPv4 (6-to-4, 6rd, and ISATAP), and teredo tunneled classification. The **ip nbar classification tunneled-traffic** command is used to enable the tunneled traffic classification. When the tunneled traffic classification is enabled, NBAR performs an application classification of IPv6 packets that are carried inside the IPv4 traffic. If the **ip nbar classification tunneled-traffic** command is disabled, the tunneled IPv6 packets are handled as IPv4 packets.

NBAR supports the capture of IPv6 fields and allows the creation of IPv6 traffic-based flow monitors. When you enable the **ipv6 flow monitor** command, the monitor is bound to the interface, NBAR classification is applied to the IPv6 traffic type, and Flexible NetFlow captures the application IDs in the IPv6 traffic flow.

NBAR Support for GETVPN

NBAR supports Group Encrypted Transport VPN (GETVPN). When ingress QoS is in crypto-map mode, the ingress QoS will work on encrypted traffic.

You can go back to backward compatible mode by using the **ip nbar disable classification encrypted-app** command in global configuration mode.

Note GETVPN is currently not supported by AVC and FNF.

NBAR Support for CAPWAP

CAPWAP (Control And Provisioning of Wireless Access Points) is a protocol is used in wireless traffic, providing point-to-point encapsulation (tunnel) for application traffic. There are two types of CAPWAP traffic: data and control.

NBAR provides a CAPWAP recognition mode that enables NBAR classification of the application traffic within a CAPWAP tunnel.

Classification Behavior: CAPWAP Recognition Disabled/Enabled

By default, CAPWAP recognition mode is not enabled. All CAPWAP traffic is reported as "capwap-data" or "capwap-control" without details about the application traffic within the tunnel.

When CAPWAP recognition is enabled:

- CAPWAP control traffic: NBAR reports as "capwap-control."
- CAPWAP data traffic: NBAR reports on the specific application traffic within the tunnel.

CAPWAP Traffic Type	NBAR CAPWAP Recognition Enabled	NBAR CAPWAP Recognition Disabled
Control traffic	NBAR reports traffic as "capwap-control"	NBAR reports traffic as "capwap-control"
Data traffic	NBAR reports application traffic within the CAPWAP tunnel	NBAR reports traffic as "capwap-data"

Requirements

The following are required for the NBAR recognition of application traffic within a CAPWAP tunnel:

- Cisco IOS XE platform
- Cisco IOS XE 3.17 or later
- NBAR enabled on the platform

Usage

The CAPWAP feature is disabled by default. Use the **ip nbar classification tunneled-traffic capwap** CLI to enable the feature. To disable, use **no ip nbar classification tunneled-traffic capwap**.

```
device# config terminal
device(config)# ip nbar classification tunneled-traffic capwap
```

NBAR Configuration Processes

You can configure NBAR in the following two ways:

- Configuring NBAR using MQC
- Enabling Protocol Discovery

For more information about the NBAR configuration, see the QoS: NBAR Configuration Guide.

Restarting NBAR

NBAR is restarted under the following circumstances.

- Custom protocol addition via CLI
- PDLM load
- RP switchover
- FP switchover
- Protocol pack installation
- Link-age change

Restart involves deactivating and reactivating NBAR. During this time, all packets are classified as 'Unknown' by NBAR. Once NBAR is reactivated, classification is activated.

Note Protocol Discovery statistics will be lost with RP Switchover.

How to Configure DNS-based Categorization

The following procedures describe how to configure NBAR DNS-based Categorization, including enabling/disabling the feature overall, and enabling/disabling DNS Guard.

For background information, see NBAR DNS-based Classification, on page 687.

Enabling and Disabling DNS-based Classification

NBAR2 employs a traffic analysis mechanism called DNS-based classification that learns the network addresses of applications by analyzing DNS query/response traffic. This enables NBAR to classify application traffic from the first packet of a flow, sometimes called "first in flow" (FIF). The mechanism, sometimes called DNS-based learning, applies to applications described by protocols in the NBAR2 Protocol Pack provided by Cisco.

The mechanism is enabled by default. Disabling the feature may be useful if the mechanism causes mis-classification of traffic. Use the **no** form of the command to disable.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **ip nbar classification dns learning**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	[**no**] **ip nbar classification dns learning** **Example:** `Device(config)#no ip nbar classification dns learning`	Enables or disables the DNS-based classification mechanism. This example disables the feature. Default: enabled

Enabling and Disabling DNS Guard for DNS-based Categorization

The DNS-based Categorization mechanism analyzes DNS request/response traffic in order to learn the network addresses of applications. When successful, this enables NBAR to classify the application traffic from the first packet in a flow. In unusual situations, it may cause mis-classification. The feature is disabled by default. See Enabling and Disabling DNS-based Classification, on page 696.

In typical use, it is recommended to apply DNS-based Categorization only when the complete DNS transaction (request, response) is available, in order to prevent mis-classification of traffic. The DNS Guard feature enables this control.

- **Enabled**: DNS-based Categorization operates only when both the DNS request and response are available to analyze.

- **Disabled**: DNS-based Categorization does not require a DNS request, and uses only the DNS response to learn the network address of applications. Use the **no** form of the command to disable.

The mechanism is disabled by default.

SUMMARY STEPS

1. **configure terminal**
2. [**no**] **ip nbar classification dns learning guard**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	[**no**] **ip nbar classification dns learning guard** **Example:** `Device(config)#no ip nbar classification dns learning guard`	Enables or disables DNS Guard. This example disables the feature. Default: disabled

How to Classify Network Traffic Using NBAR

NBAR provides two approaches to configuring attribute-based protocol matching:

- Grouping traffic into **categories and sub-categories** (see Configuring Attribute-based Protocol Match Using Categories and Sub-categories, on page 699)

  Useful for policy implementations that do not use SRND. A disadvantage of this method is that it can be difficult to keep track of the mapping between traffic and the categories and sub-categories defined within the policy.

- Using the Solution Reference Network Designs (**SRND**) model (see Configuring Attribute-based Protocol Match Using SRND, on page 701)

  Simplifies the configuration of SRND-based policies. Although the category/sub-category model can support SRND implementations, it is simpler and more efficient to use this model.

About Configuring Attribute-based Protocol Matching Using Categories

Useful for policy implementations that do not use SRND. A disadvantage of this method is that it can be difficult to keep track of the mapping between traffic and the categories and sub-categories defined within the policy. For information about the procedure, see Configuring Attribute-based Protocol Match Using Categories and Sub-categories, on page 699.

About Configuring Attribute-based Protocol Matching Using SRND

The NBAR category/sub-category model can support SRND implementations. However, beginning with the release of IOS 15.5(3)T and IOS XE 3.16S, for SRND policy implementations it is more efficient and recommended to use the SRND-specific model instead.

The SRND-specific model provides two attributes (**traffic-class** and **business-relevance**) to configure protocol matching for SRND-based policies. The attributes provided for operation with SRND-based policies are applicable only within the context of SRND implementations.

Background: SRND Policy Model

The Solution Reference Network Designs (SRND) policy model simplifies prioritization of traffic for QoS. It provides 12 classes that define traffic according to application. Each class of traffic can be directed to a specific QoS queue. Of these classes:

- 10 classes apply to business-relevant applications operating in 10 different recognized technologies, such as VoIP, video, conferencing, and so on.
- 1 class applies to business-relevant applications of unknown technology.
- 1 class applies to business-irrelevant applications.

Flexibility to Reclassify Applications

The 12 classes that NBAR provides for operating with the SRND model include default values appropriate for most enterprises. However, NBAR makes it easy to reclassify specific applications as business-relevant or business-irrelevant, as necessary. (See example of reclassifying the Skype VoIP application: Example: SRND Configuration - Reclassifying an Application as Business-relevant, on page 708)

Attribute: traffic-class

The **traffic-class** attribute specifies the general category of the traffic, such as VoIP, video, conferencing, and so on. The The following table describes the 10 values for **traffic-class**.

Table 59: Values for traffic-class

Value	Description
voip-telephony	VoIP telephony (bearer-only) traffic
broadcast-video	Broadcast TV, live events, video surveillance
real-time-interactive	High-definition interactive video applications
multimedia-conferencing	Desktop software multimedia collaboration applications
multimedia-streaming	Video-on-Demand (VoD) streaming video
network-control	Network control plane traffic
signaling	Signaling traffic that supports IP voice and video telephony
ops-admin-mgmt	Network operations, administration, and management traffic

Value	Description
transactional-data	Interactive data applications
bulk-data	Non-interactive data applications

Attribute: business-relevance

The business-relevance attribute specifies whether the application is considered relevant to the business activity of the organization. The default values reflect typical usage and business relevance, but the values can be customized according to the specific requirements of an organization.

The following table describes the values for business-relevance.

Table 60: Values for business-relevance

Value	Description
business-relevant	Application critical for an organization's business activity
default	Application used for an organization's business activity
business-irrelevant	Application not relevant to an organization's business activity

Configuring Attribute-based Protocol Match Using Categories and Sub-categories

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** [**type**] [**match-all** | **match-any**] *class-map-name*
4. **match protocol attribute application-group** *application-group* [*application-name*]
5. **match protocol attribute category** *application-category* [*application-name*]
6. **match protocol attribute encrypted** {**encrypted-no** | **encrypted-unassigned** | **encrypted-yes**} [*application-name*]
7. **match protocol attribute sub-category** *application-category* [*application-name*]
8. **match protocol attribute tunnel** {**tunnel-no** | **tunnel-unassigned** | **tunnel-yes**} [*application-name*]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map** [**type**] [**match-all** | **match-any**] *class-map-name* **Example:** `Device(config)# class-map cmap1`	Creates a class map to be used for matching packets to a specified class and enters QoS class-map mode. • Enter the name of the class map.
Step 4	**match protocol attribute application-group** *application-group* [*application-name*] **Example:** `Device(config-cmap)# match protocol attribute application-group skype`	Configures the specified application group as the match criterion. • (Optional) Use the *application-name* argument to configure the application and not the application group as the match criterion. The configuration is saved as **match protocol** *application-name* instead of **match protocol attribute application-group** *application-group*.
Step 5	**match protocol attribute category** *application-category* [*application-name*] **Example:** `Device(config-cmap)# match protocol attribute category email`	Configures the specified category as the match criteria attribute. • (Optional) Use the *application-name* argument to configure a specific application, and not the application category, as the match criterion. The configuration is saved as **match protocol** *application-name* instead of **match protocol attribute category** *application-category*.
Step 6	**match protocol attribute encrypted** {**encrypted-no** | **encrypted-unassigned** | **encrypted-yes**} [*application-name*] **Example:** `Device(config-cmap)# match protocol attribute encrypted encrypted-yes`	Configures the specified encryption status as the match criterion. • (Optional) Use the *application-name* argument to configure application within the specified encrypted status as the match criterion. The configuration is saved as **match protocol** *application-name* instead of **match protocol attribute encrypted** {**encrypted-no** | **encrypted-unassigned** | **encrypted-yes**}.
Step 7	**match protocol attribute sub-category** *application-category* [*application-name*] **Example:** `Device(config-cmap)# match protocol attribute sub-category client-server`	Configures the specified sub-category as the match criteria attribute. • (Optional) Use the *application-name* argument to configure a specific application, and not the sub-category, as the match criterion. The configuration is saved as **match protocol** *application-name* instead of **match protocol attribute sub-category** *application-category*.

	Command or Action	Purpose
Step 8	**match protocol attribute tunnel** {**tunnel-no** | **tunnel-unassigned** | **tunnel-yes**} [*application-name*] **Example:** `Device(config-cmap)# match protocol attribute tunnel tunnel-yes`	Configures the specified encryption status as the match criterion. • (Optional) Use the *application-name* argument to configure a specific application within the specified tunneling status as the match criterion. The configuration is saved as **match protocol** *application-name* instead of **match protocol attribute tunnel** {**tunnel-no** | **tunnel-unassigned** | **tunnel-yes**}.
Step 9	**end** **Example:** `Device(config-cmap)# end`	Exits Qos class-map mode and returns to privileged EXEC mode.

Configuring Attribute-based Protocol Match Using SRND

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** [**type**] [**match-all** | **match-any**] *class-map-name*
4. **match protocol attribute traffic-class** *traffic-class-option*
5. **match protocol attribute business-relevance** *business-relevance-option*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map** [**type**] [**match-all** | **match-any**] *class-map-name* **Example:** `Device(config)# class-map cmap1`	Creates a class map to be used for matching packets to a specified class and enters QoS class-map mode. • Enter the name of the class map.
Step 4	**match protocol attribute traffic-class** *traffic-class-option* **Example:** `Device(config-cmap)# match protocol attribute traffic-class voip-telephony`	Configures the specified traffic class as the match criterion. • *traff-class-option* possible values: voip-telephony, broadcast-video, real-time-interactive,

	Command or Action	Purpose
		multimedia-conferencing, multimedia-streaming, network-control, signaling, ops-admin- mgmt, transactional-data, bulk-data
Step 5	**match protocol attribute business-relevance** *business-relevance-option* **Example:** `Device(config-cmap)# match protocol attribute business-relevance business-relevant`	Configures the specified category as the match criteria attribute. • *business-relevance-option* possible values: business-relevant, default, business-irrelevant
Step 6	**end** **Example:** `Device(config-cmap)# end`	Exits QoS class-map mode and returns to privileged EXEC mode.

SRND Configuration: Typical Class-Map, Policy-Map

The following sections show a typical example of a class-map and policy-map for an SRND implementation. It illustrates how the **traffic-class** and **business-relevance** attributes address the 12-class SRND QoS model.

Class-map

```
class-map match-all VOICE
    match protocol attribute traffic-class voip-telephony
    match protocol attribute business-relevance business-relevant

class-map match-all BROADCAST-VIDEO
    match protocol attribute traffic-class broadcast-video
    match protocol attribute business-relevance business-relevant

class-map match-all INTERACTIVE-VIDEO
    match protocol attribute traffic-class real-time-interactive
    match protocol attribute business-relevance business-relevant

class-map match-all MULTIMEDIA-CONFERENCING
    match protocol attribute traffic-class multimedia-conferencing
    match protocol attribute business-relevance business-relevant

class-map match-all MULTIMEDIA-STREAMING
    match protocol attribute traffic-class multimedia-streaming
    match protocol attribute business-relevance business-relevant

class-map match-all SIGNALING
    match protocol attribute traffic-class signaling
    match protocol attribute business-relevance business-relevant

class-map match-all NETWORK-CONTROL
    match protocol attribute traffic-class network-control
     match protocol attribute business-relevance business-relevant

class-map match-all NETWORK-MANAGEMENT
    match protocol attribute traffic-class ops-admin-mgmt
    match protocol attribute business-relevance business-relevant

class-map match-all TRANSACTIONAL-DATA
```

```
    match protocol attribute traffic-class transactional-data
    match protocol attribute business-relevance business-relevant

class-map match-all BULK-DATA
    match protocol attribute traffic-class bulk-data
    match protocol attribute business-relevance business-relevant

class-map match-all SCAVENGER
    match protocol attribute business-relevance business-irrelevant
```

Policy-map

```
policy-map 12-cls-marking

class VOICE
    set dscp ef

class BROADCAST-VIDEO
    set dscp cs5

class INTERACTIVE-VIDEO
    set dscp cs4

class MULTIMEDIA-CONFERENCING
    set dscp af41

class MULTIMEDIA-STREAMING
    set dscp af31

class SIGNALING
    set dscp cs3

class NETWORK-CONTROL
    set dscp cs6

class NETWORK-MANAGEMENT
    set dscp cs2

class TRANSACTIONAL-DATA
    set dscp af21

class BULK-DATA
    set dscp af11

class SCAVENGER
    set dscp cs1

class class-default
    set dscp default
```

Configuration Examples for Classifying Network Traffic Using NBAR in Cisco Software

Example: Classification of HTTP Traffic Using the HTTP Header Fields

In the following example, any request message that contains “somebody@cisco.com” in the user-agent, referer, or from field will be classified by NBAR. Typically, a term with a format similar to "somebody@cisco.com" would be found in the From header field of the HTTP request message.

```
Device(config)# class-map match-all class1
Device(config-cmap)# match protocol http from “somebody@cisco.com”
```

In the following example, any request message that contains "http://www.cisco.com/routers" in the User-Agent, Referer, or From field will be classified by NBAR. Typically, a term with a format similar to "http://www.cisco.com/routers" would be found in the Referer header field of the HTTP request message.

```
Device(config)# class-map match-all class2
Device(config-cmap)# match protocol http referer “http://www.cisco.com/routers”
```

In the following example, any request message that contains "CERN-LineMode/2.15" in the User-Agent, Referer, or From header field will be classified by NBAR. Typically, a term with a format similar to "CERN-LineMode/2.15" would be found in the User-Agent header field of the HTTP request message.

```
Device(config)# class-map match-all class3
Device(config-cmap)# match protocol http user-agent “CERN-LineMode/2.15”
```

In the following example, any response message that contains "CERN/3.0" in the Content-Base (if available), Content-Encoding, Location, or Server header field will be classified by NBAR. Typically, a term with a format similar to "CERN/3.0" would be found in the Server header field of the response message.

```
Device(config)# class-map match-all class4
Device(config-cmap)# match protocol http server “CERN/3.0”
```

In the following example, any response message that contains “http://www.cisco.com/routers” in the Content-Base (if available), Content-Encoding, Location, or Server header field will be classified by NBAR. Typically, a term with a format similar to “http://www.cisco.com/routers” would be found in the Content-Base (if available) or Location header field of the response message.

```
Device(config)# class-map match-all class5
Device(config-cmap)# match protocol http location “http://www.cisco.com/routers”
```

In the following example, any response message that contains “gzip” in the Content-Base (if available), Content-Encoding, Location, or Server header field will be classified by NBAR. Typically, the term “gzip” would be found in the Content-Encoding header field of the response message.

```
Device(config)# class-map match-all class6
Device(config-cmap)# match protocol http content-encoding “gzip”
```

Example: Combinations of Classification of HTTP Headers and URL Host or MIME Type to Identify HTTP Traffic

In the following example, HTTP header fields are combined with a URL to classify traffic. In this example, traffic with a User-Agent field of "CERN-LineMode/3.0" and a Server field of "CERN/3.0", along with host name "cisco.com" and URL "/routers", are classified using NBAR:

```
Device(config)# class-map match-all c-http
Device(config-cmap)# match protocol http user-agent "CERN-LineMode/3.0"
Device(config-cmap)# match protocol http server "CERN/3.0"
Device(config-cmap)# match protocol http host cisco*
Device(config-cmap)# match protocol http url /routers
```

Example: NBAR and Classification of Custom Protocols and Applications

In the following example, the custom protocol LAYER4CUSTOM will look for TCP packets that have a destination or source port of 6700:

```
Device# configure terminal
Device(config)# ip nbar custom LAYER4CUSTOM  transport tcp id 14
Device(config-custom)# port 6700
```

To display other options besides port:

```
Device# configure terminal
Device(config)# ip nbar custom LAYER4CUSTOM  transport tcp id 14
Device(config-custom)# ?
Custom protocol commands:
  direction  Flow direction
  dscp       DSCP in IPv4 and IPv6 packets
  exit       Exit from custom configuration mode
  ip         ip address
  ipv6       ipv6 address
  no         Negate a command or set its defaults
  port       ports
```

Example: NBAR and Classification of Peer-to-Peer File-Sharing Applications

The **match protocol gnutella file-transfer** *regular-expression* and **match protocol fasttrack file-transfer** *regular-expression* commands are used to enable Gnutella and FastTrack classification in a traffic class. The **file-transfer** keyword indicates that a regular expression variable will be used to identify specific Gnutella or FastTrack traffic. The *regular-expression* variable can be expressed as "*" to indicate that all FastTrack or Gnutella traffic be classified by a traffic class.

In the following example, all FastTrack traffic is classified into class map nbar:

```
Device(config)# class-map match-all nbar
Device(config-cmap)# match protocol fasttrack file-transfer "*"
```

Similarly, all Gnutella traffic is classified into class map nbar in the following example:

```
Device(config)# class-map match-all nbar
Device(config-cmap)# match protocol gnutella file-transfer "*"
```

Wildcard characters in a regular expression can also be used to identify specified Gnutella and FastTrack traffic. These regular expression matches can be used to match on the basis of a filename extension or a particular string in a filename.

In the following example, all Gnutella files that have the .mpeg extension are classified into class map nbar:

```
Device(config)# class-map match-all nbar
Device(config-cmap)# match protocol gnutella file-transfer "*.mpeg"
```

In the following example, only Gnutella traffic that contains the characters "cisco" is classified:

```
Device(config)# class-map match-all nbar
Device(config-cmap)# match protocol gnutella file-transfer "*cisco*"
```

The same examples can be used for FastTrack traffic:

```
Device(config)# class-map match-all nbar
Device(config-cmap)# match protocol fasttrack file-transfer "*.mpeg"
```

or

```
Device(config)# class-map match-all nbar
Device(config-cmap)# match protocol fasttrack file-transfer "*cisco*"
```

Example: Configuring Attribute-Based Protocol Match

The **match protocol attributes** command is used to configure different attributes as the match criteria for application recognition.

In the following example, the email-related applications category is configured as the match criterion:

```
Device# configure terminal
Device(config)# class-map mygroup
Device(config-cmap)# match protocol attribute category email
```

In the following example, skype-group applications are configured as the match criterion:

```
Device# configure terminal
Device(config)# class-map apps
Device(config-cmap)# match protocol attribute application-group skype-group
```

In the following example, encrypted applications are configured as the match criterion:

```
Device# configure terminal
Device(config)# class-map my-class
Device(config-cmap)# match protocol encrypted encrypted-yes
```

In the following example, Client-server subcategory applications are configured as the match criterion:

```
Device# configure terminal
Device(config)# class-map newmap
Device(config-cmap)# match protocol attribute sub-category client-server
```

In the following example, tunneled applications are configured as the match criterion:

```
Device# configure terminal
Device(config)# class-map mygroup
Device(config-cmap)# match protocol attribute tunnel tunnel-yes
```

The following sample output from the **show ip nbar attribute** command displays the details of all the attributes:

```
Device# show ip nbar attribute

      Name :  category
      Help :  category attribute
      Type :  group
    Groups :  email, newsgroup, location-based-services, instant-messaging, netg
      Need :  Mandatory
   Default :  other

      Name :  sub-category
      Help :  sub-category attribute
      Type :  group
    Groups :  routing-protocol, terminal, epayment, remote-access-terminal, nen
      Need :  Mandatory
   Default :  other

      Name :  application-group
      Help :  application-group attribute
      Type :  group
    Groups :  skype-group, wap-group, pop3-group, kerberos-group, tftp-group, bp
      Need :  Mandatory
   Default :  other

      Name :  tunnel
      Help :  Tunnelled applications
      Type :  group
    Groups :  tunnel-no, tunnel-yes, tunnel-unassigned
      Need :  Mandatory
   Default :  tunnel-unassigned

      Name :  encrypted
      Help :  Encrypted applications
      Type :  group
    Groups :  encrypted-yes, encrypted-no, encrypted-unassigned
      Need :  Mandatory
   Default :  encrypted-unassigned
```

The following sample output from the **show ip nbar protocol-attribute** command displays the details of the protocols:

```
Device# show ip nbar protocol-attribute

           Protocol Name :  ftp
                category :  file-sharing
            sub-category :  client-server
       application-group :  ftp-group
                  tunnel :  tunnel-no
               encrypted :  encrypted-no

           Protocol Name :  http
                category :  browsing
            sub-category :  other
       application-group :  other
                  tunnel :  tunnel-no
               encrypted :  encrypted-no

           Protocol Name :  egp
                category :  net-admin
```

```
     sub-category :  routing-protocol
application-group :  other
           tunnel :  tunnel-no
        encrypted :  encrypted-no

    Protocol Name :  gre
         category :  net-admin
     sub-category :  tunneling-protocols
application-group :  other
           tunnel :  tunnel-yes
        encrypted :  encrypted-no
```

Example: SRND Configuration - Reclassifying an Application as Business-relevant

Skype is a consumer VoIP product typically not used in business. In SRND-specific protocol mapping, Skype is classified as business-irrelevant by default. However, some organizations may use Skype as a business-critical application. This examples shows how to reclassify Skype as business-relevant.

1. Show the current protocol attributes for Skype. The results indicate (in the last two lines) that Skype is classified as a voip-telephony technology, and is business-irrelevant.

   ```
   show ip nbar protocol-attribute skype
   encrypted           encrypted-yes
   tunnel              tunnel-no
   category            voice-and-video
   sub-category        consumer-multimedia-messaging
   application-group   skype-group
   p2p-technology      p2p-tech-yes
   traffic-class       voip-telephony
   business-relevance  business-irrelevant
   ```

   At this stage, Skype will be matched by the SCAVENGER class-map, which is part of the standard default SRND class-map configuration.

   ```
   class-map match-all SCAVENGER
       match protocol attribute business-relevance business-irrelevant
   ```

2. Change the value of business-relevance for Skype to business-relevant.

   ```
   ip nbar attribute-map demo
       attribute business-relevance business-relevant
   ip nbar attribute-set skype demo
   ```

   At this stage, Skype will be matched by the VOIP-TELEPHONY class-map, which is part of the standard default SRND class-map configuration.

   ```
   class-map match-all VOIP-TELEPHONY
       match protocol attribute traffic-class voip-telephony
       match protocol attribute business-relevance business-relevant
   ```

3. Confirm that Skype is now classified as business-relevant. The new value appears on the last line of the following results.

   ```
   show ip nbar protocol-attribute skype
   encrypted           encrypted-yes
   tunnel              tunnel-no
   ```

```
category            voice-and-video
sub-category        consumer-multimedia-messaging
application-group   skype-group
p2p-technology      p2p-tech-yes
traffic-class       voip-telephony
business-relevance  business-relevant
```

Example: Customizing a Built-in Protocol

Customizing an NBAR built-in protocol (provided by the Cisco Protocol Pack) to include an additional user-specified domain extends the scope of the built-in protocol. Any policy associated with the protocol will then apply to the user-specified domain also. The following example configures a customization called myOffice365, which extends the built-in office365 protocol to include domains that match to "*uniqueOffice365".

In the following example, the email-related applications category is configured as the match criterion:

```
Device# configure terminal
Device(config)# ip nbar custom myOffice365 dns domain-name "*uniqueOffice365" extends
office365
```

Additional References

The following sections provide references related to enabling Protocol Discovery.

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Concepts and information about NBAR	"Classifying Network Traffic Using NBAR" module
Configuring NBAR using the MQC	"Configuring NBAR Using the MQC" module
Adding application recognition modules (also known as PDLMs)	"Adding Application Recognition Modules" module
Creating a custom protocol	"Creating a Custom Protocol" module

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Classifying Network Traffic Using NBAR

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 61: Feature Information for Classifying Network Traffic Using NBAR

Feature Name	Releases	Feature Information
Additional PDL Support for NBAR	Cisco IOS XE Release 3.1S	The additional PDL Support for NBAR feature provides support for additional PDLs. The following section provides information about this feature: NBAR and Classification of HTTP Traffic
Enable NBAR URI Extraction for HTTP Transactions for Persistent Connections	Cisco IOS XE Release 3.9S	The Enable NBAR URI Extraction for HTTP Transactions for Persistent Connections feature supports extraction and export of URL field per transaction. The following section provides information about this feature: Classification of HTTP Traffic by a URL Host or MIME.
Enhanced NBAR	Cisco IOS XE Release 3.2S	The Enhanced NBAR feature provides additional PDLs for Cisco IOS XE Release 3.2S. The following section provides information about this feature: NBAR-Supported Protocols
NBAR Classification Enhancements for IOS-XE3.5	Cisco IOS XE Release 3.5S	The NBAR Classification Enhancements feature provides additional classification support for native IPv6 classification and classification of flows inside tunneled IPv6 over IPv4. The following section provides information about this feature: NBAR Support for IPv6 The following commands were introduced or modified: **ip nbar classification tunneled-traffic**, **option** (FNF).

Feature Name	Releases	Feature Information
NBAR PDLM Supported in ASR 1000 Release 2.5	Cisco IOS XE Release 2.5 Cisco IOS XE Release 3.1S Cisco IOS XE Release 3.3S	This feature was integrated into Cisco IOS XE Release 2.5. NBAR-supported protocols were added for this release. The following section provides information about this feature: NBAR-Supported Protocols The following command was modified: **match protocol** (NBAR).
NBAR Protocols	Cisco IOS XE Release 2.3	This feature was integrated into Cisco IOS XE Release 2.3. NBAR-supported protocols were added for this release. The following section provides information about this feature: NBAR-Supported Protocols The following command was modified: **match protocol**(NBAR).
NBAR Real-time Transport Protocol Payload Classification	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers. The following section provides information about this feature: NBAR-Supported Protocols
NBAR Static IPv4 IANA Protocols Pack1	Cisco IOS XE Release 3.1S	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers. The following section provides information about this feature: NBAR-Supported Protocols
NBAR VRF-Aware	Cisco IOS XE Release 3.3S	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers. The following section provides information about this feature: NBAR Scalability
NBAR Multi stage Classification	Cisco IOS XE Release 3.7S	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers. The following section provides information about this feature: NBAR Multi-stage Classification.
NBAR2: Add/Rename Static Attributes	Cisco IOS XE Release 3.11S	The custom values enable you to name the attributes based on grouping of protocols. You can create custom values for the attributes application-group, category, and sub-category. The following section provides information about this feature: NBAR Categorization and Attributes. The following commands were introduced or modified: **ip nbar attribute**, **show ip nbar attribute-custom**, and **show ip nbar category**.

Feature Name	Releases	Feature Information
NBAR2 GETVPN (Cryptomap) Support	Cisco IOS XE Release 3.11S	This feature was introduced on Cisco ASR 1000 Series Aggregation Services Routers. The following section provides information about this feature: NBAR Support for GETVPN, on page 694
NBAR Support for CAPWAP	Cisco IOS XE Release 3.17S	CAPWAP (Control And Provisioning of Wireless Access Points) is a protocol is used in wireless traffic, providing point-to-point encapsulation (tunnel) for application traffic. NBAR provides a CAPWAP recognition mode that enables NBAR classification of the application traffic within a CAPWAP tunnel. The following section provides information about this feature: NBAR Support for CAPWAP
NBAR DNS-based Classification	Cisco IOS XE Release 3.17S	This feature can improve traffic classification by using DNS transaction information exchanged when a user initiates a connection with an application server. This method offers the significant advantage of classifying flows from the first packet in the flow. The following section provides information about this feature: NBAR DNS-based Classification
Customizing Built-in Protocols	Cisco IOS XE Denali 16.3	Customizing an NBAR built-in protocol (provided by the Cisco Protocol Pack) to include an additional user-specified domain extends the scope of the built-in protocol. Any policy associated with the protocol will then apply to the user-specified domain also. The following section provides information about this feature: Customizing Built-in Protocols

Glossary

Encryption—Encryption is the application of a specific algorithm to data so as to alter the appearance of the data, making it incomprehensible to those who are not authorized to see the information.

HTTP—Hypertext Transfer Protocol. The protocol used by web browsers and web servers to transfer files, such as text and graphic files.

IANA—Internet Assigned Numbers Authority. An organization operated under the auspices of the Internet Society (ISOC) as a part of the Internet Architecture Board (IAB). IANA delegates authority for IP address-space allocation and domain-name assignment to the InterNIC and other organizations. IANA also maintains a database of assigned protocol identifiers used in the TCP/IP stack, including autonomous system numbers.

LAN—Local-area network. A high-speed, low-error data network that covers a relatively small geographic area (up to a few thousand meters). LANs connect workstations, peripherals, terminals, and other devices in a single building or other geographically limited area. LAN standards specify cabling and signaling at the

physical and data link layers of the Open System Interconnection (OSI) model. Ethernet, FDDI, and Token Ring are widely used LAN technologies.

MIME—Multipurpose Internet Mail Extension. The standard for transmitting nontext data (or data that cannot be represented in plain ASCII code) in Internet mail, such as binary, foreign language text (such as Russian or Chinese), audio, and video data. MIME is defined in RFC 2045, *Multipurpose Internet Mail Extension (MIME) Part One: Format of Internet Message Bodies* .

MPLS—Multiprotocol Label Switching. A switching method that forwards IP traffic using a label. This label instructs the routers and the switches in the network where to forward the packets based on preestablished IP routing information.

MQC—Modular quality of service command-line interface. A CLI that allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach the policy maps to interfaces. Policy maps are used to apply the appropriate quality of service (QoS) to network traffic.

Protocol Discovery—A feature included with NBAR. Protocol Discovery provides a way to discover the application protocols that are operating on an interface.

QoS—Quality of service. A measure of performance for a transmission system that reflects its transmission quality and service availability.

RTCP—RTP Control Protocol. A protocol that monitors the QoS of an IPv6 real-time transport protocol (RTP) connection and conveys information about the ongoing session.

Stateful protocol—A protocol that uses TCP and UDP port numbers that are determined at connection time.

Static protocol—A protocol that uses well-defined (predetermined) TCP and UDP ports for communication.

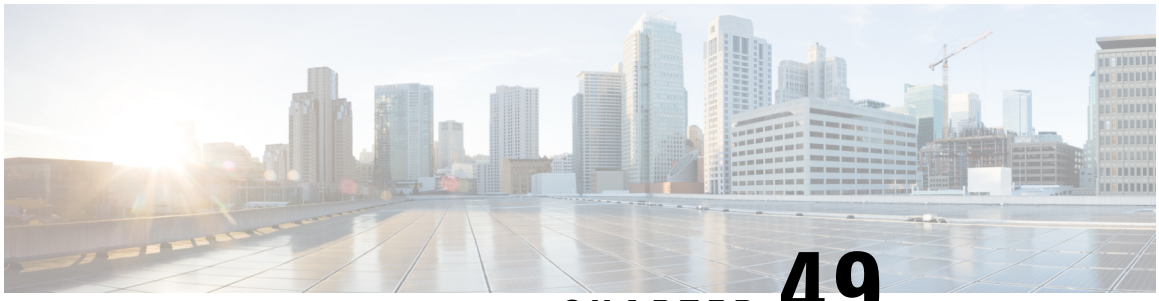
Subport classification—The classification of network traffic by information that is contained in the packet payload, that is, information found beyond the TCP or UDP port number.

TCP—Transmission Control Protocol. A connection-oriented transport layer protocol that provides reliable full-duplex data transmission. TCP is part of the TCP/IP protocol stack.

Tunneling—Tunneling is an architecture that is designed to provide the services necessary to implement any standard point-to-point encapsulation scheme.

UDP—User Datagram Protocol. A connectionless transport layer protocol in the TCP /IP protocol stack. UDP is a simple protocol that exchanges datagrams without acknowledgments or guaranteed delivery, requiring that error processing and retransmission be handled by other protocols. UDP is defined in RFC 768, *User Datagram Protocol* .

WAN—Wide-area network. A data communications network that serves users across a broad geographic area and often uses transmission devices provided by common carriers.



CHAPTER 49

NBAR2 Protocol Pack

The NBAR2 Protocol Pack provides an easy way to update protocols supported by NBAR2 without replacing the base IOS image that is already present in the device. A Protocol Pack is a set of protocols developed and packaged together. To view the list of protocols supported in a Protocol Pack, see NBAR2 Protocol Library.

Prerequisites for the NBAR2 Protocol Pack

The Protocol Pack must be copied to your local disk to avoid any errors after rebooting.

Note It is strongly recommended to load the NBAR2 Protocol Pack that is the exact match for the NBAR2 engine, and also load the latest rebuild of Cisco software. See the NBAR2 Protocol Library page for compatibility information.

Information About the NBAR Protocol Pack

Protocol Pack Overview

NBAR2 Protocol Packs are software packages that update the protocol support on a device without replacing the Cisco software on the device. A Protocol Pack contains a set of signatures supported by NBAR2.

Protocol Packs are sets of protocols developed and packaged together. Each Cisco IOS image comes with a built-in Protocol Pack. With a standard license, a subset of protocols and Protocol Pack features are supported. With an advanced license, all protocols and features are supported. Updating the Protocol Pack on a Cisco IOS release requires an advanced license. For information about licensing, see AVC Licensing and Feature Activation.

To view the list of protocols supported in a Protocol Pack, see NBAR2 Protocol Library.

The NBAR2 taxonomy file contains the information such as common name, description, underlying protocol, for every protocol that is available in the Protocol Pack. Use the **show ip nbar protocol-pack active taxonomy**, **show ip nbar protocol-pack inactive taxonomy**, and **show ip nbar protocol-pack loaded taxonomy** commands to view the taxonomy file for an active, inactive, and all loaded Protocol Packs respectively.

The NBAR2 taxonomy file generally contains the information for more than 1000 protocols, and the taxonomy file size is ~2 MB. It is recommended to redirect the output from the **show ip nbar protocol-pack** [**active** | **inactive** | **loaded**] taxonomy command to a file by using the redirect output modifier, for example, **show ip nbar protocol-pack active taxonomy** | **redirect harddisk:***nbar_taxonomy.xml*.

Protocols Available with Standard License

The default Protocol Pack available with a standard license includes the protocols shown below. For information about the Protocol Packs available with an advanced license, see the NBAR2 Protocol Library.

bgp
bittorrent
cifs
citrix
cuseeme
dhcp
dht
directconnect
dns
edonkey
egp
eigrp
exchange
fasttrack
finger
ftp
gnutella
gopher
gre
http
http-local-net
https
icmp
imap
ipinip
ipsec
ipv6-icmp
irc
kazaa2
kerberos
l2tp
ldap
mgcp

ms-rpc
netbios
nfs
nntp
notes
novadigm
ntp
ospf
pop3
pptp
printer
rip
rsvp
rtcp
rtp
rtsp
secure-ftp
secure-http
secure-imap
secure-irc
secure-ldap
secure-nntp
secure-pop3
secure-telnet
sip
skinny
skype
smtp
snmp
socks
sqlnet
sqlserver
ssh
ssl
stun-nat
sunrpc
syslog
telepresence-control
telnet
teredo-ipv6-tunneled
tftp
winmx
xmpp-client
xwindows

SSL Unique-name Sub-classification

The "unique-name" sub-classification parameter can be used to match SSL sessions of servers that are not known globally, or are not yet supported by NBAR2. The unique-name will match the server name indication (SNI) field in the client request if the SNI field exists, or it will match the common name (CN) field in the first certificate of the server's response.

Note The SSL sub-classification parameters have priority over the built in signatures. Therefore, when a unique-name defined by a user matches a known application such as Facebook, it will not match the built-in protocol but will match SSL with the configured sub-classification.

Note Similar to the other sub-classification features, the classification result (for example, as seen in protocol-discovery), does not change and will remain as SSL. However, the flows matching the class maps will receive the services such as QoS and Performance monitor configured for them. To view the detailed matching statistics, refer to the policy map counters.

For more information on SSL, see http://tools.ietf.org/html/rfc6101.

RTP Dynamic Payload Type Sub-classification

The sub-classification parameters for Real-time Transport Protocol (RTP) audio and RTP video detect RTP flows that use dynamic payload types (PT). Dynamic PTs are PTs in the dynamic range from 96 to 127, as defined in the RTP RFC, and are used by protocols such as SIP and RTSP.

Note The RTP audio/video sub-classification parameters are generic in nature and will match only on generic RTP traffic. More specific classification such as ms-lync-audio, cisco-jabber-audio, facetime, and cisco-phone will not match as RTP, and therefore will not match the audio/video sub-classification.

How to Load the NBAR Protocol Pack

Loading the NBAR2 Protocol Pack

Before you begin

Loading a new Protocol Pack requires an advanced license.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar protocol-pack** *protocol-pack* [**force**]

4. **exit**
5. **show ip nbar protocol-pack** {protocol-pack | **active**} [**detail**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar protocol-pack** *protocol-pack* [**force**] **Example:** `Device(config)# ip nbar protocol-pack harddisk:defProtoPack`	Loads the protocol pack. • Use the **force** keyword to specify and load a Protocol Pack of a lower version, which is different from the base protocol pack version. Doing so also removes any configurations that are not supported by the lower version Protocol Pack.
Step 4	**exit** **Example:** `Device(config)# exit`	Returns to privileged EXEC mode.
Step 5	**show ip nbar protocol-pack** {protocol-pack | **active**} [**detail**] **Example:** `Device(config)# show ip nbar protocol-pack active`	Displays the protocol pack information. • Verify the loaded protocol pack version, publisher, and other details using this command. • Use the *protocol-pack* argument to display information about the specified protocol pack. • Use the **active** keyword to display active protocol pack information. • Use the **detail** keyword to display detailed protocol pack information.

Configuration Examples for the NBAR2 Protocol Pack

Example: Loading the NBAR2 Protocol Pack

The following example shows how to load an NBAR2 Protocol Pack named defProtoPack from the harddisk:

```
Device> enable
Device# configure terminal
Device(config)# ip nbar protocol-pack harddisk:defProtoPack
Device(config)# exit
```

The following example shows how to revert to the base image version of NBAR2 Protocol Pack:

```
Device> enable
Device# configure terminal
Device(config)# default ip nbar protocol-pack
Device(config)# exit
```

The following example shows how to load a Protocol Pack of a lower version using the **force** keyword:

```
Device> enable
Device# configure terminal
Device(config)# ip nbar protocol-pack harddisk:olddefProtoPack force
Device(config)# exit
```

Example: Verifying the Loaded NBAR2 Protocol Pack

The following sample output from the **show ip nbar protocol-pack active** command shows information about the Protocol Pack that is provided by default with a licensed Cisco image on a device:

```
Device# show ip nbar protocol-pack active

ACTIVE protocol pack:
Name:                            Advanced Protocol Pack
Version:                         1.0
Publisher:                       Cisco Systems Inc.
NBAR Engine Version:             14
```

The following sample output from the **show ip nbar protocol-pack active detail** command shows detailed information about the active Protocol Pack that is provided by default with a licensed Cisco image on a device:

```
Device# show ip nbar protocol-pack active detail

ACTIVE protocol pack:
Name:                            Advanced Protocol Pack
Version:                         1.0
Publisher:                       Cisco Systems Inc.
NBAR Engine Version:             14
Protocols:
base                             Mv: 4
ftp                              Mv: 5
http                             Mv: 18
static                           Mv: 6
socks                            Mv: 2
nntp                             Mv: 2
tftp                             Mv: 2
exchange                         Mv: 3
vdolive                          Mv: 1
sqlnet                           Mv: 2
netshow                          Mv: 3
sunrpc                           Mv: 3
```

```
streamwork                      Mv: 2
citrix                          Mv: 11
fasttrack                       Mv: 3
gnutella                        Mv: 7
kazaa2                          Mv: 11
```

The following sample output from the **show ip nbar protocol-pack** command shows the protocol pack information of an advanced Protocol Pack that is present in the specified device location:

```
Device# show ip nbar protocol-pack disk:0ppsmall_higherversion

Name:                           Advanced Protocol Pack
Version:                        2.0
Publisher:                      Cisco Systems Inc.
NBAR Engine Version:            14
Creation time:                  Mon Jul 16 09:29:34 UTC 2012
```

The following sample output from the **show ip nbar protocol-pack** command shows detailed protocol pack information present in the specified disk location:

```
Device# show ip nbar protocol-pack disk:0ppsmall_higherversion detail

Name:                           Advanced Protocol Pack
Version:                        2.0
Publisher:                      Cisco Systems Inc.
NBAR Engine Version:            14
Creation time:                  Mon Jul 16 09:29:34 UTC 2012
Protocol Pack contents:
iana                    Mv: 1
base                    Mv: 4
tftp                    Mv: 2
```

The following sample output from the **show ip nbar protocol-pack** command shows information about the active Protocol Pack with an unlicensed Cisco image on a device:

```
Device# show ip nbar protocol-pack active

ACTIVE protocol pack:
Name:                           Standard Protocol Pack
Version:                        1.0
Publisher:                      Cisco Systems Inc.
```

Example: Viewing the NBAR2 Taxonomy Information

The following sample output from the **show ip nbar protocol-pack active taxonomy** command shows the information about the protocols in the active Protocol Pack:

```
Device# show ip nbar protocol-pack active taxonomy

Protocol Pack Taxonomy for Advanced Protocol Pack:
<?xml version="1.0"?>
<NBAR2-Taxonomy>
  <protocol>
    <name>active-directory</name>
    <engine-id>7</engine-id>
    <enabled>true</enabled>
```

```
    <selector-id>473</selector-id>
    <help-string>Active Directory Traffic</help-string>
    <global-id>L7:473</global-id>
    <common-name>Active Directory</common-name>
    <static>false</static>
    <attributes>
      <category>net-admin</category>
      <application-group>other</application-group>
      <p2p-technology>false</p2p-technology>
      <tunnel>false</tunnel>
      <encrypted>false</encrypted>
      <sub-category>network-management</sub-category>
    </attributes>
    <ip-version>
      <ipv4>true</ipv4>
      <ipv6>true</ipv6>
    </ip-version>

<references>http://www.microsoft.com/en-us/server-cloud/windows-server/active-directory.aspx</references>

    <id>1194</id>
    <underlying-protocols>cifs,ldap,ssl,ms-rpc</underlying-protocols>
    <long-description-is-final>true</long-description-is-final>
    <long-description>a directory service created by Microsoft for Windows domain networks,
 responsible for authenticating and authorizing all users and computers within a network
of Windows domain type, assigning and enforcing security policies for all computers in a
network and installing or updating software on network computers</long-description>
    <pdl-version>1</pdl-version>
    <uses-bundling>false</uses-bundling>
  </protocol>
  <protocol>
    <name>activesync</name>
    <engine-id>7</engine-id>
    <enabled>true</enabled>
    <selector-id>490</selector-id>
    <help-string>Microsoft Activesync protocol </help-string>
    <global-id>L7:490</global-id>
    <common-name>ActiveSync</common-name>
    <static>false</static>
    <attributes>
      <category>business-and-productivity-tools</category>
      <application-group>other</application-group>
      <p2p-technology>false</p2p-technology>
      <tunnel>false</tunnel>
      <encrypted>true</encrypted>
      <sub-category>client-server</sub-category>
    </attributes>
    <ip-version>
      <ipv4>true</ipv4>
      <ipv6>true</ipv6>
    </ip-version>
   <references>http://msdn.microsoft.com/en-us/library/dd299446(v=exchg.80).aspx</references>

    <id>1419</id>
    <underlying-protocols>http</underlying-protocols>
    <long-description-is-final>true</long-description-is-final>
    <long-description>ActiveSync is a mobile data synchronization technology and protocol
based on HTTP, developed by Microsoft. There are two implementations of the technology: one
 which synchronizes data and information with handheld devices with a specific desktop
computer, and another technology, commonly known as Exchange ActiveSync (or EAS), which
provides push synchronization of contacts, calendars, tasks, and email between
ActiveSync-enabled servers and devices.</long-description>
    <pdl-version>1</pdl-version>
    <uses-bundling>false</uses-bundling>
```

```
</protocol>
.
.
.
.
```

Example: Classifying SSL Sessions

The following example shows how an SSL-based service with the server name as 'finance.cisco.com' is matched using **unique-name**:

```
Device> enable
Device# configure terminal
Device(config)# class-map match-any cisco-finance
Device(config-cmap)# match protocol ssl unique-name finance.cisco.com
```

Additional References for NBAR2 Protocol Pack

Related Documents

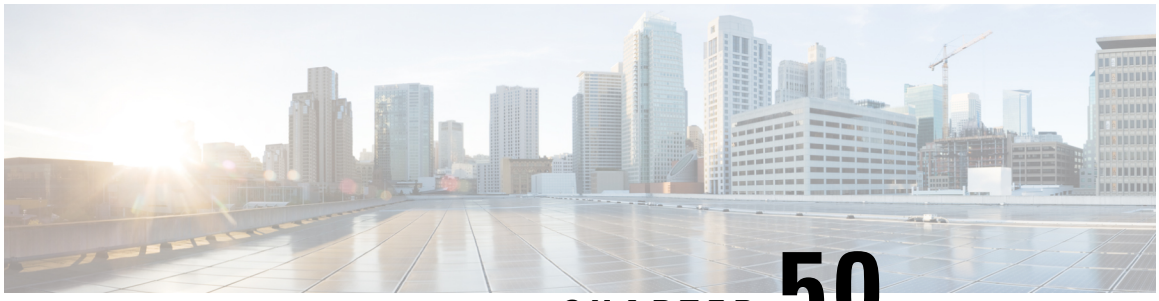
Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Cisco IOS LAN Switching commands	Cisco IOS LAN Switching Command Reference
Cisco IOS QoS configuration information	QoS Configuration Guide

Standards and RFCs

Standards/RFCs	Document Title
RFC 3551	RTP Profile for Audio and Video Conferences with Minimal Control
RFC 6101	The Secure Sockets Layer (SSL) Protocol Version 3.0

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 50

Enabling Protocol Discovery

Network-Based Application Recognition (NBAR) includes a feature called Protocol Discovery. Protocol discovery provides an easy way to discover the application protocol packets that are passing through an interface. When you configure NBAR, the first task is to enable protocol discovery.

This module contains concepts and tasks for enabling the Protocol Discovery feature.

Prerequisites for Enabling Protocol Discovery

Before enabling Protocol Discovery, read the information in the "Classifying Network Traffic Using NBAR" module.

Restrictions for Enabling Protocol Discovery

NBAR protocol discovery does not support the following:

- Asymmetric flows with stateful protocols.

Note In the NBAR context, asymmetric flows are the flows in which different packets of the flow go through different routers, for reasons such as load balancing implementation or asymmetric routing where packets flow through different routes to different directions.

- NBAR processing. By design, NBAR processing is temporarily disabled during the In-Service Software Upgrade (ISSU). The following syslog message indicates restart of NBAR classification once ISSU is complete.

"%NBAR_HA-5-NBAR_INFO: NBAR sync DONE!"

- Multicast packet classification.
- Multiprotocol Label Switching (MPLS)-labeled packets. NBAR classifies IP packets only. You can, however, use NBAR to classify IP traffic before the traffic is handed over to MPLS. Use the modular quality of service (QoS) CLI (MQC) to set the IP differentiated services code point (DSCP) field on the NBAR-classified packets and make MPLS map the DSCP setting to the MPLS experimental (EXP) setting inside the MPLS header.
- Non-IP traffic.
- Packets that originate from or that are destined to the router running NBAR.

NBAR is not supported on the following logical interfaces:

- Dialer interfaces
- Dynamic tunnels such as Dynamic Virtual Tunnel Interface (DVTI)
- IPv6 tunnels that terminate on the device
- MPLS
- Overlay Transport Virtualization (OTV) overlay interfaces

Note In cases where encapsulation is not supported by NBAR on some links, you can apply NBAR on other interfaces of the device to perform input classification. For example, you can configure NBAR on LAN interfaces to classify output traffic on the WAN link.

The following virtual interfaces are supported depending on the image of your Cisco IOS:

- Generic routing encapsulation (GRE)
- IPsec IPv4 tunnel (including tunneled IPv6) in protocol discovery mode and MQC mode
- IPsec IPv6 tunnel in protocol discovery mode but not in MQC mode
- Multipoint GRE/Dynamic Multipoint VPN (DMVPN) in protocol discovery mode

Note NBAR requires more CPU power when NBAR is enabled on tunneled interfaces.

If protocol discovery is enabled on both the tunnel interface and the physical interface on which the tunnel interface is configured, the packets that are designated to the tunnel interface are counted on both interfaces. On the physical interface, the packets are classified and are counted based on the encapsulation. On the tunnel interface, packets are classified and are counted based on the Layer 7 protocol.

Note You cannot use NBAR to classify output traffic on a WAN link where tunneling or encryption is used. Therefore, you should configure NBAR on other interfaces of the router (such as a LAN link) to perform input classification before the traffic is switched to the WAN link.

Information About Protocol Discovery

Protocol Discovery Overview

The Protocol Discovery feature of NBAR provides an easy way of discovering the application protocols passing through an interface so that appropriate QoS features can be applied.

NBAR determines which protocols and applications are currently running on your network. Protocol discovery provides an easy way of discovering the application protocols that are operating on an interface so that appropriate QoS features can be applied. With protocol discovery, you can discover any protocol traffic that is supported by NBAR and obtain statistics that are associated with that protocol.

Protocol discovery maintains the following per-protocol statistics for enabled interfaces:

- Total number of input packets and bytes
- Total number of output packets and bytes
- Input bit rates
- Output bit rates

These statistics can be used when you define classes and traffic policies (sometimes known as policy maps) for each traffic class. The traffic policies (policy maps) are used to apply specific QoS features and functionality to the traffic classes.

How to Enable Protocol Discovery

Enabling Protocol Discovery on an Interface

Perform this task to enable protocol discovery on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **ip nbar protocol-discovery**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [*name-tag*] **Example:** `Router(config)# interface fastethernet1/1/1`	Configures an interface type and enters interface configuration mode. • Enter the interface type and the interface number.
Step 4	**ip nbar protocol-discovery** **Example:** `Router(config-if)# ip nbar protocol-discovery`	Configures NBAR to discover traffic for all protocols that are known to NBAR on a particular interface. **Note** The **ipv4** and **ipv6** keywords are deprecated from the Cisco IOS XE Cupertino 17.10.1 release onwards. • Enables the protocol discovery statistics collection for both IPv4 and IPv6. • The **no** form of this command is not required to disable a keyword because the statistics collection is enabled for the specified keyword only.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode.

Reporting Protocol Discovery Statistics

Perform this task to display a report of the protocol discovery statistics per interface.

SUMMARY STEPS

1. **enable**
2. **show policy-map interface** *type number*
3. **show ip nbar protocol-discovery** [**interface** *type number*] [**stats** {**byte-count** | **bit-rate** | **packet-count**| **max-bit-rate**}] [**protocol** *protocol-name* | **top-n** *number*]
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**show policy-map interface** *type number* **Example:** `Router# show policy-map interface FastEthernet 1/1/1`	(Optional) Displays the packet and class statistics for all policy maps on the specified interface. • Enter the interface type and interface number.
Step 3	**show ip nbar protocol-discovery** [**interface** *type number*] [**stats** {**byte-count** | **bit-rate** | **packet-count**| **max-bit-rate**}] [**protocol** *protocol-name* | **top-n** *number*] **Example:** `Router# show ip nbar protocol-discovery interface Fastethernet1/1/1`	Displays the statistics gathered by the NBAR Protocol Discovery feature. • (Optional) Enter keywords and arguments to fine-tune the statistics displayed. For more information on each of the keywords, refer to the **show ip nbar protocol-discovery** command in Cisco IOS Quality of Service Solutions Command Reference.
Step 4	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Configuration Examples for Protocol Discovery

Example: Enabling Protocol Discovery on an Interface

In the following sample configuration, protocol discovery is enabled on Fast Ethernet interface 1/1/1:

```
Router> enable
Router# configure terminal
Router(config)# interface fastethernet1/1/1
Router(config-if)# ip nbar protocol-discovery
Router(config-if)# end
```

In the following sample configuration, protocol discovery is enabled on Fast Ethernet interface 1/1/2 for IPv6 packets:

```
Router> enable
```

```
Router# configure terminal

Router(config)# interface fastethernet1/1/2

Router(config-if)# ip nbar protocol-discovery

Router(config-if)# end
```

Example: Reporting Protocol Discovery Statistics

The following sample output from the **show ip nbar protocol-discovery** command displays the five most active protocols on the Fast Ethernet interface 2/0/1:

```
Router# show ip nbar protocol-discovery top-n 5

 FastEthernet2/0/1
                            Input                     Output
                            -----                     ------
  Protocol                  Packet Count              Packet Count
                            Byte Count                Byte Count
                            30sec Bit Rate (bps)      30sec Bit Rate (bps)
                            30sec Max Bit Rate (bps)  30sec Max Bit Rate (bps)
--------------------------- ------------------------  ------------------------
   rtp                      3272685                   3272685
                            242050604                 242050604
                            768000                    768000
                            2002000                   2002000
   gnutella                 513574                    513574
                            118779716                 118779716
                            383000                    383000
                            987000                    987000
   ftp                      482183                    482183
                            37606237                  37606237
                            121000                    121000
                            312000                    312000
   http                     144709                    144709
                            32351383                  32351383
                            105000                    105000
                            269000                    269000
   netbios                  96606                     96606
                            10627650                  10627650
                            36000                     36000
                            88000                     88000
   unknown                  1724428                   1724428
                            534038683                 534038683
                            2754000                   2754000
                            4405000                   4405000
   Total                    6298724                   6298724
                            989303872                 989303872
                            4213000                   4213000
                            8177000                   8177000
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Concepts and information about NBAR	"Classifying Network Traffic Using NBAR" module
MQC	"Applying QoS Features Using the MQC" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

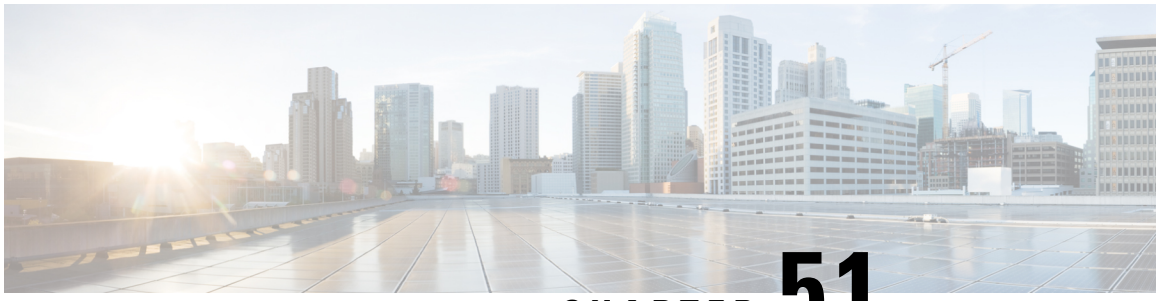
Feature Information for Enabling Protocol Discovery

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 62: Feature Information for Enabling Protocol Discovery

Feature Name	Releases	Feature Information
Protocol Discovery	Cisco IOS XE 2.1 Cisco IOS XE 3.3S	This feature was introduced on Cisco ASR 1000 Series Routers. The following sections provide information about this feature: The following commands were introduced: **ip nbar protocol discovery, show ip nbar protocol discovery.**



CHAPTER 51

Configuring NBAR Using the MQC

You can configure Network-Based Application Recognition (NBAR) using the functionality of the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC). The MQC uses traffic classes and traffic policies (policy maps) to apply QoS features to classes of traffic and applications recognized by NBAR.

This module contains concepts and tasks for configuring NBAR using the MQC.

Prerequisites for Configuring NBAR Using the MQC

Before configuring NBAR using the MQC, read the information in the "Classifying Network Traffic Using NBAR" module.

Information About NBAR Coarse-Grain Classification

NBAR and the MQC Functionality

To configure NBAR using the MQC, you must define a traffic class, configure a traffic policy (policy map), and then attach that traffic policy to the appropriate interface. These three tasks can be accomplished by using the MQC. The MQC is a command-line interface that allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach these traffic policies to interfaces.

In the MQC, the **class-map** command is used to define a traffic class (which is then associated with a traffic policy). The purpose of a traffic class is to classify traffic.

Using the MQC to configure NBAR consists of the following:

- Defining a traffic class with the **class-map** command.

- Creating a traffic policy by associating the traffic class with one or more QoS features (using the **policy-map** command).
- Attaching the traffic policy to the interface with the **service-policy** command.

A traffic class contains three major elements: a name, one or more **match** commands, and, if more than one **match** command exists in the traffic class, an instruction on how to evaluate these **match** commands (that is, **match-all** or **match-any**). The traffic class is named in the **class-map** command line; for example, if you enter the **class-map cisco** command while configuring the traffic class in the CLI, 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.

Note For NBAR, the **match protocol** commands are used to specify the match criteria, as described in the NBAR and the match protocol Commands, on page 734.

NBAR and the match protocol Commands

NBAR recognizes specific network protocols and network applications that are used in your network. Once a protocol or application is recognized by NBAR, you can use the MQC to group the packets associated with those protocols or applications into classes. These classes are grouped on the basis of whether the packets conform to certain criteria.

For NBAR, the criterion is whether the packet matches a specific protocol or application known to NBAR. Using the MQC, network traffic with one network protocol (citrix, for example) can be placed into one traffic class, while traffic that matches a different network protocol (gnutella, for example) can be placed into another traffic class. Later, the network traffic within each class can be given the appropriate QoS treatment by using a traffic policy (policy map).

You specify the criteria used to classify traffic by using a **match protocol**command. The table below lists some of the available **match protocol**commands and the corresponding protocol or traffic type recognized and supported by NBAR.

Note For a more complete list of the protocol types supported by NBAR, see the "Classifying Network Traffic Using NBAR" module.

Table 63: match protocol Commands and Corresponding Protocol or Traffic Type

match protocol Command[1]	Protocol Type
match protocol (NBAR)	Protocol type supported by NBAR
match protocol citrix	Citrix protocol
match protocol fasttrack	FastTrack peer-to-peer traffic

match protocol Command[1]	Protocol Type
match protocol gnutella	Gnutella peer-to-peer traffic
match protocol http	Hypertext Transfer Protocol
match protocol rtp	Real-Time Transport Protocol traffic
match protocol unknown [final]	All unknown and/or unclassified traffic

[1] Cisco IOS match protocol commands can vary by release. For more information, see the command documentation for the Cisco IOS release that you are using.

How to Configure NBAR Using the MQC

Configuring DSCP-Based Layer 3 Custom Applications

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *name* **transport** {**tcp** | **udp** | **udp-tcp** }**id** *id*
4. **dscp** *dscp-value*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar custom** *name* **transport** {**tcp**	**udp**	**udp-tcp** }**id** *id* **Example:** `Device(config)# ip nbar custom mycustom transport tcp id 100`	Specifies the transport protocol to match as TCP, UDP, or both TCP and UDP, and enters custom configuration mode.
Step 4	**dscp** *dscp-value* **Example:**	Specifies the differentiated service code points (DSCP) value.

	Command or Action	Purpose
	`Device(config-custom)# dscp ef`	**Note** In cases where two custom applications have the same filters, the priority is set according to the order of configuration.
Step 5	**exit** **Example:** `Device(config-custom)# exit`	Exits custom configuration mode.

Configuring NBAR Using the MQC

You can configure Network-Based Application Recognition (NBAR) using the functionality of the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC). The MQC uses traffic classes and traffic policies (policy maps) to apply QoS features to classes of traffic and applications recognized by NBAR.

This module contains concepts and tasks for configuring NBAR using the MQC.

Configuring a Traffic Policy

Traffic that matches a user-specified criterion can be organized into a specific class that can, in turn, receive specific user-defined QoS treatment when that class is included in a policy map.

To configure a traffic policy, perform the following steps.

Note The **bandwidth** command is shown in Step 5. The **bandwidth** command configures the QoS feature class-based weighted fair queuing (CBWFQ). CBWFQ is just an example of a QoS feature that can be configured. Use the appropriate command for the QoS feature that you want to use. As of Cisco IOS Release 12.2(18)ZY, CBWFQ is not supported on the Catalyst 6500 series switch that is equipped with a Supervisor 32/programmable intelligent services accelerator (PISA).

Note For Cisco IOS Release 12.2(18)ZY, an existing traffic policy (policy map) cannot be modified if the traffic policy is already attached to the interface. To remove the policy map from the interface, use the **no** form of the **service-policy** command.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **bandwidth** {*bandwidth-kbps*| **remaining percent** *percentage*| **percent** *percentage*}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map policy1`	Creates or modifies a policy map that can be attached to one or more interfaces and enters policy-map configuration mode. • Enter the name of the policy map.
Step 4	**class** {*class-name* | **class-default**} **Example:** `Device(config-pmap)# class class1`	Specifies the name of the class whose policy you want to create or change and enters policy-map class configuration mode. • Enter the specific class name or enter the **class-default**keyword.
Step 5	**bandwidth** {*bandwidth-kbps*| **remaining percent** *percentage*| **percent** *percentage*} **Example:** `Device(config-pmap-c)# bandwidth percent 50` **Example:**	(Optional) Specifies or modifies the bandwidth allocated for a class belonging to a policy map. • Enter the amount of bandwidth as a number of kbps, a relative percentage of bandwidth, or an absolute amount of bandwidth. **Note** The **bandwidth** command configures the QoS feature class-based weighted fair queuing (CBWFQ). CBWFQ is just an example of a QoS feature that can be configured. Use the appropriate command for the QoS feature that you want to use. **Note** As of Cisco IOS Release 12.2(18)ZY, CBWFQ is not supported on the Catalyst 6500 series switch that is equipped with a Supervisor 32/PISA.
Step 6	**end** **Example:** `Device(config-pmap-c)# end`	(Optional) Returns to privileged EXEC mode.

Attaching a Traffic Policy to an Interface or Subinterface

After a policy map is created, the next step is to attach the traffic policy (sometimes called a policy map) to an interface or subinterface. Traffic policies can be attached to either the input or output direction of the interface or subinterface.

Note Depending on the needs of your network, you may need to attach the traffic policy to an ATM PVC, a Frame Relay data-link connection identifier (DLCI), or other type of interface.

To attach a traffic policy (policy map) to an interface, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **pvc** [*name*] *vpi* / *vci* [**ilmi**| **qsaal**| **smds**| **l2transport**]
5. **exit**
6. **service-policy** {**input** | **output**} *policy-map-name*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [*name-tag*] **Example:** `Device(config)# interface ethernet 2/4`	Configures an interface type and enters interface configuration mode. • Enter the interface type and the interface number.
Step 4	**pvc** [*name*] *vpi* / *vci* [**ilmi**| **qsaal**| **smds**| **l2transport**] **Example:** `Device(config-if)# pvc cisco 0/16`	(Optional) Creates or assigns a name to an ATM permanent virtual circuit (PVC), specifies the encapsulation type on an ATM PVC, and enters ATM virtual circuit configuration mode. • Enter the PVC name, the ATM network virtual path identifier, and the network virtual channel identifier.

	Command or Action	Purpose
		Note This step is required only if you are attaching the policy map to an ATM PVC. If you are not attaching the policy map to an ATM PVC, advance to Step 6.
Step 5	**exit** **Example:** `Device(config-atm-vc)# exit`	(Optional) Returns to interface configuration mode. **Note** This step is required only if you are attaching the policy map to an ATM PVC and you completed Step 4. If you are not attaching the policy map to an ATM PVC, advance to Step 6.

Step 6	**service-policy** {**input** | **output**} *policy-map-name*
	Example:
	`Device(config-if)# service-policy input policy1`
		Attaches a policy map (traffic policy) to an input or output interface.
		• Specify either the **input** or **output** keyword, and enter the policy map name.
		Note Policy maps can be configured on ingress or egress Devices. They can also be attached in the input or output direction of an interface. The direction (input or output) and the Device (ingress or egress) to which the policy map should be attached vary according your network configuration. When using the **service-policy** command to attach the policy map to an interface, be sure to choose the Device and the interface direction that are appropriate for your network configuration.
		Note After you use the **service-policy** command, you may see two messages similar to the following:
		`%PISA-6-NBAR_ENABLED: feature accelerated on input direction of: [interface name and type ]`
		`%PISA-6-NBAR_ENABLED: feature accelerated on output direction of: [interface name and type`
		While both of these messages appear, NBAR is enabled in the direction specified by the **input** or **output** keyword only.

Step 7	**end** **Example:** `Device(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Verifying NBAR Using the MCQ

After you create the traffic classes and traffic policies (policy maps), you may want to verify that the end result is the one you intended. That is, you may want to verify whether your traffic is being classified correctly and whether it is receiving the QoS treatment as intended. You may also want to verify that the protocol-to-port mappings are correct.

To verify the NBAR traffic classes, traffic policies, and protocol-to-port mappings, perform the following steps.

SUMMARY STEPS

1. **show class-map** [*class-map-name*]
2. **show policy-map** [*policy-map*]
3. **show policy-map interface** *type number*
4. **show ip nbar port-map** [*protocol-name*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**show class-map** [*class-map-name*] **Example:** `Device# show class-map`	(Optional) Displays all class maps and their matching criteria. • (Optional) Enter the name of a specific class map.
Step 2	**show policy-map** [*policy-map*] **Example:** `Device# show policy-map`	(Optional) Displays the configuration of all classes for a specified service policy map or all classes for all existing policy maps. • (Optional) Enter the name of a specific policy map.
Step 3	**show policy-map interface** *type number* **Example:** `Device# show policy-map interface Fastethernet 6/0`	(Optional) Displays the packet and class statistics for all policy maps on the specified interface. • Enter the interface type and the interface number.
Step 4	**show ip nbar port-map** [*protocol-name*] **Example:** `Device# show ip nbar port-map`	(Optional) Displays the current protocol-to-port mappings in use by NBAR. • (Optional) Enter a specific protocol name.

Verifying Unknown and Unclassified Traffic Management

To verify the management of unknown and unclassified traffic, perform the following steps.

SUMMARY STEPS

1. **show ip nbar protocol-id unknown**
2. **show ip nbar link-age unknown**

3. **show ip nbar protocol-attribute unknown**

DETAILED STEPS

	Command or Action	Purpose

Step 1	**show ip nbar protocol-id unknown**
	Example:
	`Device# show ip nbar protocol-id unknown`
	`Protocol Name id type`
	`unknown 1 L7 STANDARD`
		(Optional) Displays protocol classification ID for unknown and unclassified traffic.

Step 2	**show ip nbar link-age unknown**
	Example:
	`Device# show ip nbar link-age unknown`
	`Protocol Link Age (seconds)`
	`unknown 60`
		(Optional) Displays the protocol link age for unknown and unclassified traffic.

Step 3	**show ip nbar protocol-attribute unknown**
	Example:
	`Device# show ip nbar protocol-attribute unknown`
	`Protocol Name : unknown`
	`encrypted : encrypted-no`
	`tunnel : tunnel-no`
	`category : other`
	`sub-category : other`
	`application-group : other`
	`p2p-technology : p2p-tech-no`
		(Optional) Displays list of configured attributes for unknown and unclassified traffic.

Configuration Examples for Configuring DSCP-Based Layer 3 Custom Applications

Example Configuring a Traffic Class

In the following example, a class called cmap1 has been configured. All traffic that matches the citrix protocol will be placed in the cmap1 class.

```
Device> enable

Device# configure terminal
```

```
Device(config)# class-map cmap1
Device(config-cmap)# match protocol citrix
Device(config-cmap)# end
```

Example Configuring a Traffic Policy

In the following example, a traffic policy (policy map) called policy1 has been configured. Policy1 contains a class called class1, within which CBWFQ has been enabled.

```
Device> enable
Device# configure terminal
Device(config)# policy-map policy1
Device(config-pmap)# class class1
Device(config-pmap-c)# bandwidth percent 50
Device(config-pmap-c)# end
```

Note In the above example, the **bandwidth** command is used to enable Class-Based Weighted Fair Queuing (CBWFQ). CBWFQ is only an example of one QoS feature that can be applied in a policy map. Use the appropriate command for the QoS feature that you want to use. As of Cisco IOS Release 12.2(18)ZY, CBWFQ is not supported on the Catalyst 6500 series switch that is equipped with a Supervisor 32/PISA.

Example Attaching a Traffic Policy to an Interface or Subinterface

In the following example, the traffic policy (policy map) called policy1 has been attached to Ethernet interface 2/4 in the input direction of the interface.

```
Device> enable
Device# configure terminal
Device(config)# interface ethernet 2/4
Device(config-if)# service-policy input policy1
Device(config-if)# end
```

Example Verifying the NBAR Protocol-to-Port Mappings

The following is sample output of the **show ip nbar port-map** command. This command displays the current protocol-to-port mappings in use by NBAR. Use the display to verify that these mappings are correct.

```
Device# show ip nbar port-map
port-map bgp      udp 179
port-map bgp      tcp 179
port-map cuseeme  udp 7648 7649
port-map cuseeme  tcp 7648 7649
port-map dhcp     udp 67 68
port-map dhcp     tcp 67 68
```

If the **ip nbar port-map** command has been used, the **show ip nbar port-map** command displays the ports assigned to the protocol.

If the **no ip nbar port-map**command has been used, the **show ip nbar port-map** command displays the default ports. To limit the display to a specific protocol, use the *protocol-name* argument of the **show ip nbar port-map** command.

Example: L3 Custom any IP Port

```
Device> enable
Device# configuration terminal
Device (config)# ip nbar custom mycustom transport udp-tcp
Device(config-custom)# dscp ef
Device (config-custom)# exit
```

Where to Go Next

To add application recognition modules (also known as Packet Description Language Modules or PDLMs) to your network, see the "Adding Application Recognition Modules" module.

To classify network traffic on the basis of a custom protocol, see the "Creating a Custom Protocol" module.

Additional References

The following sections provide references related to configuring NBAR using the MQC.

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*

Related Topic	Document Title
QoS features and functionality on the Catalyst 6500 series switch	"Configuring PFC QoS" chapter of the *Catalyst Supervisor Engine 32 PISA Cisco IOS Software Configuration Guide* , Release 12.2ZY
MQC, traffic policies (policy maps), and traffic classes	"Applying QoS Features Using the MQC" module
CBWFQ	"Configuring Weighted Fair Queueing" module
Concepts and information about NBAR	"Classifying Network Traffic Using NBAR" module
Information about enabling Protocol Discovery	"Enabling Protocol Discovery" module
Information about adding application recognition modules (also known as PDLMs)	"Adding Application Recognition Modules" module
Creating a custom protocol	"Creating a Custom Protocol" module

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

Feature Information for Configuring NBAR Using the MQC

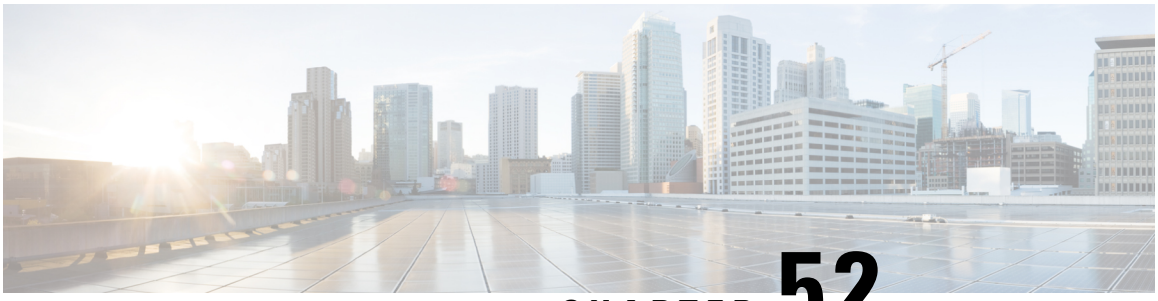
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 64: Feature Information for Configuring NBAR Using the MQC

Feature Name	Releases	Feature Information
NBAR MQC Support for Pre-resolved and Unknown Applications	IOS Release 15.5(1)T IOS XE Release 3.14S	The NBAR MQC Support for Pre-resolved and Unknown Applications feature provides support for matching all unknown and unclassified traffic using MQC. The following commands were modified: **class-map**, **match protocol**
QoS: DirectConnect PDLM	12.4(4)T	Provides support for the DirectConnect protocol and Packet Description Language Module (PDLM). The DirectConnect protocol can now be recognized when using the MQC to classify traffic. The following sections provide information about the QoS: DirectConnect PDLM feature:
QoS: Skype Classification	12.4(4)T	Provides support for the Skype protocol. The Skype protocol can now be recognized when using the MQC to classify traffic. **Note** Cisco currently supports Skype Version 1 only. The following sections provide information about the QoS: Skype Classification feature:
NBAR--BitTorrent PDLM	12.4(2)T	Provides support for the BitTorrent PDLM and protocol. The BitTorrent protocol can now be recognized when using the MQC to classify traffic. The following sections provide information about the NBAR-BitTorrent PDLM feature:
NBAR--Citrix ICA Published Applications	12.4(2)T	Enables NBAR to classify traffic on the basis of the Citrix Independent Computing Architecture (ICA) published application name and tag number. The following sections provide information about the NBAR-Citrix ICA Published Applications feature:
NBAR--Multiple Matches Per Port	12.4(2)T	Provides the ability for NBAR to distinguish between values of an attribute within the traffic stream of a particular application on a TCP or UDP port. The following sections provide information about the NBAR-Multiple Matches Per Port feature:
NBAR Extended Inspection for HTTP Traffic	12.3(4)T	Allows NBAR to scan TCP ports that are not well known and identify HTTP traffic that traverses these ports. The following sections provide information about the NBAR Extended Inspection for HTTP Traffic feature:

Feature Name	Releases	Feature Information
NBAR Real-Time Transport Protocol Payload Classification	12.2(15)T	Enables stateful identification of real-time audio and video traffic. The following section provides information about the NBAR Real-Time Transport Protocol Payload Classification feature:
NBAR--Network-Based Application Recognition	12.2(18)ZYA	Integrates NBAR and Firewall Service Module (FWSM) functionality on the Catalyst 6500 series switch that is equipped with a Supervisor 32/programmable intelligent services accelerator (PISA). Additional protocols are now recognized by NBAR. The following sections provide information about the NBAR feature: The following command was modified: **match protocol (NBAR)**.
NBAR--Network-Based Application Recognition (Hardware Accelerated NBAR)	12.2(18)ZY	Enables NBAR functionality on the Catalyst 6500 series switch that is equipped with a Supervisor 32/programmable intelligent services accelerator (PISA). The following section provides information about the NBAR--Network-Based Application Recognition (Hardware Accelerated NBAR) feature:



CHAPTER 52

DSCP-Based Layer 3 Custom Applications

Network-Based Application Recognition (NBAR) supports the use of custom protocols to identify customer-specific applications and applications that NBAR does not support. IP address and port-based custom protocol includes supporting an IP subnet or a list of IP addresses with a specific TCP or UDP transport. DSCP-Based Layer 3 Custom Applications feature is an enhancement that enables the customer to identify traffic that belongs to Layer 3 or Layer 4 custom applications by using Differentiated Services Code Point (DSCP) values in the traffic.

Restriction of DSCP-Based Layer 3 Custom Applications

DSCP-Based Layer 3 Custom Applications feature treats the Differentiated Services Code Point (DSCP) classification as a property of the flow and checks only the DSCP value of the first packet in the flow. To identify different packets in the flow and apply policies on them, use the **match dscp** command.

DSCP-Based Layer 3 Custom Applications Overview

Network-Based Application Recognition (NBAR) supports the use of custom protocols to identify customer specific applications and applications that NBAR does not support. IP address and port-based custom protocol includes supporting an IP subnet or a list of IP addresses with a specific TCP or UDP transport. DSCP-based Layer 3 Custom Application feature is an enhancement that enables the customer to identify traffic that belongs to Layer 3 or Layer 4 custom applications by using Differentiated Services Code Point (DSCP) values in the traffic. You define a custom protocol transport by using the keywords and arguments of the **ip nbar custom transport** command.

How to Configure NBAR2 Auto-learn

Configuring DSCP-Based Layer 3 Custom Applications

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *name* **transport** {**tcp** | **udp** | **udp-tcp** }**id** *id*
4. **dscp** *dscp-value*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar custom** *name* **transport** {**tcp** | **udp** | **udp-tcp** }**id** *id* **Example:** `Device(config)# ip nbar custom mycustom transport tcp id 100`	Specifies the transport protocol to match as TCP, UDP, or both TCP and UDP, and enters custom configuration mode.
Step 4	**dscp** *dscp-value* **Example:** `Device(config-custom)# dscp ef`	Specifies the differentiated service code points (DSCP) value. **Note** In cases where two custom applications have the same filters, the priority is set according to the order of configuration.
Step 5	**exit** **Example:** `Device(config-custom)# exit`	Exits custom configuration mode.

Configuration Examples for Configuring DSCP-Based Layer 3 Custom Applications

Example: DSCP-Based Layer 3 Custom Applications

```
Device> enable
Device# configuration terminal
Device (config)# ip nbar custom mycustom transport tcp id 100
Device(config-custom)# dscp ef
Device (config-custom)# exit
```

Example: L3 Custom any IP Port

```
Device> enable
Device# configuration terminal
Device (config)# ip nbar custom mycustom transport udp-tcp
Device(config-custom)# dscp ef
Device (config-custom)# exit
```

Additional References for DSCP-Based Layer 3 Custom Applications

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	Cisco IOS Quality of Service Solutions Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for DSCP-based Layer 3 Custom Applications

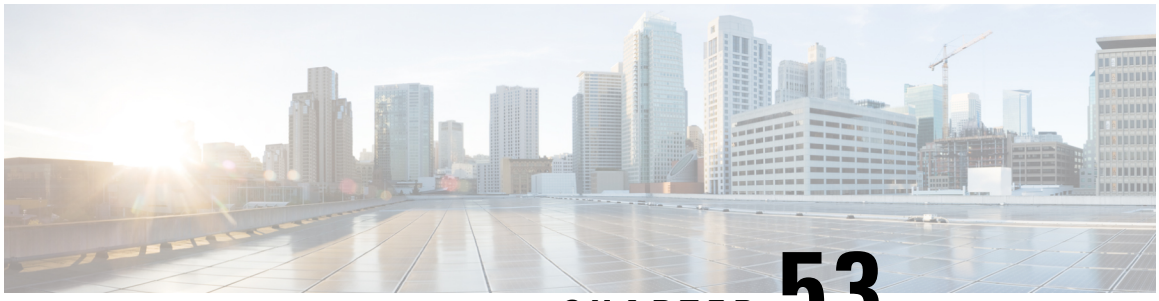
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 65:

Feature Name	Releases	Feature Information
DSCP-based Layer 3 Custom Applications	15.5(2)T, 15.5(3)T	NBAR supports the use of custom protocols to identify customer specific applications and applications that NBAR does not support. IP address and port-based custom protocol includes supporting an IP subnet or a list of IP addresses with a specific TCP or UDP transport. DSCP-based Layer 3 Custom Application feature is an enhancement that enables the customer to identify traffic that belongs to Layer 3 or Layer 4 custom applications by using DSCP values in the traffic. The following command was introduced or modified: **ip nbar custom**

Feature Name	Releases	Feature Information
L3 custom any IP/Port	Cisco IOS XE 3.16S	NBAR supports the use of custom protocols to identify customer specific applications and applications that NBAR does not support. IP address and port-based custom protocol includes supporting an IP subnet or a list of IP addresses with a specific TCP or UDP transport or TCP and UDP transport. DSCP-based Layer 3 Custom Application feature is an enhancement that enables the customer to identify traffic that belongs to Layer 3 or Layer 4 custom applications by using DSCP values in the traffic. The L3 Custom any IP/Port feature is an enhancement that enable users to to configure L3 or L4 custom applications over non UDP/TCP or over both UDP and TCP transport. The following command was introduced or modified: **ip nbar custom**



CHAPTER 53

MQC Based on Transport Hierarchy

The MQC Based on Transport Hierarchy(TPH) feature enables the use of TPH to apply policies according to a specific underlying protocol, instead of only according to the final classified protocol, for example, an email application over HTTP. A new MQC filter configured within a class-map matches all traffic which has this protocol in the hierarchy.

Restrictions for MQC Based on Transport Hierarchy

- The MQC Based on Transport Hierarchy feature is supported only for DNS, HTTP, RTP, and SSL.
- Does not allow adding the match of the protocol and in-app-hierarchy to the same class-map.
- Match protocol http in-app-hierarchy and match protocol rtp in-app-hierarchy are not supported while match protocol attribute tunnel is configured, even on a different class-map.

Information About MQC Based on Transport Hierarchy

MQC Based on Transport Hierarchy Overview

The MQC based on transport hierarchy(TPH) feature enables NBAR to use TPH to apply policies according to a specific underlying protocol, instead of only according to the final classified protocol. The TPH of a particular application is the stack of protocols on which the application is delivered. For example, an application is being transported over HTTP and HTTP runs over TCP.

Prior to the configuartion of the MQC based on transport hierarchy(TPH) feature, it is only possible to apply a class-map filter on the final classified protocol using the **match protocol** *protocol-id* class-map filter. However, to apply QoS policies on all the traffic of HTTP, then include all the protocols which run over HTTP into the class-map makes the configuration of such use-cases considerably difficult. A solution for this problem

is an in-app-hierarchy class-map filter which uses TPH to apply policies according to a specific underlying protocol, instead of only according to the final classified protocol. For example, the rule **match protocol** *http* **in-app-hierarchy** matches if HTTP is present in the hierarchy.

How to Configure MQC Based on Transport Hierarchy

Configuring MQC Based on Transport Hierarchy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** [**match-all** | **match-any**] *class-map-name*
4. **match protocol** *protocol-name* **in-app-hierarchy**
5. **end**
6. **configure terminal**
7. **policy-map** *policy-map-name*
8. **class** { *class-name* | **class-default** }
9. **end**
10. **configure terminal**
11. **interface** *type number*
12. **service-policy** { **input** | **output** } *policy-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**class-map** [**match-all**	**match-any**] *class-map-name* **Example:** `Device(config)# class-map match-all C1`	Creates a class map to be used for matching packets to a specified class and enters QoS class-map mode. • Enter the name of the class map.
Step 4	**match protocol** *protocol-name* **in-app-hierarchy** **Example:** `Device(config-cmap)# match protocol http in-app-hierarchy`	Configures the match criterion for a class map on the basis of the specified protocol. The keyword **in-app-hierarchy** matches if the protocol is present in the transport hierarchy. Possible values for *protocol-name*: DNS, HTTP, RTP, SSL

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config-cmap)# end`	Exits class-map mode and returns to privileged EXEC mode.
Step 6	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 7	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map P1`	Specifies the name of the policy map and enters policy-map configuration mode.
Step 8	**class** { *class-name* | **class-default** } **Example:** `Device(config-pmap)# class C1`	Specifies the name of the class whose policy you want to create and enters policy-map class configuration mode.
Step 9	**end** **Example:** `Device(config-cmap)# end`	Exits class-map mode and returns to privileged EXEC mode.
Step 10	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 11	**interface** *type number* **Example:** `Device(config)# interface GigabitEthernet 0/0/1`	Configures an interface type and enters interface configuration mode.
Step 12	**service-policy** { **input** | **output** } *policy-map-name* **Example:** `Device(config-if)# service-policy input P1`	Specifies the name of the policy map to be attached to the input or output direction of the interface.

Verifying MQC Based on Transport Hierarchy

To verify the MQC Based on Transport Hierarchy feature perform the following steps:

SUMMARY STEPS

1. **enable**
2. **show policy-map interface** *type number*
3. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device# enable`	(Optional) Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show policy-map interface** *type number* **Example:** `Device# show policy-map interface GigabitEthernet0/0/1`	Displays the packet statistics of all classes that are configured for allservice policies either on the specified interface • Enter the interface type and the interface number.
Step 3	**exit** **Example:** `Device# exit`	(Optional) Exits privileged EXEC mode.

Configuration Examples for MQC Based on Transport Hierarchy

Example: Configuring MQC Based on Transport Hierarchy

The following is an example of the configuring MQC based on Transport Hierarchy feature:

```
Device> enable
Device# configure terminal
Device(config)# class-map match-all C1
Device(config-cmap)# match protocol http in-app-hierarchy
Device(config-cmap)# match protocol youtube
Device(config-cmap)# end
Device# configure terminal
Device(config)# policy-map P1
Device(config-pmap)# class C1
Device(config-cmap)# end
Device# configure terminal
Device(config)# interface GigabitEthernet 0/0/1
Device(config-if)# service-policy input P1
```

A traffic policy called P1 is configured. P1 contains a class called C1 for which QoS bandwidth limitation is configured as an example. All traffic that has final classification of Youtube with HTTP as a transport will be placed in the C1 class. Other possible transports for Youtube, such as DNS, SSL or RTSP, will not be matched by this class-map

Example: Verifying the MQC Based on Transport Hierarchy configuration

The following is a sample output from the **show policy-map interface** command:

```
Device#  show policy-map interface GigabitEthernet0/0/1
```

```
GigabitEthernet0/0/1
  Service-policy input: P1

Class-map: C1 (match-all)
      17 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: protocol http in-app-hierarchy
      Match: protocol youtube

    Class-map: class-default (match-any)
      3 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	Cisco IOS Quality of Service Solutions Command Reference

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

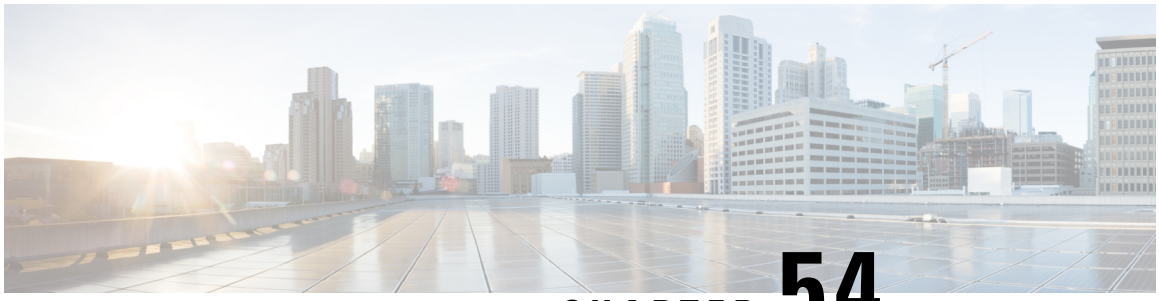
Feature Information for MQC Based on Transport Hierarchy

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 66: Feature Information for MQC Based on Transport Hierarchy

Feature Name	Releases	Feature Information
MQC Based on Transport Hierarchy		The MQC Based on Transport Hierarchy feature enables the use of Transport Hierarchy to apply policies according to a specific underlying protocol, instead of only according to the final classified protocol. A new MQC filter is introduced which can be configured within a class-map. The following command was modified: **match protocol**
Transport Hierarchy support for DNS	Cisco IOS XE Denali 16.3	The **match protocol** CLI can match according to the following protocol types: DNS, HTTP, SSL, and RTP. Example: **match protocol dns in-app-hierarchy**



CHAPTER 54

NBAR Categorization and Attributes

NBAR Categorization and Attributes feature provides the mechanism to match protocols or applications based on statically assigned attributes such as application-group, category, sub-category, encrypted and tunnel. Categorizing the protocols and applications into different groups helps with reporting and applying Quality of Service (QoS) policies.

Information About NBAR2 Custom Protocol

NBAR Categorization and Attributes

The NBAR Categorization and Attributes feature provides the mechanism to match protocols or applications based on certain attributes. Categorizing the protocols and applications into different groups will help with reporting and performing group actions, such as applying QoS policies, on them. Attributes are statically assigned to each protocol or application, and they are not dependent on the traffic. The following attributes are available to configure the match criteria using the **match protocol attribute** command:

- **application-group**: The **application-group** keyword allows the configuration of applications grouped together based on the same networking application as the match criteria. For example, Yahoo-Messenger, Yahoo-VoIP-messenger, and Yahoo-VoIP-over-SIP are grouped together under the yahoo-messenger-group.
- **category**: The **category** keyword allows you to configure applications that are grouped together based on the first level of categorization for each protocol as the match criteria. Similar applications are grouped together under one category. For example, the email category contains all email applications such as, Internet Mail Access Protocol (IMAP), Simple Mail Transfer Protocol (SMTP), Lotus Notes, and so forth.
- **sub-category**: The **sub-category** keyword provides the option to configure applications grouped together based on the second level of categorization for each protocol as the match criteria. For example, clearcase, dbase, rda, mysql and other database applications are grouped under the database group.

- **encrypted**: The **encrypted** keyword provides the option to configure applications grouped together based on whether the protocol is an encrypted protocol or not as the match criteria. Applications are grouped together based on the encrypted and nonencrypted status of the applications. Protocols for which the NBAR does not provide any value are categorized under the unassigned encrypted group.
- **tunnel**: The **tunnel** keyword provides the option to configure protocols based on whether or not a protocol tunnels the traffic of other protocols. Protocols for which the NBAR does not provide any value are categorized under the unassigned tunnel group. For example, Layer 2 Tunneling Protocols (L2TP).
- **p2p-technology**: The **p2p(Peer-to-Peer)-technology** attribute provides the option to indicate whether or not a protocol uses p2p technology.

Note Attribute-based protocol match configurations do not impact the granularity of classification either in reporting or in the Protocol Discovery information.

You can create custom values for the attributes application-group, category, and sub-category. The custom values enable you to name the attributes based on grouping of protocols. Use the **ip nbar attribute application-group custom application-group-name**, **ip nbar attribute category custom category-name**, and **ip nbar attribute sub-category custom sub-category-name** commands to add custom values for the attributes application-group, category, and sub-category, respectively.

The dynamically created custom attribute values can be used for attribute-map creation when using the **ip nbar attribute-map** command, and for configuring the match criterion for a class-map when using the **match protocol attribute** command.

The output from the **show ip nbar attribute-custom** command displays the number of custom values that can be defined for attributes, and the custom values that are currently defined. The **show ip nbar attribute** command displays all the attributes including the custom attributes used by NBAR.

To remove the custom values, use the **no ip nbar attribute** command.

Overview of NBAR2 Custom Protocol

Network-Based Application Recognition (NBAR) supports the use of custom protocols to identify custom applications. Custom protocols support static port-based protocols and applications that NBAR does not support.

For more information about custom protocols, refer to "Creating a Custom Protocol" module.

How to Configure NBAR2 Custom Protocol

Customizing NBAR Attributes

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar attribute-map** *profile-name*

4. [**attribute category** *category-name*]
5. [**attribute sub-category** *sub-category-name*]
6. [**attribute application-group** *application-group-name*]
7. [**attribute tunnel** *tunnel-info*]
8. [**attribute encrypted** *encrypted-info*]
9. [**attribute traffic-class** *traffic-class*]
10. [**attribute business-relevance** *business-relevance*]
11. [**attribute p2p-technology** *p2p-technology-info*]
12. **ip nbar attribute-set** *protocol-name profile-name*
13. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar attribute-map** *profile-name* **Example:** `Device(config)# ip nbar attribute-map actdir-attrib`	Creates an attribute profile with the name that you specify, and enters the attribute-map configuration mode.
Step 4	[**attribute category** *category-name*] **Example:** `Device(config-attribute-map)# attribute category net-admin`	Adds attribute values from the application-group attribute, on to your profile.
Step 5	[**attribute sub-category** *sub-category-name*] **Example:** `Device(config-attribute-map)# attribute sub-category network-management`	Adds attribute values from the sub-category attribute, on to your profile.
Step 6	[**attribute application-group** *application-group-name*] **Example:** `Device(config-attribute-map)# attribute application-group other`	Adds attribute values from the application-group attribute, on to your profile.

	Command or Action	Purpose
Step 7	[**attribute tunnel** *tunnel-info*] **Example:** `Device(config-attribute-map)# attribute tunnel no`	Adds attribute values from the tunnel attribute, on to your profile.
Step 8	[**attribute encrypted** *encrypted-info*] **Example:** `Device(config-attribute-map)# attribute encrypted no`	Adds attribute values from the encrypted attribute, on to your profile.
Step 9	[**attribute traffic-class** *traffic-class*] **Example:** `Device(config-attribute-map)# attribute traffic-class multimedia-conferencing`	Adds traffic-class attribute value to the profile.
Step 10	[**attribute business-relevance** *business-relevance*] **Example:** `Device(config-attribute-map)# attribute business-relevance business-relevant`	Adds business-relevance attribute value to the profile.
Step 11	[**attribute p2p-technology** *p2p-technology-info*] **Example:** `Device(config-attribute-map)# attribute p2p-technology no`	Adds attribute values from the p2p-technology attribute, on to your profile.
Step 12	**ip nbar attribute-set** *protocol-name profile-name* **Example:** `Device(config-attribute-map)# ip nbar attribute-set active-directory actdir-attrib`	Adds attribute values from the specified profile to the specified protocol.
Step 13	**end** **Example:** `Device(config-attribute-map)# end`	Returns to privileged EXEC mode.

Configuration Examples for NBAR2 Custom Protocol

Example: Adding Custom Values for Attributes

The following example shows how to add custom values for the attributes application-group, category, and sub-category:

```
Device> enable
Device# configure terminal
Device(config)# ip nbar attribute application-group custom Home_grown_finance_group "our
finance tools network traffic"
Device(config)# ip nbar attribute category custom dc_backup_category "Data center backup
traffic"
Device(config)# ip nbar attribute sub-category custom hr_sub_category "HR custom applications
 traffic"
Device(config)# exit
```

Examples: Viewing the Information About Custom Values for Attributes

The following sample output from the **show ip nbar attribute-custom** command displays the number of custom values that can be defined, and the custom values that are currently defined for the attributes:

```
Device# show ip nbar attribute-custom

                 Name :  category
                 Help :  category attribute
   Custom Groups Limit :  1
 Custom Groups Created :  dc_backup_category

                 Name :  sub-category
                 Help :  sub-category attribute
   Custom Groups Limit :  1
 Custom Groups Created :  hr_sub_category

                 Name :  application-group
                 Help :  application-group attribute
   Custom Groups Limit :  1
 Custom Groups Created :  Home_grown_finance_group
```

The following sample output from the **show ip nbar attribute category** command displays the details about the Category attribute:

```
Device# show ip nbar attribute category

      Name :  category
      Help :  category attribute
      Type :  group
    Groups :  newsgroup
           :  instant-messaging
           :  net-admin
           :  trojan
           :  email
           :  file-sharing
           :  industrial-protocols
           :  business-and-productivity-tools
```

```
          :  internet-privacy
          :  social-networking
          :  layer3-over-ip
          :  obsolete
          :  streaming
          :  location-based-services
          :  voice-and-video
          :  other
          :  gaming
          :  browsing
          :  dc_backup_category
   Need :  Mandatory
   Default :  other
```

Example: Creating a Profile and Configuring Attributes for the Profile

The following example shows how to create an attribute profile with attributes configured for the Network News Transfer Protocol (NNTP) protocol:

```
Device> enable
Device# configure terminal
Device(config)# ip nbar attribute-map nntp-attrib
Device(config-attribute-map)# attribute category newsgroup
Device(config-attribute-map)# attribute application-group nntp-group
Device(config-attribute-map)# attribute tunnel tunnel-no
Device(config-attribute-map)# attribute encrypted encrypted-yes
Device(config-attribute-map)# attribute p2p-technology p2p-tech-no
Device(config-attribute-map)# end
```

The following example shows how to verify the above configuration:

```
Device> enable
Device# show ip nbar attribute-map nntp-attrib
Device# Profile Name :  nntp-attrib
            category :  newsgroup
   application-group :  nntp-group
           encrypted :  encrypted-yes
Device# end
```

Example: Attaching an Attribute Profile to a Protocol

The following example shows how to set an attribute profile to the Application Communication Protocol (ACP) protocol:

```
Device> enable
Device# configure terminal
Device(config)# ip nbar attribute-set acp test-profile
Device(config)# exit
```

Additional References for NBAR2 Custom Protocol

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Cisco IOS LAN switching commands	Cisco IOS LAN Switching Command Reference
Cisco IOS QoS configuration information	*QoS Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for NBAR Categorization and Attributes

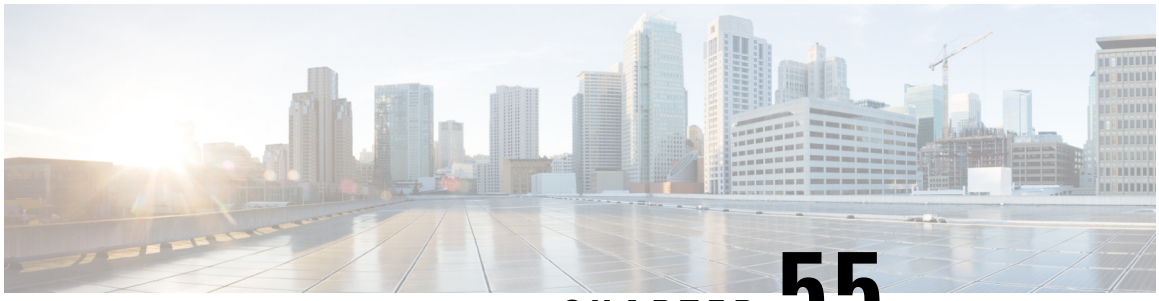
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 67: Feature Information for NBAR2 Custom Protocol

Feature Name	Releases	Feature Information
NBAR Categorization and Attributes	Cisco IOS XE Release 3.4S	This feature was introduced on Cisco ASR 1000 series Aggregation Services Routers. The following command was introduced or modified: **ip nbar custom**

Feature Name	Releases	Feature Information
NBAR2 Custom Protocol	Cisco IOS XE Release 3.8S	The NBAR2 Custom Protocol feature configures attributes profiles for protocols, and maps profiles to protocols. The following command was introduced or modified: **ip nbar attribute-map**, **ip nbar attribute-set**.



CHAPTER 55

Reporting Extracted Fields Through Flexible NetFlow

The Reporting Extracted Fields Through Flexible NetFlow feature allows Network-Based Application Recognition (NBAR) to send subapplication table fields to the collector through Flexible NetFlow.

Information About Reporting Extracted Fields Through Flexible NetFlow

Subapplication Table Fields

Use the **option sub-application-table** command to send an options table periodically to the collector, thereby enabling the collector to map NBAR subapplication tags, subapplication names, and subapplication descriptions provided in the flow records to application IDs.

How to Report Extracted Fields Through Flexible NetFlow

Reporting Subapplication Table Fields

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **flow exporter** *exporter-name*
4. **option sub-application-table**

5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**flow exporter** *exporter-name* **Example:** `Device(config)# flow exporter EXPORTER-1`	Enters Flexible NetFlow flow exporter configuration mode.
Step 4	**option sub-application-table** **Example:** `Device(config-flow-exporter)# option sub-application-table`	Enables periodic sending of an options table that allows the collector to map NBAR subapplication tags, subapplication names, and subapplication descriptions provided in flow records to application IDs.
Step 5	**exit** **Example:** `Device(config-flow-exporter)# exit`	Exits Flexible NetFlow flow exporter configuration mode and returns to global configuration mode.

Configuration Examples for Reporting Extracted Fields Through Flexible NetFlow

Example: Reporting Subapplication Fields

The following example shows how to enable the periodic sending of an options table, which allows the collector to map NBAR subapplication tags, subapplication names, and subapplication descriptions provided in the flow records to application IDs:

```
Device(config)# flow exporter FLOW-EXPORTER-1
Device(config-flow-exporter)# option sub-application-table
```

Additional References

The following sections provide references related to configuring NBAR using the MQC.

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
QoS features and functionality on the Catalyst 6500 series switch	"Configuring PFC QoS" chapter of the *Catalyst Supervisor Engine 32 PISA Cisco IOS Software Configuration Guide* , Release 12.2ZY
MQC, traffic policies (policy maps), and traffic classes	"Applying QoS Features Using the MQC" module
CBWFQ	"Configuring Weighted Fair Queueing" module
Concepts and information about NBAR	"Classifying Network Traffic Using NBAR" module
Information about enabling Protocol Discovery	"Enabling Protocol Discovery" module
Information about adding application recognition modules (also known as PDLMs)	"Adding Application Recognition Modules" module
Creating a custom protocol	"Creating a Custom Protocol" module

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/techsupport

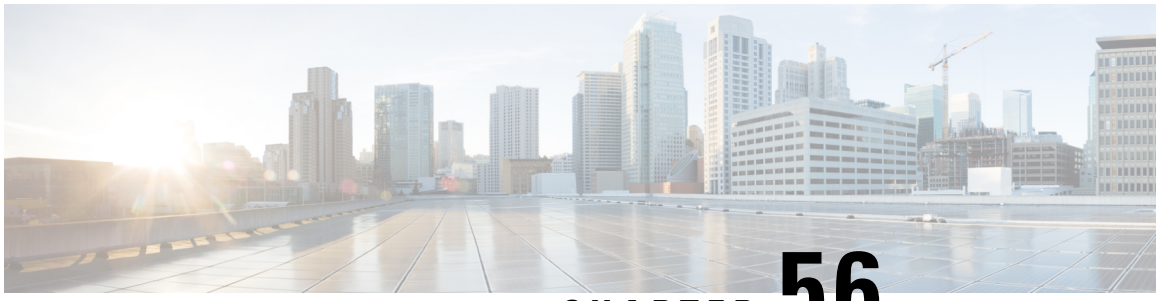
Feature Information for Reporting Extracted Fields Through Flexible NetFlow

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 68: Feature Information for Reporting Extracted Fields Through Flexible NetFlow

Feature Name	Releases	Feature Information
Reporting Extracted Fields Through Flexible NetFlow	Cisco IOS XE Release 3.7	The Reporting Extracted Fields Through Flexible NetFlow feature allows NBAR to send subapplication table fields to the collector through Flexible NetFlow. The following command was introduced or modified: **option (Flexible NetFlow)**.



CHAPTER 56

NBAR2 Custom Protocol

Network-Based Application Recognition (NBAR) supports the use of custom protocols to identify custom applications. Custom protocols support static port-based protocols and applications that NBAR does not support.

Prerequisites for Creating a Custom Protocol

Before creating a custom protocol, read the information in the "Classifying Network Traffic Using NBAR" module.

Information About Creating a Custom Protocol

NBAR and Custom Protocols

NBAR supports the use of custom protocols to identify custom applications. Custom protocols support static port-based protocols and applications that NBAR does not currently support.

Note For a list of NBAR-supported protocols, see the "Classifying Network Traffic Using NBAR" module.

With NBAR supporting the use of custom protocols, NBAR can map static TCP and UDP port numbers to the custom protocols.

Initially, NBAR included the following features related to custom protocols and applications:

- Custom protocols had to be named custom-xx, with xx being a number.

- Ten custom applications can be assigned using NBAR, and each custom application can have up to 16 TCP and 16 UDP ports each mapped to the individual custom protocol. The real-time statistics of each custom protocol can be monitored using Protocol Discovery.

NBAR includes the following characteristics related to user-defined custom protocols and applications:

- The ability to inspect the payload for certain matching string patterns at a specific offset.
- The ability to allow users to define the names of their custom protocol applications. The user-named protocol can then be used by Protocol Discovery, the Protocol Discovery MIB, the **match protocol** command, and the **ip nbar port-map** command as an NBAR-supported protocol.
- The ability of NBAR to inspect the custom protocols specified by traffic direction (that is, traffic heading toward a source or a destination rather than traffic in both directions).
- CLI support that allows a user configuring a custom application to specify a range of ports rather than specify each port individually.
- The **http**/**dns**/**ssl** keyword group that lets you add custom host and URL signatures.

Note Defining a user-defined custom protocol restarts the NBAR feature, whereas defining predefined custom protocol does not restart the NBAR feature.

MQC and NBAR Custom Protocols

NBAR recognizes and classifies network traffic by protocol or application. You can extend the set of protocols and applications that NBAR recognizes by creating a custom protocol. Custom protocols extend the capability of NBAR Protocol Discovery to classify and monitor additional static port applications and allow NBAR to classify nonsupported static port traffic. You define a custom protocol by using the keywords and arguments of the **ip nbar custom** command. However, after you define the custom protocol, you must create a traffic class and configure a traffic policy (policy map) to use the custom protocol when NBAR classifies traffic. To create traffic classes and configure traffic polices, use the functionality of the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC). The MQC is a command-line interface that allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach these traffic policies to interfaces. For more information about NBAR and the functionality of the MQC, see the "Configuring NBAR Using the MQC" module.

IP Address and Port-based Custom Protocol

IP address and port-based custom protocol includes supporting an IP subnet or a list of IP addresses with a specific TCP or UDP transport. This enables Network-Based Application Recognition (NBAR) to recognize traffic based on IP addresses and to associate an application ID to traffic from and to specified IP addresses. You define a custom protocol transport by using the keywords and arguments of the **ip nbar custom transport** command.

To support the IP address and port-based custom protocol option, the custom configuration mode (config-custom) is introduced with the **ip nbar custom transport** command. This mode supports options to specify a maximum of eight individual IP addresses, subnet IP addresses, and subnet mask length. You can also specify a list of eight ports or a start port range and an end port range.

Comparison of Custom NBAR Protocols: Based on a Single Network Protocol or Based on Multiple Network Protocols

Note In this description, the term "protocol" is used in two ways: as an NBAR protocol used for identifying traffic, and as a network protocol (HTTP, SSL, and so on).

NBAR provides:

- **Custom NBAR protocols based on single network protocol**

  Useful for identifying a single type of traffic (HTTP, SSL, and so on) according to a specified pattern.

  Syntax: ip nbar custom <protocol_name> <traffic_type> <criteria>

- **Custom NBAR protocols based on multiple network protocols** (called a "composite" custom NBAR protocol)

  Useful for identifying traffic using signatures for multiple network protocols. Currently, the composite method provides an option, "server-name" (value for <composite_option> in the CLI syntax) that identifies all HTTP, SSL, and DNS traffic associated with a specific server.

  Useful for identifying multiple types of traffic (HTTP, SSL, and so on) according to a specified pattern, using a single protocol.

  Syntax: ip nbar custom <protocol_name> composite <composite_option> <criteria>

Example Use Case: Custom NBAR Protocol Based on Multiple Network Protocols

- **Objective**: Identify all HTTP, SSL, and DNS traffic associated with the abc_example.com server.
- **Preferred method**: Use a composite custom NBAR protocol.
- **CLI**: `ip nbar custom abc_example_custom composite server-name *abc_example`

Limitations of Custom Protocols

The following limitations apply to custom protocols:

- NBAR supports a maximum of 120 custom protocols. All custom protocols are included in this maximum, including single-signature and composite protocols.
- Cannot define two custom protocols for the same target regular expression.

  For example, after configuring **`ip nbar custom 1abcd http url www.abcdef.com`**, cannot then configure:

  `ip nbar custom 2abcd http url www.abcdef.com`

  Attempting to do so results in an error.

- Maximum length for the regular expression that defines the custom protocol: 30 characters

How to Create a Custom Protocol

Defining a Custom NBAR Protocol Based on a Single Network Protocol

Custom protocols extend the capability of NBAR Protocol Discovery to classify and monitor additional static port applications and allow NBAR to classify non-supported static port traffic.

This procedure creates a custom NBAR protocol based on a single network protocol (HTTP, SSL, and so on).

Note NBAR supports a maximum of 120 custom protocols. All custom protocols are included in this maximum, including single-signature and composite protocols.

To define a custom protocol, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *protocol-name* [*offset* [*format value*]] [**variable** *field-name field-length*] [*source* | *destination*] [**tcp** | **udp**] [**range** *start end* | *port-number*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar custom** *protocol-name* [*offset* [*format value*]] [**variable** *field-name field-length*] [*source*	*destination*] [**tcp**	**udp**] [**range** *start end*	*port-number*] **Example:** `Router(config)# ip nbar custom app_sales1 5 ascii SALES source tcp 4567`	Extends the capability of NBAR Protocol Discovery to classify and monitor additional static port applications or allows NBAR to classify non-supported static port traffic. • Creates a custom NBAR protocol that identifies traffic based on a single network protocol. • Useful for identifying a single type of traffic (HTTP, SSL, and so on) according to a specified pattern. • Enter the custom protocol name and any other optional keywords and arguments.

	Command or Action	Purpose
Step 4	**end** **Example:** `Router(config)# end`	(Optional) Exits global configuration mode.

Examples

In the following example, the custom protocol LAYER4CUSTOM will look for TCP packets that have a destination or source port of 6700:

```
Device# configure terminal
Device(config)# ip nbar custom LAYER4CUSTOM  transport tcp id 14
Device(config-custom)# port 6700
```

To display other options besides port:

```
Device# configure terminal
Device(config)# ip nbar custom LAYER4CUSTOM  transport tcp id 14
Device(config-custom)# ?
Custom protocol commands:
  direction  Flow direction
  dscp       DSCP in IPv4 and IPv6 packets
  exit       Exit from custom configuration mode
  ip         ip address
  ipv6       ipv6 address
  no         Negate a command or set its defaults
  port       ports
```

Defining a Custom NBAR Protocol Based on Multiple Network Protocols

Custom protocols extend the capability of NBAR Protocol Discovery to classify and monitor additional static port applications and allow NBAR to classify non-supported static port traffic.

This procedure creates a custom NBAR protocol based on multiple network protocols.

Note In this description, the term "protocol" is used in two ways: as an NBAR protocol used for identifying traffic, and as a network protocol (HTTP, SSL, and so on).

Note NBAR supports a maximum of 120 custom protocols. All custom protocols are included in this maximum, including single-signature and composite protocols.

To define a composite-signature custom protocol, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *protocol-name* **composite server-name** *server-name*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar custom** *protocol-name* **composite server-name** *server-name* **Example:** `Router(config)# ip nbar custom abc_example_custom composite server-name *abc_example`	Extends the capability of NBAR Protocol Discovery to classify and monitor additional static port applications or allows NBAR to classify non-supported static port traffic. • Creates a custom NBAR protocol that identifies traffic using signatures for multiple network protocols. Currently, the only option for *composite-option* is **server-name**, which identifies all HTTP, SSL, and DNS traffic associated with a specific server. • Useful for identifying multiple types of traffic (HTTP, SSL, and so on) according to a specified pattern, using a single protocol. In the example, the objective is to identify all HTTP, SSL, and DNS traffic associated with the **abc_example.com** server.
Step 4	**end** **Example:** `Router(config)# end`	(Optional) Exits global configuration mode.

Configuring a Traffic Class to Use the Custom Protocol

Traffic classes can be used to organize packets into groups on the basis of a user-specified criterion. For example, traffic classes can be configured to match packets on the basis of the protocol type or application recognized by NBAR. In this case, the traffic class is configured to match on the basis of the custom protocol.

To configure a traffic class to use the custom protocol, perform the following steps.

Note The **match protocol**command is shown at Step 4. For the *protocol-name* argument, enter the protocol name used as the match criteria. For a custom protocol, use the protocol specified by the *name* argument of the **ip nbar custom**command. (See Step 3 of the Defining a Custom Protocol task.)

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** [**match-all** | **match-any**] *class-map-name*
4. **match protocol** *protocol-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** [**match-all**	**match-any**] *class-map-name* **Example:** `Router(config)# class-map cmap1`	Creates a class map to be used for matching packets to a specified class and enters class-map configuration mode. • Enter the name of the class map.
Step 4	**match protocol** *protocol-name* **Example:** `Router(config-cmap)# match protocol app_sales1`	Configures NBAR to match traffic on the basis of the specified protocol. • For the *protocol-name* argument, enter the protocol name used as the match criterion. For a custom protocol, use the protocol specified by the *name* argument of the **ip nbar custom**command. (See Step 3 of the "Defining a Custom Protocol" task.)
Step 5	**end** **Example:** `Router(config-cmap)# end`	(Optional) Exits class-map configuration mode.

Examples

In the following example, the **variable** keyword is used while creating a custom protocol, and class maps are configured to classify different values within the variable field into different traffic classes. Specifically, in the example below, variable scid values 0x15, 0x21, and 0x27 will be classified into class map active-craft, while scid values 0x11, 0x22, and 0x25 will be classified into class map passive-craft.

```
Router(config)#
 ip nbar custom ftdd 23 variable scid 1 tcp range 5001 5005

Router(config)#
class-map active-craft
Router(config-cmap)# match protocol ftdd scid 0x15
Router(config-cmap)# match protocol ftdd scid 0x21
Router(config-cmap)# match protocol ftdd scid 0x27

Router(config)#
class-map passive-craft
Router(config-cmap)# match protocol ftdd scid 0x11
Router(config-cmap)# match protocol ftdd scid 0x22
Router(config-cmap)# match protocol ftdd scid 0x25
```

Configuring a Traffic Policy

Traffic that matches a user-specified criterion can be organized into specific classes. The traffic in those classes can, in turn, receive specific QoS treatment when that class is included in a policy map.

To configure a traffic policy, perform the following steps.

Note The **bandwidth** command is shown at Step 5. The **bandwidth** command configures the QoS feature class-based weighted fair queuing (CBWFQ). CBWFQ is just an example of a QoS feature that can be configured. Use the appropriate command for the QoS feature that you want to use.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **bandwidth** {*bandwidth-kbps*| **remaining percent** *percentage*| **percent** *percentage*}
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy1`	Creates or modifies a policy map that can be attached to one or more interfaces and enters policy-map configuration mode. • Enter the name of the policy map.
Step 4	**class** {*class-name* | **class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the name of the class whose policy you want to create or change and enters policy-map class configuration mode. • Enter the specific class name or enter the **class-default**keyword.
Step 5	**bandwidth** {*bandwidth-kbps*| **remaining percent** *percentage*| **percent** *percentage*} **Example:** `Router(config-pmap-c)# bandwidth percent 50`	(Optional) Specifies or modifies the bandwidth allocated for a class belonging to a policy map. • Enter the amount of bandwidth as a number of kbps, a relative percentage of bandwidth, or an absolute amount of bandwidth. **Note** The **bandwidth** command configures the QoS feature class-based weighted fair queuing (CBWFQ). CBWFQ is just an example of a QoS feature that can be configured. Use the appropriate command for the QoS feature that you want to use.
Step 6	**end** **Example:** `Router(config-pmap-c)# end`	(Optional) Exits policy-map class configuration mode.

Attaching the Traffic Policy to an Interface

After a traffic policy (policy map) is created, the next step is to attach the policy map to an interface. Policy maps can be attached to either the input or output direction of the interface.

Note Depending on the needs of your network, you may need to attach the policy map to a subinterface, an ATM PVC, a Frame Relay DLCI, or other type of interface.

To attach the traffic policy to an interface, perform the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [*name-tag*]
4. **pvc** [*name*] *vpi* / *vci* [**ilmi**| **qsaal**| **smds**| **l2transport**]
5. **exit**
6. **service-policy** {**input** | **output**} *policy-map-name*
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [*name-tag*] **Example:** `Router(config)# interface ethernet 2/4`	Configures an interface type and enters interface configuration mode. • Enter the interface type and the interface number.
Step 4	**pvc** [*name*] *vpi* / *vci* [**ilmi**| **qsaal**| **smds**| **l2transport**] **Example:** `Router(config-if)# pvc cisco 0/16`	(Optional) Creates or assigns a name to an ATM permanent virtual circuit (PVC), specifies the encapsulation type on an ATM PVC, and enters ATM virtual circuit configuration mode. • Enter the PVC name, the ATM network virtual path identifier, and the network virtual channel identifier. **Note** This step is required only if you are attaching the policy map to an ATM PVC. If you are not attaching the policy map to an ATM PVC, advance to Step 6.
Step 5	**exit** **Example:** `Router(config-atm-vc)# exit`	(Optional) Returns to interface configuration mode. **Note** This step is required only if you are attaching the policy map to an ATM PVC and you completed Step 4. If you are not attaching the policy map to an ATM PVC, advance to Step 6.

	Command or Action	Purpose
Step 6	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# service-policy input policy1`	Attaches a policy map to an input or output interface. • Enter the name of the policy map. **Note** Policy maps can be configured on ingress or egress routers. They can also be attached in the input or output direction of an interface. The direction (input or output) and the router (ingress or egress) to which the policy map should be attached vary according to your network configuration. When using the **service-policy** command to attach the policy map to an interface, be sure to choose the router and the interface direction that are appropriate for your network configuration.
Step 7	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Displaying Custom Protocol Information

After you create a custom protocol and match traffic on the basis of that custom protocol, you can use the **show ip nbar port-map** command to display information about that custom protocol.

To display custom protocol information, complete the following steps.

SUMMARY STEPS

1. **enable**
2. **show ip nbar port-map** [*protocol-name*]
3. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show ip nbar port-map** [*protocol-name*] **Example:** `Router# show ip nbar port-map`	Displays the current protocol-to-port mappings in use by NBAR. • (Optional) Enter a specific protocol name.

	Command or Action	Purpose
Step 3	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Configuring IP Address and Port-based Custom Protocol

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *name* **transport {tcp | udp} {id** *id* **} {ip address** *ip-address* **| subnet** *subnet-ip subnet-mask***}| ipv6 address {***ipv6-address* **| subnet** *subnet-ipv6 ipv6-prefix***} | port {***port-number* **| range** *start-range end-range***} | direction {any | destination | source}**
4. **ip nbar custom** *name* **transport {tcp | udp} {id** *id***}**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

Step 3	**ip nbar custom** *name* **transport {tcp | udp} {id** *id* **} {ip address** *ip-address* **| subnet** *subnet-ip subnet-mask***}| ipv6 address {***ipv6-address* **| subnet** *subnet-ipv6 ipv6-prefix***} | port {***port-number* **| range** *start-range end-range***} | direction {any | destination | source}**
	Example:
	Specifies the IP address.
	`Device(config)# ip nbar custom mycustomprotocol transport tcp id 100`
	`Device(config-custom)# ip address 10.2.1.1`
	Example:
	Specifies the subnet IP and a subnet mask of 0.
		Configures the custom protocol, with options to specify IP address, subnet, port, direction, and so on. In the examples given, the command is executed on multiple lines, using the custom configuration mode, rather than the single-line format.

	Command or Action	Purpose
	`Device(config)# ip nbar custom mycustomprotocol transport tcp` `Device(config-custom)# ip subnet 255.255.255.255 0`	

Step 4	**ip nbar custom** *name* **transport {tcp | udp} {id** *id***}**
	Example:
	`Device(config)# ip nbar custom mycustom transport tcp id 100`
	`Device(config-custom)#`
		Specifies TCP or UDP as the transport protocol and enters custom configuration mode.

Step 5	**end** **Example:** `Device(config-custom)# end`	(Optional) Exits custom configuration mode.

Configuration Examples for Creating a Custom Protocol

Example Creating a Custom Protocol

In the following example, the custom protocol called app_sales1 identifies TCP packets that have a source port of 4567 and that contain the term SALES in the first payload packet:

```
Router> enable

Router# configure terminal

Router(config)# ip nbar custom app_sales1 5 ascii SALES source tcp 4567

Router(config)# end
```

Example Configuring a Traffic Class to Use the Custom Protocol

In the following example, a class called cmap1 has been configured. All traffic that matches the custom app_sales1 protocol will be placed in the cmap1 class.

```
Router> enable

Router# configure terminal

Router(config)# class-map cmap1

Router(config-cmap)# match protocol app_sales1
```

```
Router(config-cmap)# end
```

Example Configuring a Traffic Policy

In the following example, a traffic policy (policy map) called policy1 has been configured. Policy1 contains a class called class1, within which CBWFQ has been enabled.

```
Router> enable

Router# configure terminal

Router(config)# policy-map policy1

Router(config-pmap)# class class1

Router(config-pmap-c)# bandwidth percent 50

Router(config-pmap-c)# end
```

Note In the above example, the **bandwidth** command is used to enable Class-Based Weighted Fair Queuing (CBWFQ). CBWFQ is only an example of one QoS feature that can be applied in a traffic policy (policy map). Use the appropriate command for the QoS feature that you want to use.

Example Attaching the Traffic Policy to an Interface

In the following example, the traffic policy (policy map) called policy1 has been attached to ethernet interface 2/4 in the input direction of the interface.

```
Router> enable

Router# configure terminal

Router(config)# interface ethernet 2/4

Router(config-if)# service-policy input policy1

Router(config-if)# end
```

Example Displaying Custom Protocol Information

The following is sample output of the **show ip nbar port-map** command. This command displays the current protocol-to-port mappings in use by NBAR. Use the display to verify that these mappings are correct.

```
Router# show ip nbar port-map
port-map bgp      udp 179
port-map bgp      tcp 179
port-map cuseeme  udp 7648 7649
port-map cuseeme  tcp 7648 7649
port-map dhcp     udp 67 68
port-map dhcp     tcp 67 68
```

If the **ip nbar port-map** command has been used, the **show ip nbar port-map** command displays the ports assigned to the protocol.

If the **no ip nbar port-map**command has been used, the **show ip nbar port-map** command displays the default ports. To limit the display to a specific protocol, use the *protocol-name* argument of the **show ip nbar port-map** command.

Example: Configuring IP Address and Port-based Custom Protocol

The following example shows how to enter custom configuration mode from global configuration mode and configure a subnet IP address and its mask length:

```
Device(config)# ip nbar custom mycustomprotocol transport tcp id 100
Device(config-custom)# ip subnet 10.1.2.3 22
```

The following example configures two custom protocols, one for TCP and one for UDP traffic. In each, the subnet, subnet mask, DSCP value, and direction are configured.

```
Device(config)# ip nbar custom mycustomprotocol_tcp transport tcp
Device(config-custom)# ip subnet 255.255.255.255 0
Device(config-custom)# dscp 18
Device(config-custom)# direction any
Device(config-custom)# end
Device(config)# ip nbar custom mycustomprotocol_udp transport udp
Device(config-custom)# ip subnet 255.255.255.255 0
Device(config-custom)# dscp 18
Device(config-custom)# direction any
```

Additional References

The following sections provide references related to creating a custom protocol.

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
MQC, traffic policies (policy maps), and traffic classes	"Applying QoS Features Using the MQC" module

Related Topic	Document Title
Concepts and information about NBAR	"Classifying Network Traffic Using NBAR" module
Information about enabling Protocol Discovery	"Enabling Protocol Discovery" module
Configuring NBAR using the MQC	"Configuring NBAR Using the MQC" module
Adding application recognition modules (also known as PDLMs)	"Adding Application Recognition Modules" module

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for NBAR2 Custom Protocol

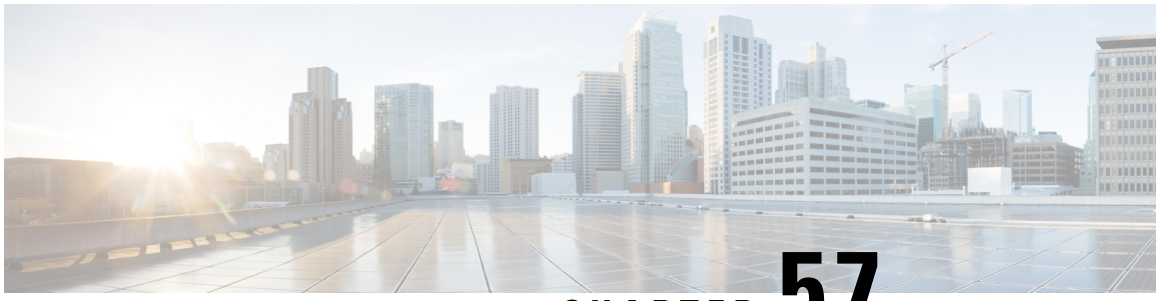
The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 69: Feature Information for NBAR2 Custom Protocol

Feature Name	Releases	Feature Information
NBAR2 Custom Protocol	Cisco IOS XE Release 3.8S	This feature was introduced on Cisco ASR 1000 series Aggregation Services Routers. The following command was introduced or modified: **ip nbar custom**

Feature Name	Releases	Feature Information
NBAR2 Custom Protocol Enhancements Ph II	Cisco IOS XE Release 3.12S	The NBAR2 Custom Protocol Enhancements Phase II feature enables supporting an IP subnet or a list of IP addresses with a specific TCP or UDP transport. The following command was introduced or modified: **ip nbar custom**



CHAPTER 57

NBAR2 Protocol Pack Hitless Upgrade

The NBAR2 Protocol Pack Hitless Upgrade feature enables users to seamlessly upgrade a Network-Based Application Recognition (NBAR) protocol pack or change the NBAR configurations without impacting any of the current classification configurations on a device.

Restrictions for NBAR2 Protocol Pack Hitless Upgrade

Additional memory is required to support the NBAR2 Protocol Pack Hitless Upgrade feature because it holds together two configurations until the previous configuration is aged.

Information About NBAR2 Protocol Pack Hitless Upgrade

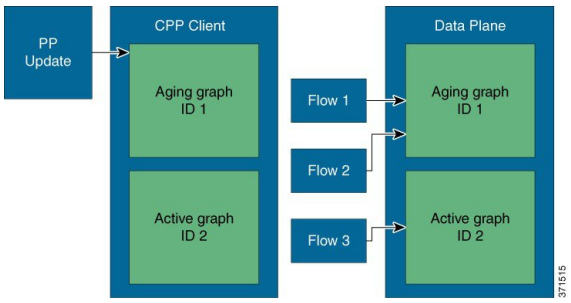
Overview of NBAR2 PP Hitless Upgrade

Hitless Upgrade is the method to upgrade the NBAR2 Protocol Pack (PP) components on an NBAR engine without incurring any service downtime. In earlier Cisco IOS software releases, NBAR could hold only a single configuration graph on the control plane client that is transferred to the data path. From Cisco IOS XE Release 3.12S onward, NBAR can hold several configurations graphs at a single time. When a new configuration change occurs, a new configuration graph is created on the control plane client. The new graph is downloaded to the data plane, and all new flows are directed to the new graph.

If a packet arrives from a flow that was being classified, the packet is directed to the correct configuration graph (the one that was active when the flow was created).

The following illustration displays the NBAR system state after a configuration or protocol pack update:

Figure 93: Aging a Graph



In the illustration above, when a new graph is created, the old graph is moved to the aging state. In an aged state, only flows that are associated with the graph are referenced with the graph. If a flow is not classified until aging time, it is reported as unknown by NBAR.

Note Due to memory limitations, it is important to limit the number of parallel existing graphs and aging graphs in the NBAR system. Currently, all platforms can hold a maximum two configurations at a given time.

Use the **show platform software nbar statistics** command to view the status of NBAR.

Benefits of NBAR2 Protocol Pack Hitless Upgrade

NBAR2 Protocol Pack Hitless Upgrade provides the following benefits:

- No loss of information for classified flows during a protocol upgrade
- No impact on new flows
- No impact on in-progress flows

Additional References for NBAR2 Protocol Pack Hitless Upgrade

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
NBAR Protocol Pack	*QoS: NBAR Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

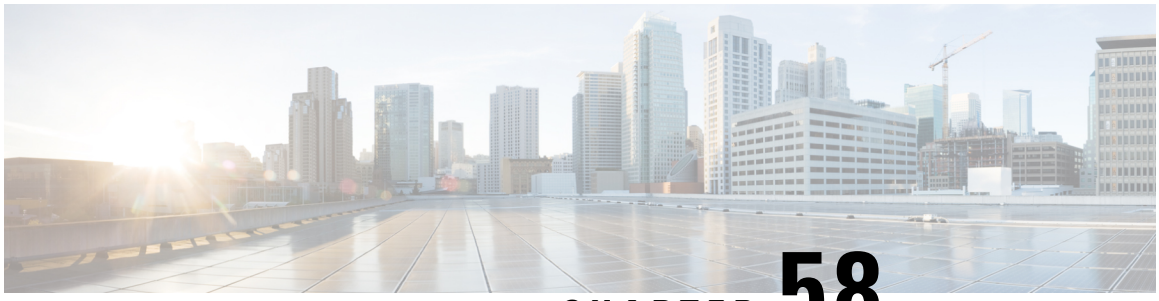
Feature Information for NBAR2 Protocol Pack Hitless Upgrade

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 70: Feature Information for NBAR2 Protocol Pack Hitless Upgrade

Feature Name	Releases	Feature Information
NBAR2 Protocol Pack Hitless Upgrade	Cisco IOS XE Release 3.12S	The NBAR2 Protocol Pack Hitless Upgrade feature enables seamless upgrade of a NBAR protocol pack or NBAR configurations without impacting any of the current classification configurations on a device. In Cisco IOS XE Release 3.12S, support was added for the Cisco ASR 1000 Series Routers.



CHAPTER 58

NBAR Web-based Custom Protocols

The NBAR Web-based Custom Protocols feature provides the mechanism to define custom protocols to match based on HTTP URL and/or host name.

Restrictions for NBAR Web-based Custom Protocols

The HTTP URL and the Host name defined for custom protocol match should be unique. The length of the protocol name should be at least 4 characters long and the prefix of the protocol name should be different from the prefixes of any other protocol name.

Information About NBAR Web-based Custom Protocols

Overview of NBAR Web-based Custom Protocols

The NBAR Web-based Custom Protocols feature provides the mechanism to define custom protocols to match the traffic based on HTTP URL and/or host name.

All 120 custom protocols can be defined to match based on HTTP URL and/or host name. While matching web-based custom protocols, the custom protocol that has both HTTP URL and the host name defined has the highest priority, followed by HTTP URL as the second priority, and then followed by Host name as the last priority. Matching a web-based sub-protocol has higher priority than matching any type of web-based custom protocol, for example the **match protocol** *http url http-url* command has a higher priority than a custom priority with the same URL configuration.

How to Define NBAR Web-based Custom Protocols Match

Defining a Web-based Custom Protocol Match

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *custom-protocol-name* **http** {**host** *host-name* | **url** *http-url* [**host** *host-name*]} [**id** *selector-id*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar custom** *custom-protocol-name* **http** {**host** *host-name* | **url** *http-url* [**host** *host-name*]} [**id** *selector-id*] **Example:** `Router(config)# ip nbar custom app_sales1 http url www.example.com`	Defines web-based custom protocol match. • Enter the custom protocol name and any other optional keywords and arguments. **Note** To add a custom protocol, use the **ip nbar custom** command. To enable the protolcol, use the **match protocol** command or **ip nbar protocol discovery** command.
Step 4	**end** **Example:** `Router(config)# end`	(Optional) Exits global configuration mode.

Configuration Examples for NBAR Web-based Custom Protocols

Examples: Defining Web-based Custom Protocol Match

The following example displays how to match a custom protocol based on http url:

```
Router> enable
Router# configure terminal
Router(config)# ip nbar custom app_sales1 http url www.example.com
```

The following example displays how to match a custom protocol that contains the string 'example' as a part of host name:

```
Router> enable
Router# configure terminal
Router(config)# ip nbar custom app_sales1 http host *example*
```

Additional References for NBAR Web-based Custom Protocols

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*
Custom Protocols	*Creating a Custom Protocol* module

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

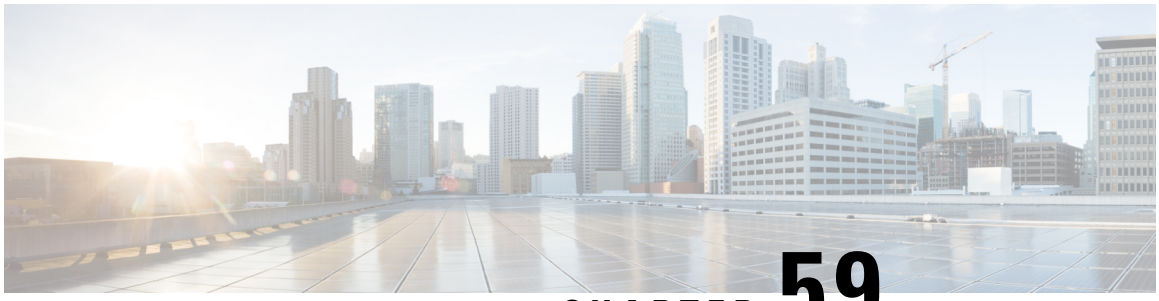
Feature Information for NBAR Web-based Custom Protocols

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 71: Feature Information for NBAR Web-based Custom Protocols

Feature Name	Releases	Feature Information
NBAR Web-based Custom Protocols Scalability	Cisco IOS XE Release 3.13S	The NBAR Web-based Custom Protocols Scalability feature enables defining custom protocols match based on http host name and/or url. The following command was introduced or modified: **ip nbar custom**.



CHAPTER 59

NBAR2 HTTP-Based Visibility Dashboard

The NBAR2 HTTP-based Visibility Dashboard provides a web interface displaying network traffic data and related information. The information is presented in an intuitive, interactive graphical format.

Overview of NBAR2 HTTP-based Visibility Dashboard

The NBAR2 HTTP-based Visibility Dashboard provides a graphical display of network information, such as network traffic details and bandwidth utilization. The Visibility Dashboard includes interactive charts and a graph of bandwidth usage.

The basic workflow for using the Visibility Dashboard is:

1. Using the procedure described in Configuring NBAR2 HTTP-Based Visibility Dashboard, on page 799, configure the router to provide information for the Visibility Dashboard. This includes:
   - Enabling an HTTP server.
   - Setting up the router service that collects and stores traffic data.
   - Specifying an interface to monitor.
   - Enabling protocol discovery.
2. In a browser, connect to the Visibility Dashboard web interface to display traffic information for the monitored interface(s), using the router IP address or hostname, and appending **/flash/nbar2/home.html**.

   Example: **10.56.1.1/flash/nbar2/home.html**

   See Accessing the Visibility Dashboard, on page 801.
3. The HTTP server that operates with the Visibility Dashboard requires HTTP command access to the router to collect traffic data to present in the dashboard. Specifically, the HTTP server executes **show ip nbar** CLI commands on the router to collect the data. Access is provided to the Visibility Dashboard HTTP server by one of the following methods:

- Providing "privilege 15" general access to the router.

  Use the **ip http authentication enable** CLI command on the router to set a password. When logging into the Visibility Dashboard web interface, use the specified password. No username is required.

- Setting a local username and password for the router.

  Use the **ip http authentication local** command to set a local username/password providing HTTP command access. When logging into the Visibility Dashboard web interface, enter the specified username and password.

  Example configuration:

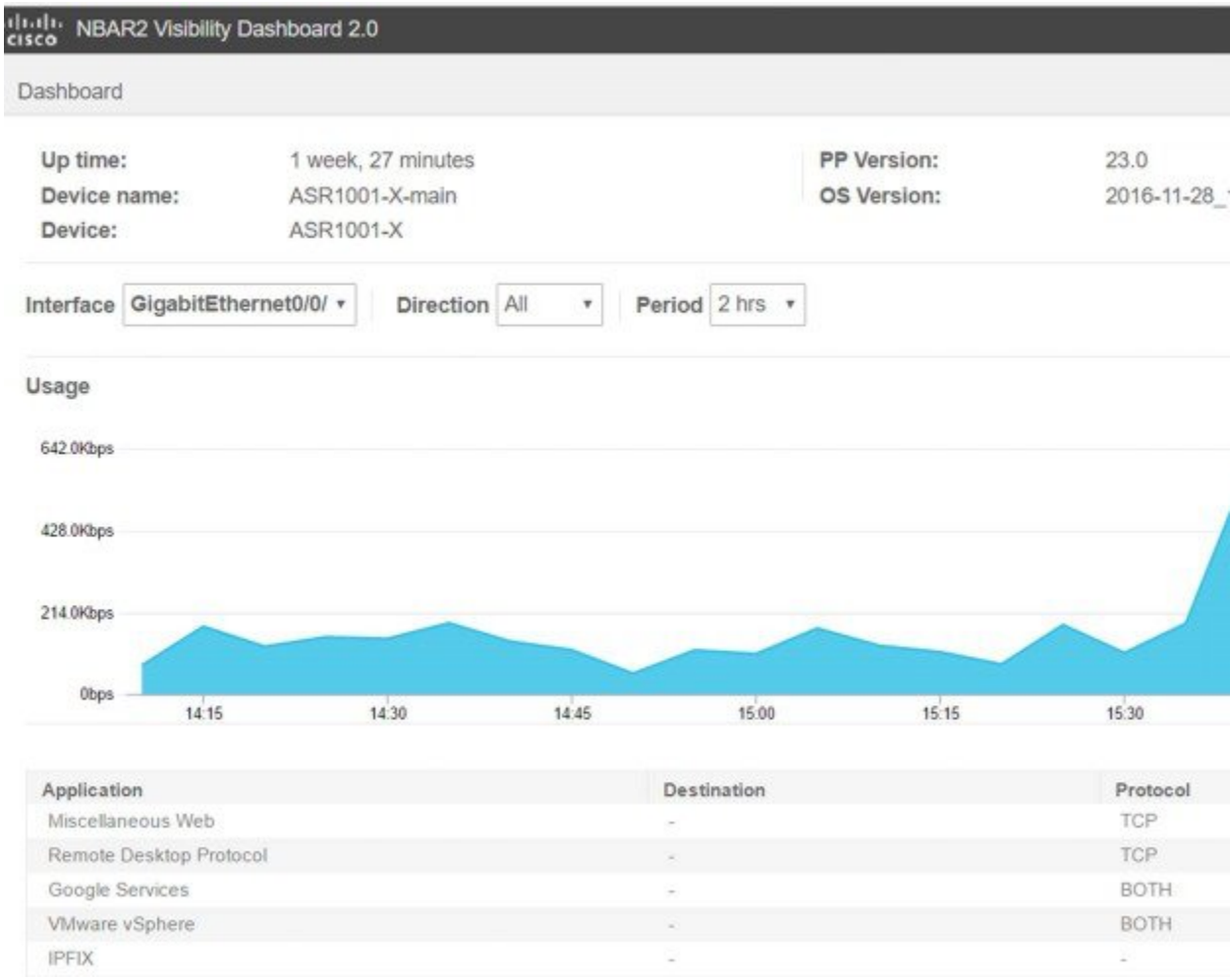
```
Device(config)#ip http authentication enable
Device(config)#ip http authentication local
Device(config)#username cisco
Device(config)#password n449rbpsvq
```

- Using an Authentication, Authorization, and Accounting (AAA) server.

  The AAA server manages accounts, including username/password credentials. When logging into the Visibility Dashboard web interface, enter the username and password for an account managed by the AAA server.

  Note: The account must include authorization to execute **show ip nbar** commands on the router. If the account does not provide this authorization, a user could log in and pass authentication, but no traffic data would be available from the router. The Visibility Dashboard would appear in the browser, but showing no information.

Figure 94: Visibility Dashboard



Configuring NBAR2 HTTP-Based Visibility Dashboard

Before you begin

The HTTP-based Visibility Dashboard uses the Protocol Discovery feature.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip http server**
4. **ip nbar http-services**
5. **interface gigabitethernet** *interface*
6. **ip nbar protocol-discovery**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device>enable`	Enables privileged EXEC mode. Enter a password if prompted.
Step 2	**configure terminal** **Example:** `Device#configure terminal`	Enters global configuration mode.
Step 3	**ip http server** **Example:** `Device(config)#ip http server`	Enables an HTTP server. The server operates with the Visibility Dashboard, providing the data collected by the router.
Step 4	**ip nbar http-services** **Example:** `Device(config)#ip nbar http-services`	Configures the HTTP services to collect traffic data and store it in a database.
Step 5	**interface gigabitethernet** *interface* **Example:** `Device(config)#interface gigabitethernet 0/0/2`	Specifies an interface to monitor.
Step 6	**ip nbar protocol-discovery** **Example:** `Device(config)#ip nbar protocol-discovery`	

Example: NBAR2 HTTP-Based Visibility Dashboard

Example: Enabling NBAR2 HTTP-Services

```
Device> enable
Device# configure terminal
Device(config)# ip nbar http-services
Device(config)# end
```

Accessing the Visibility Dashboard

In a browser with access to the router, connect to the Visibility Dashboard web interface to display traffic information for the monitored interface(s), using the router IP address or hostname, and appending **/flash/nbar2/home.html**. This string is shown in the CLI help for **ip nbar http-services** by typing: **ip nbar ?**

Options:

- **http://<router-IP-address>/flash/nbar2/home.html**
- **http://<router-hostname>/flash/nbar2/home.html**

Example:

```
http://10.56.1.1/flash/nbar2/home.html
```

Additional References for NBAR2 HTTP-Based Visibility Dashboard

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

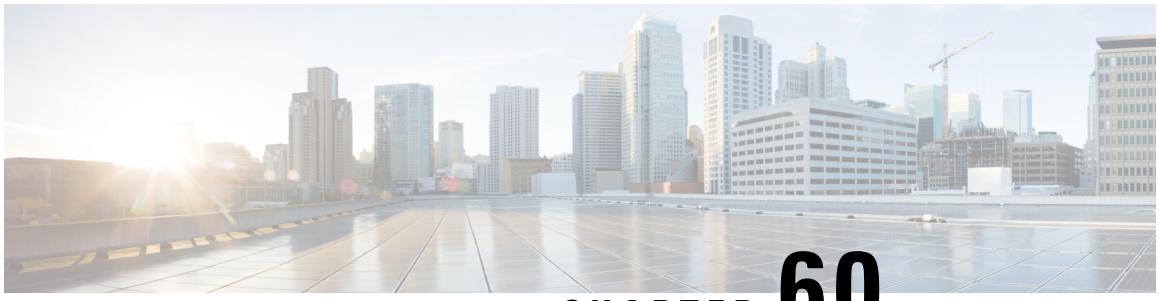
Feature Information for NBAR2 HTTP-Based Visibility Dashboard

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 72: Feature Information for NBAR2 HTTP-Based Visibility Dashboard

Feature Name	Releases	Feature Information
NBAR2 HTTP-Based Visibility Dashboard	Cisco IOS XE Release 3.16S	The NBAR2 HTTP-based Visibility Dashboard provides a web interface displaying network traffic data and related information. The information is presented in an intuitive, interactive graphical format. The following command was modified or introduced by this feature: **ip nbar http-services**



CHAPTER 60

NBAR Coarse-Grain Classification

Information About NBAR Coarse-Grain Classification

Overview of NBAR Coarse-Grain Classification

NBAR provides two levels of application recognition-coarse-grain and fine-grain. By default NBAR operates in the coarse-grain mode.

By minimizing deep packet inspection, coarse-grain mode offers a performance advantage and reduces memory resource demands. This mode is useful in scenarios where the full power of fine-grain classification is not required.

Simplified Classification

Coarse-grain mode employs a simplified mode of classification, minimizing deep packet inspection. NBAR caches classification decisions made for earlier packets, then classifies later packets from the same server similarly.

Limitations of Coarse-Grain Mode

Coarse-grain mode has the following limitations in metric reporting detail:

- Granularity: Caching may result in some reduction in the granularity. For example, NBAR might classify some traffic as **ms-office-365** instead of as the more specific **ms-office-web-apps**.
- Evasive applications: Classification of evasive applications, such as BitTorrent, eMule, and Skype, may be less effective than in fine-grain mode. Consequently, blocking or throttling may not work as well for these applications.

Comparison of Fine-grain and Coarse-grain Modes

Coarse-grain mode has the following limitations in metric reporting detail:

	Fine-Grain Mode	Coarse-Grain Mode
Classification	Full-power of deep packet inspection	Simplified classification Some classification according to similar earlier packets.
Performance	Slower	Faster
Memory Resources	Higher memory demands	Lower memory demands
Sub-classification	Full supported	Partial support
Field Extraction	Full supported	Partial support
Ideal usage	Per-packet policy Example: class-map that looks for specific url	When there is no requirement for specific per-packet operations.

Additional References for NBAR Coarse-Grain Classification

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*
AVC information	*AVC User Guide*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	https://www.cisco.com/c/en/us/support/index.html

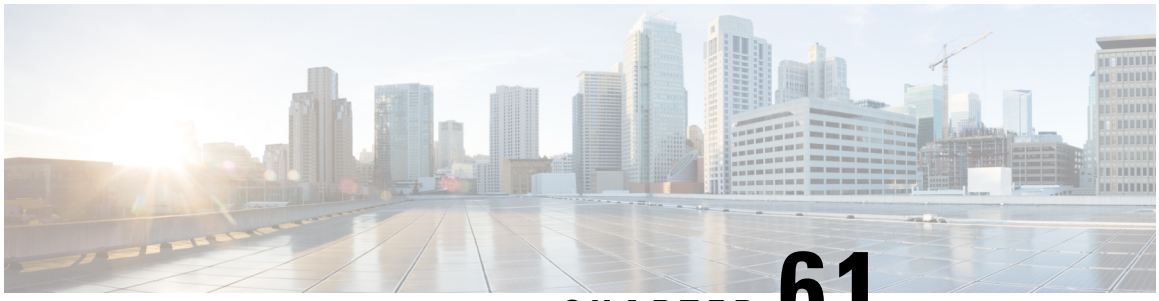
Feature Information for NBAR Coarse-Grain Classification

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 73: Feature Information for NBAR Coarse-Grain Classification

Feature Name	Releases	Feature Information
NBAR Coarse-Grain Classification		Network Based Application Recognition (NBAR) provides two levels of application recognition—coarse-grain and fine-grain. By default NBAR operates in the fine-grain mode, offering NBAR's full application recognition capabilities. By minimizing deep packet inspection, coarse-grain mode offers a performance advantage and reduces memory resource demands. The following command was introduced or modified: **ip nbar classification granularity** and **show ip nbar classification granularity**.
NBAR Coarse-Grain Classification	Cisco IOS XE Release 3.16S Cisco IOS XE 16.x releases	Default mode changed to coarse-grain.



CHAPTER 61

SSL Custom Application

SSL Custom Application feature enables users to customize applications that run on any protocol over Secure Socket Layer (SSL), including HTTP over Secure Socket Layer (HTTPS), using the server name, if it exists in the Client Hello extensions, or the common name from the certificate that the server sends to the client.

Information About SSL Custom Application

Overview of SSL Custom Application

SSL Custom Application feature enables users to customize applications that run on any protocol over Secure Socket Layer (SSL), including HTTP over Secure Socket Layer (HTTPS), using the server name, if it exists in the Client Hello extensions, or the common name from the certificate that the server sends to the client.

HTTP over Secure Socket Layer (HTTPS) is a communication protocol for secure communication. HTTPS is the result of layering HTTP on SSL protocol.

In SSL sub-classification, the rule that ends later in the packet will match. For example, consider the server name ‘finance.example.com’, if there is a rule for ‘finance’ and another rule for example.com, then the rule for ‘example.com’ will match.

SSL Unique Name Sub-Classification

The SSL unique-name parameter is used to match SSL sessions of servers that are not known globally, or are not yet supported by NBAR. The unique-name matches the server name indication (SNI) field in the client request, if the SNI field exists, or it matches the common name (CN) field in the first certificate of the server's response.

The feature also supports cases of SSL sessions that use session-id than the SSL sessions that use handshake.

The server name is available as part of a HTTPS URL itself. For example, in the URL https://www.facebook.com, the server name is www.facebook.com. However, the certificate is found in the browser. The user can observe the certificate information by clicking on the HTTPS icon.

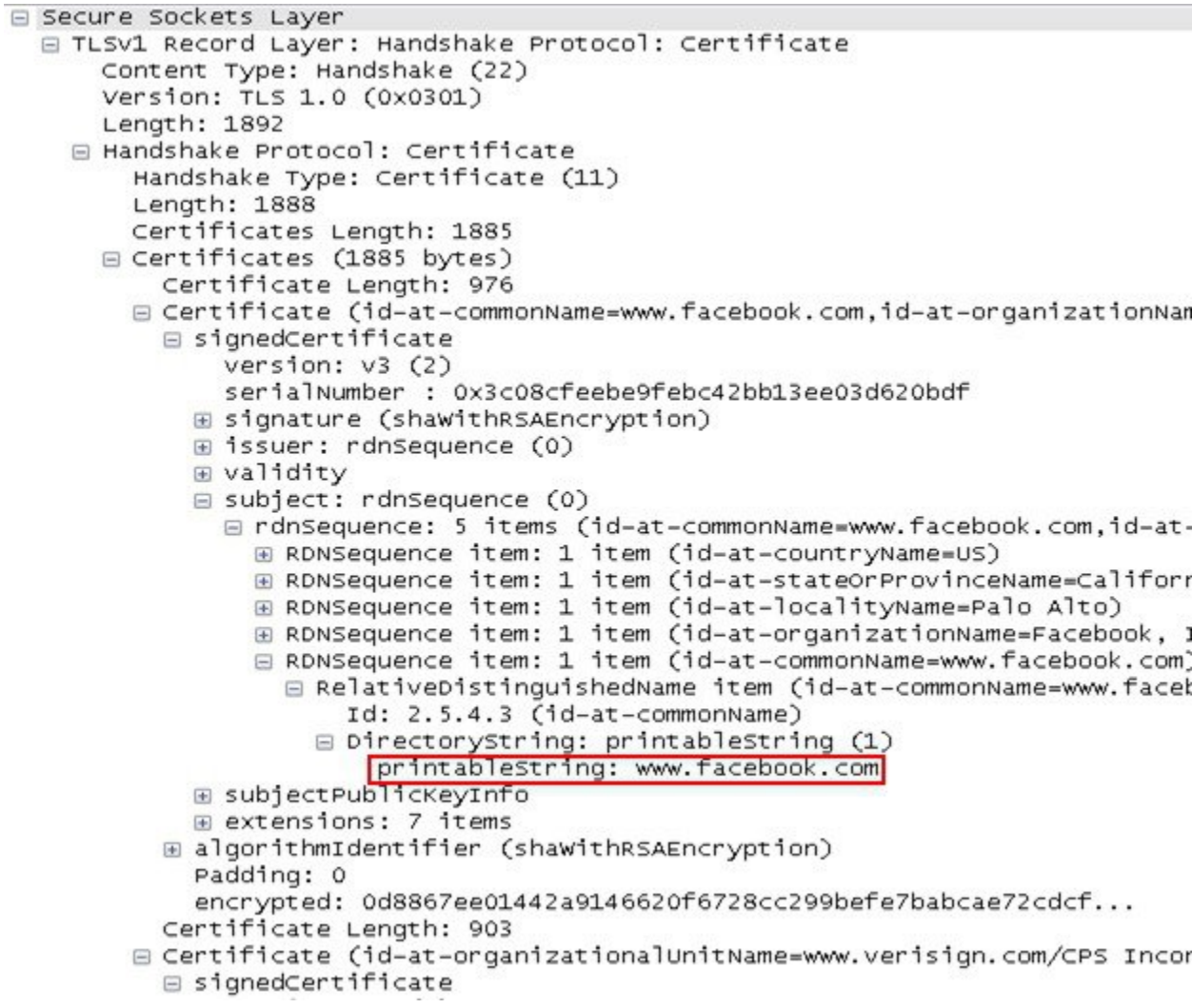
The following two figures display the location of the server name and common name as it is visible to the user using Wireshark tool.

The figure below highlights the location of the SNI field:

Figure 95: Server Name Indication Field

The figure below highlights the location of the CN field:

Figure 96: Common Name Field



How to Configure SSL Custom Application

Configuring SSL Custom Application

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *custom-protocol-name* **ssl unique-name** *regex* **id** *selector-id*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar custom** *custom-protocol-name* **ssl unique-name** *regex* **id** *selector-id* **Example:** `Device (config)# ip nbar custom name ssl unique-name www.example.com id 11`	Defines the SSL-based custom protocol match and provides a hostname in the form of a regular expression. **Note** The hostname that is configured in this command is found either in the Server Name Indication (SNI) field in the Client Hello extensions or in the Common Name (CN) field in the digital certificate that the server sends to the client.
Step 4	**end** **Example:** `Router(config)# end`	(Optional) Exits global configuration mode.

Configuration Examples for the SSL Custom Application

Example: SSL Custom Applications

The following example displays how to configure SSL Custom Application. The hostname that is configured in this command is found either in the Server Name Indication (SNI) field in the Client Hello extensions or in the Common Name (CN) field in the digital certificate that the server sends to the client.

```
Device> enable
Device# configuration terminal
Device(config)# ip nbar custom name ssl unique-name  www.example.com id  11
Device(config)# exit
```

Additional References for SSL Custom Application

Related Documents for SSL Custom Application

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*
SSL Sub-classification	NBAR Protocol Pack module

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

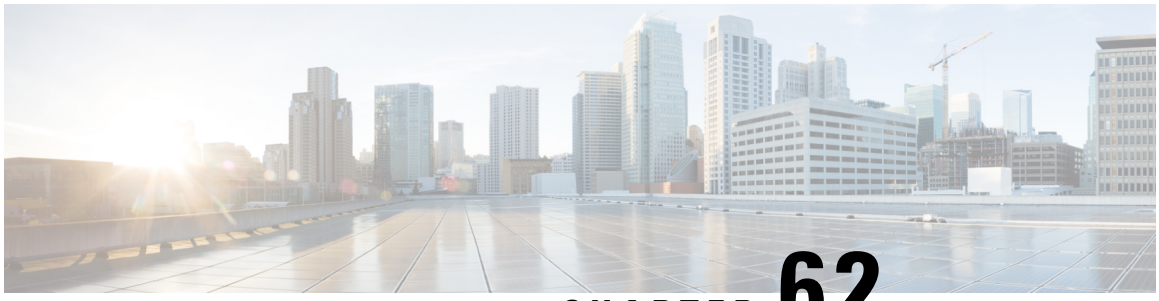
Feature Information for SSL Custom Application

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 74: Feature Information for SSL Custom Application

Feature Name	Releases	Feature Information
SSL Custom Application	Cisco IOS XE Release 3.15S	SSL Custom Application feature enables users to customize applications that run on any protocol over Secure Socket Layer (SSL), including HTTP over Secure Socket Layer (HTTPS), using the server name, if it exists in the Client Hello extensions, or the common name from the certificate that the server sends to the client. The following command was introduced or modified: **ip nbar custom**.



CHAPTER 62

Fine-Grain NBAR for Select Applications

NBAR provides two levels of application recognition: coarse-grain and fine-grain modes. Coarse-grain mode optimizes performance. Fine-grain mode provides NBAR's full application recognition capabilities, but with a higher performance cost. By default, NBAR operates in coarse-grain mode.

Feature Information

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 75: Feature Information for NBAR Fine-Grain Application Recognition Mode

Feature Name	Releases	Feature Information
Fine-grain application recognition mode	Cisco IOS XE Release 3.15S	By default NBAR operates in the fine-grain mode, offering NBAR's full application recognition capabilities. Used when per-packet reporting is required, fine-grain mode offers a troubleshooting advantage. Cisco recommends using fine-grain mode only when detailed Layer 7 metrics is required to be extracted by NBAR for critical applications. The fine-grain NBAR for Selective Applications feature enables a customer to dynamically monitor critical applications including collection of detailed Layer 7 metrics. The feature helps troubleshoot slowness in a particular application while the rest of the applications are running in coarse-grain mode and thus preventing any impact on the performance of the system. The following command was introduced or modified: **ip nbar custom**.
Fine-grain application recognition mode	Cisco IOS XE Release 3.16S Cisco IOS XE 16.x releases	Default mode changed to coarse-grain.

Fine-Grain NBAR for Selective Applications

Overview

NBAR provides two levels of application recognition: coarse-grain and fine-grain modes. Coarse-grain mode optimizes performance. Fine-grain mode provides NBAR's full application recognition capabilities, but with a higher performance cost.

By default, NBAR operates in coarse-grain mode. NBAR automatically changes to fine-grain mode when required, based on the configuration and traffic patterns. Typically, it is not necessary to change NBAR's automatic behavior, but you can configure fine-grain mode manually, using the procedure described below.

Forcing fine-grain mode for specific applications may be useful for monitoring a subset of applications, without adversely affecting performance, while other applications continue in coarse-grain mode.

How to Configure Fine-Grain NBAR for Specific Applications

To override NBAR's automatic behavior and force fine-grain mode, use the following procedure. The procedure enables specifying applications individually by name or specifying applications that match a specific attribute value, such as "business-relevance = business-relevant".

Configure fine-grain mode:

```
enable
configure terminal
ip nbar classification granularity fine-grain { [protocol protocol-name] | [attribute
attribute-type attribute-value] }
exit
```

Display the currently configured NBAR classification mode:

```
show ip nbar classification granularity { [protocol protocol-name] | [attribute attribute-type
 attribute-value] }
```

Example

This example configures fine-grain mode for the application protocol, **cisco-media-audio**, then verifies with the **show** command.

```
Device#enable
Device#configuration terminal
Device(config)#ip nbar classification granularity fine-grain protocol cisco-media-audio
Device(config)#exit
Device#show ip nbar classification granularity protocol cisco-media-audio

Protocol                  Force mode
------------------------------------
cisco-media-audio         fine-grain
```

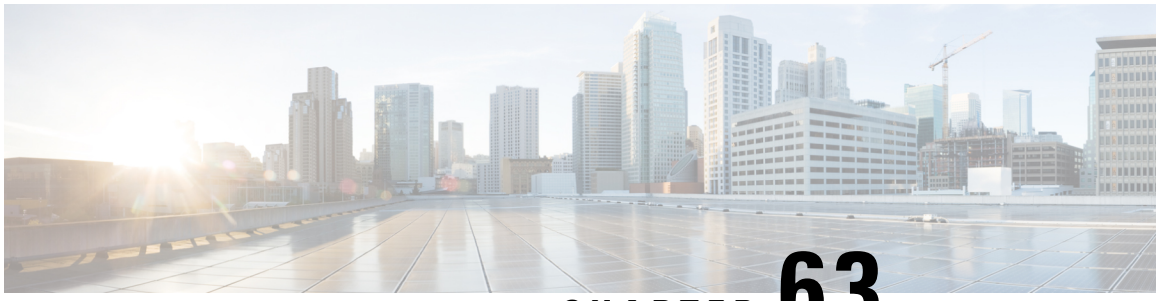
Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*
AVC information	*AVC User Guide*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	https://www.cisco.com/c/en/us/support/index.html



CHAPTER 63

NBAR Custom Applications Based on DNS Name

NBAR Custom Applications based on DNS Name feature provides the mechanism to customize applications based on the Domain Name System (DNS) hostnames.

Prerequisites for NBAR Custom Applications Based on DNS Name

You must have basic knowledge of domain names.

Restrictions for NBAR Custom Applications Based on DNS Name

To use Domain Name System (DNS), you must have a DNS name server on your network.

DNS permits reading of UDP type messages only and considers only those response packets which have a source port of 53.

Information About NBAR Custom Applications Based on DNS Name

Overview of NBAR Custom Applications Based on DNS Name

Network-Based Application Recognition (NBAR) recognizes and classifies network traffic on the basis of a set of protocols and application types. The user adds to the set of protocols and application types that NBAR recognizes by creating custom protocols.

The user provides the DNS hostname signatures using the**ip nbar custom** *custom1 dns domain-name regular-expression id* command in the form of a simplified regular expression, which the DNS server pushes to the DNS templates. The DNS-based classification functions only when the IP addresses derived as direct responses are added to the look up table (LUT) for future classification lookups.

The following types of domains are supported:

- A
- AAAA
- CNAME

When you define the **ip nbar custom myDns dns domain-name** **example* command, the DNS traffic for a domain name that matches the expression "example" reaches the device. NBAR stores the corresponding IP address A.B.C.D of domain that matches the domain name with the expression "example" in its tables. When any TCP or UDP traffic with IP address A.B.C.D arrives, it is classified as myDns protocol.

How to Configure NBAR Custom Applications Based on DNS Name

Configuring the NBAR Custom Applications Based on DNS Name

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *custom-name* **dns** *domain-name regular-expression* **id** *1*
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar custom** *custom-name* **dns** *domain-name regular-expression* **id** *1* **Example:** `Device(config)# ip nbar custom cust1 dns dns-name *example.com id 1`	Configures the NBAR Custom Applications Based on DNS Name feature. **Note** You can provide either the full domain name or a part of it as a regular expression. For example: the expression "*example" will match any domain that contains the word "example".
Step 4	**exit** **Example:** `Device(config)# exit`	Exits the global configuration mode and enters privileged EXEC mode.

Configuration Examples for NBAR Custom Applications Based on DNS Name

Example: Configuring NBAR Custom Applications Based on DNS Name

```
Device> enable
Device# configure terminal
Device(config)#  ip nbar custom custom1 dns domain-name *example id 11
Device(config)# exit
```

Additional References for NBAR Custom Applications Based on DNS Name

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

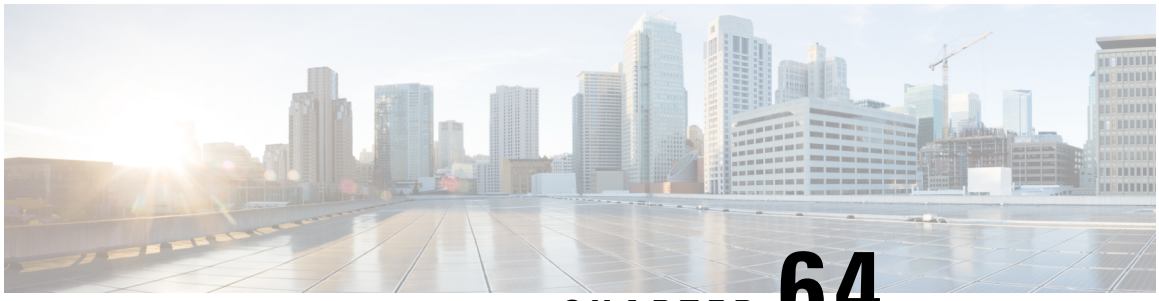
Feature Information for NBAR Custom Applications Based on DNS Name

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 76: Feature Information for NBAR Custom Applications Based on DNS Name

Feature Name	Releases	Feature Information
NBAR Custom Applications Based on DNS Name	Cisco IOS XE Release 3.15S	NBAR custom applications based on Domain Name Service (DNS) Name feature provides the mechanism to customize applications based on the DNS hostnames. The following command was introduced or modified: **ip nbar custom**.



CHAPTER 64

DNS Protocol Classification Change

Traffic for a network application includes DNS query/response traffic and the actual application flow. Using the DNS Protocol Classification Change feature, NBAR2 can be configured to classify and handle DNS traffic in the same way as its associated application traffic.

This module describes DNS Protocol Classification Change and the how to enable it.

Prerequisites for DNS Protocol Class Change

None.

Information About DNS Protocol Classification Change

DNS Protocol Classification Change

Traffic for a network application includes DNS query/response traffic and the actual application flow. When classifying traffic, most attention is given to the application flow, both for reporting (application visibility) and control (QoS policy).

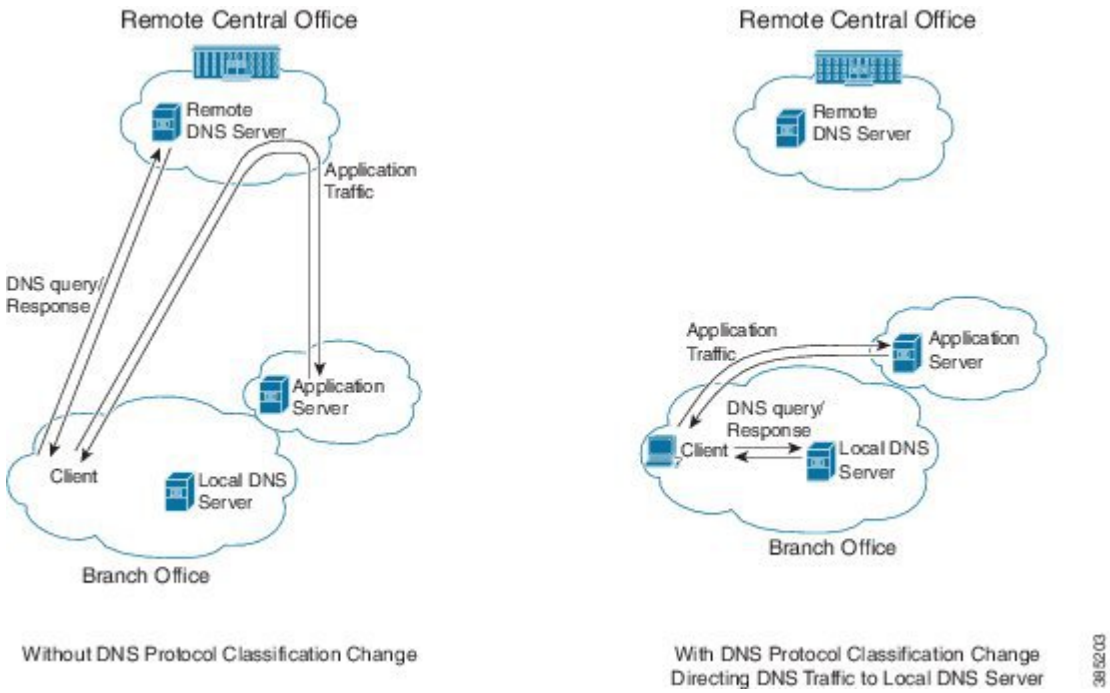
The DNS Protocol Classification Change feature enables an NBAR client, such as a router, to classify and handle DNS traffic in the same way as its associated application traffic. This is accomplished using the domain name that appears in the DNS flow.

Use of DNS Protocol Classification Change

DNS Protocol Classification Change can be especially useful in networks employing Cisco Intelligent WAN (IWAN), for optimizing the performance of network applications.

For example, in an IWAN spanning a wide geography, it might happen that a specific type of application traffic (example: Microsoft Office 365) may be routed first to a geographically distant node in the IWAN, and then to the relevant server. This route may diminish performance of the application. Using DNS protocol classification change, it is possible to redirect the DNS query/response to a local DNS server, and route the application traffic directly to the relevant cloud-based application server, improving application performance.

Figure 97: DNS Protocol Classification Change Improving Application Performance in an IWAN Environment



Usage Notes

- DNS Protocol Classification Change classifies the DNS flow in the same way as the application, based on built-in protocols or custom signatures.
- The DNS flow classification inherits the attributes of the application – category, business-relevance, traffic-class, encryption, and so on. For example, for a DNS flow classified as “Google-accounts” the encryption attribute is TRUE.
- DNS flows are not cached using the socket cache mechanism.
- To catch all DNS traffic for QoS, use the following “transport hierarchy” CLI:

  match protocol dns in-app-hierarchy

- Default: enabled.

How to Enable DNS Protocol Classification Change

Enabling DNS Protocol Classification Change

Enabling the DNS Protocol Classification Change feature enables an NBAR client, such as a router, to classify and handle DNS traffic in the same way as its associated application traffic.

The **no** form of the command disables the feature.

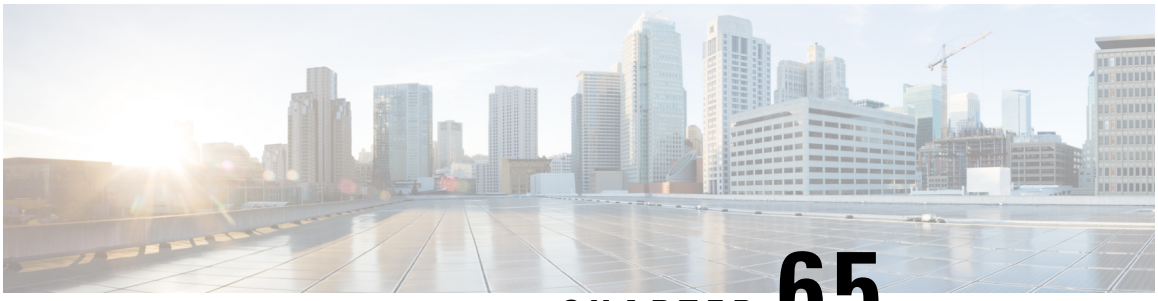
[**no**] **ip nbar classification dns classify-by-domain**

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar classification dns classify-by-domain**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip nbar classification dns classify-by-domain** **Example:** `Device(config)# ip nbar classification dns classify-by-domain`	Enables the DNS Protocol Classification Change feature.



CHAPTER 65

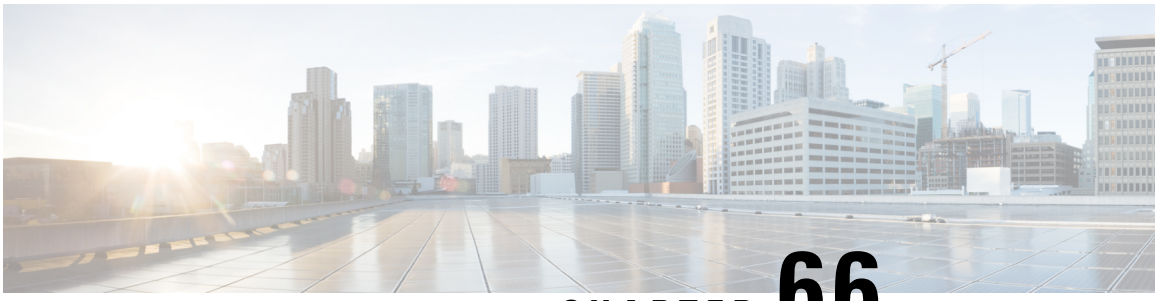
About Attributes

The information that NBAR2 uses to recognize and classify application traffic is organized as application protocols. Each protocol has a set of attributes that relate to the specific network application. The list of attribute types are provided here.

- Attribute Types, on page 825

Attribute Types

	Attribute	Description
Categorization	application-family	Categorization of the application: scheme 1.
	application-set	Categorization of the application: scheme 2.
	category	Categorization of the application: scheme 3.
	sub-category	Application usage.
Service	application-group	Group of applications that belong to the same service.
Priority	business-relevance	Indicates business-oriented applications.
Traffic attributes	encrypted	Possible encrypted traffic.
	p2p-technology	Peer-to-peer traffic.
	traffic-class	Application class-of-service (based on RFC 4594).
	tunnel	Tunnel-related traffic.



CHAPTER 66

Customizing NBAR2 Built-in Protocols

Built-in protocols provided by the Cisco Protocol Pack recognize traffic of a specific type of network application. It can be useful to “customize” a protocol, adding to the scope of traffic that it matches and recognizes.

This module describes the process and shows how to customize built-in protocols.

Information About Customizing a Built-in Protocol

Customizing Built-in Protocols

Each built-in NBAR2 protocol (provided by the Cisco Protocol Pack) is pre-configured to recognize traffic of a specific type of network application. In some situations, it can be useful to “customize” a protocol, adding to the scope of traffic that it matches and recognizes. This is accomplished by configuring user-specified domains that extend the scope of the protocol. Each customization is identified by a user-supplied name.

For example, the built-in office365 protocol matches Microsoft Office 365 application traffic. Customizing the office365 protocol by adding additional domains can extend its scope.

Visibility and Control

- **Application visibility**: Traffic that matches the user-specified extension of the built-in protocol is reported by the name of the user-specified customization.
- **Application control**: After extending a built-in protocol, any policy associated with the protocol applies also to the user-specified domain.

Usage Notes

- The maximum number of customizations is 120. This count includes other types of customization.
- Customizing a protocol does not change its priority.
- The *custom-name* of a customization cannot be used for defining policy.

- It is possible to configure multiple domains for the same *custom-name*. Example:

```
ip nbar custom myOffice365 dns domain-name "*uniqueOffice365" extends office365
ip nbar custom myOffice365 dns domain-name "*anotherUniqueOffice365" extends office365
```

- Multiple customization commands can extend the same built-in protocol. Example:

```
ip nbar custom myOffice365_D1 dns domain-name "*uniqueOffice365" extends office365
ip nbar custom myOffice365_D2 dns domain-name "*anotherUniqueOffice365" extends office365
```

How to Customize a Built-in Protocol

The following CLI commands can be used to customize a protocol.

- Adding user-specified domains for DNS traffic only:

  ip nbar custom *custom-name* **dns domain-name** "*regex-text-string*" **extends** *built-in-protocol*

- Adding user-specified domains for any type of transport protocol (DNS, HTTP, SSL):

  ip nbar custom *custom-name* **composite server-name** "*regex-text-string*" **extends** *built-in-protocol*

- The **no** form of the command removes the customization. Specify the custom name, regular expression (regex), and built-in protocol name exactly as they were specified when the customization was added.

  no ip nbar custom *custom-name* {**dns domain-name** | **composite server-name**} "*regex-text-string*" **extends** *built-in-protocol*

Customizing a Built-in Protocol

Use the following procedure to customize a protocol.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip nbar custom** *custom-name* {**dns domain-name** | **composite server-name**} "*regex-text-string*" **extends** *built-in-protocol*

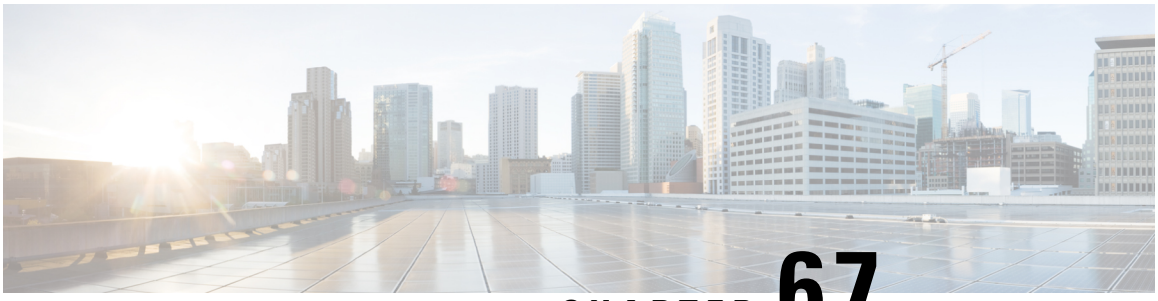
DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**ip nbar custom** *custom-name* {**dns domain-name** | **composite server-name**} "*regex-text-string*" **extends** *built-in-protocol* **Example:** `Device(config)# ip nbar custom myOffice365 dns domain-name "*uniqueOffice365" extends office365`	Adds the custom domain, defined by a regular expression (regex). • Use **dns domain-name** to add a user-specified domain for DNS traffic only. • Use **composite server-name** to add a user-specified domain for any type of transport protocol (DNS, HTTP, SSL). • *custom-name*: User-specified name for the customization. • *regex-text-string*: Specifies domain text to match. • *built-in-protocol*: Name of the built-in protocol to customize. The command extends the scope of this built-in protocol to include traffic matched by the *regex-text-string*. The example configures a customization called myOffice365, which extends the built-in office365 protocol to include domains that match to the regex, "*uniqueOffice365".

PART III

QoS RSVP



CHAPTER 67

RSVP Aggregation

The RSVP Aggregation feature allows the Resource Reservation Protocol (RSVP) state to be reduced within an RSVP/DiffServ network by aggregating many smaller reservations into a single, larger reservation at the edge.

Prerequisites for RSVP Aggregation

You must configure at least two aggregating nodes (provider edge [PE] devices), one interior node (provider [P] device) and two end user nodes (customer edge [CE] devices) within your network.

You must configure your network to support the following Cisco IOS features:

- RSVP
- Class Based Weighted Fair Queuing (CBWFQ)
- RSVP Scalability Enhancements

Note You configure these features because Cisco IOS Release 12.2(33)SRC supports control plane aggregation only. Dataplane aggregation must be achieved by using the RSVP Scalability Enhancements.

Restrictions for RSVP Aggregation

Functionality Restrictions

The following functionality is not supported:

- Multilevel aggregation
- Multiple, adjacent aggregation regions
- Dynamic resizing of aggregate reservations
- Policing of end-to-end (E2E) reservations by the aggregator
- Policing of aggregate reservations by interior devices
- Differentiated Services Code Point (DSCP) marking by the aggregator
- Equal Cost Multiple Paths (ECMP) load-balancing within the aggregation region
- RSVP Fast Local Repair in case of a routing change resulting in a different aggregator or deaggregator, admission control is performed on E2E PATH refresh
- Multicast RSVP reservations
- RSVP policy servers including Common Open Policy Server (COPS)
- Dataplane aggregation

The following functionality is supported:

- Multiple, non-adjacent aggregation regions
- Control plane aggregation

Note RSVP/DiffServ using CBWFQ provides the dataplane aggregation.

Configuration Restrictions

- Sources should not send marked packets without an installed reservation.
- Sources should not send marked packets that exceed the reserved bandwidth.
- Sources should not send marked packets to a destination other than the reserved path.
- All RSVP capable devices within an aggregation region regardless of role must support the aggregation feature to recognize the RFC 3175 RSVP message formats properly.
- E2E reservations must be present to establish dynamic aggregates; aggregates cannot be established manually.
- Aggregates are established at a fixed bandwidth regardless of the number of current E2E reservations being aggregated.

- Aggregators and deaggregators must be paired to avoid null routing of E2E reservations because of dynamic aggregate establishment.

Note Null routing means that the reservation is never established. If an E2E reservation crosses from an exterior to an interior interface, the E2E reservation turns into an RSVP-E2E-IGNORE protocol packet. If there is no corresponding deaggregator, a device where this RSVP-E2E-IGNORE reservation crosses an interior to an exterior interface, then the RSVP-E2E-IGNORE reservation is never restored to an E2E reservation. The RSVP-E2E-IGNORE reservation eventually reaches its destination, which is the RSVP receiver; however, the RSVP receiver does not know what to do with the RSVP-E2E-IGNORE reservation and discards the packet.

Information About RSVP Aggregation

Feature Overview of RSVP Aggregation

High Level Overview

The establishment of a single RSVP reservation requires a large amount of resources including memory allocated for the associated data structures, CPU for handling signaling messages, I/O operations for datapath programming, interprocess communication, and signaling message transmission.

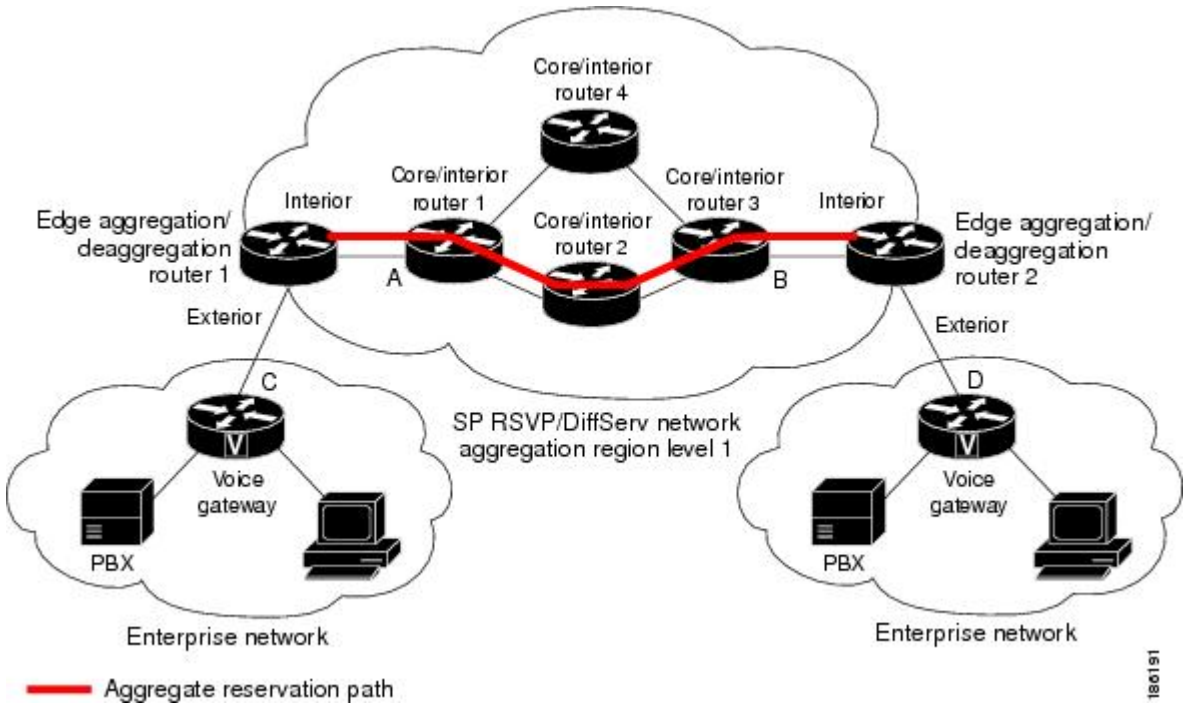
When a large number of small reservations are established, the resources required for setting and maintaining these reservations may exceed a node's capacity to the point where the node's performance is significantly degraded or it becomes unusable. The RSVP Aggregation feature addresses this scalability issue by introducing flow aggregation.

Flow aggregation is a mechanism wherein RSVP state can be reduced within a core device by aggregating many smaller reservations into a single, larger reservation at the network edge. This preserves the ability to perform connection admission control on core device links within the RSVP/DiffServ network while reducing signaling resource overhead.

How Aggregation Functions

Common segments of multiple end-to-end (E2E) reservations are aggregated over an aggregation region into a larger reservation that is called an aggregate reservation. An aggregation region is a connected set of nodes that are capable of performing RSVP aggregation as shown in the figure below.

Figure 98: RSVP Aggregation Network Overview



There are three types of nodes within an aggregation region:

- Aggregator--Aggregates multiple E2E reservations.
- Deaggregator--Deaggregates E2E reservations; provides mapping of E2E reservations onto aggregates.
- Interior--Neither aggregates or deaggregates, but is an RSVP core router that understands RFC 3175 formatted RSVP messages. Core/interior routers 1 through 4 are examples shown in the figure above.

There are two types of interfaces on the aggregator/deaggregator nodes:

- Exterior interface--The interface is not part of the aggregate region.
- Interior interface--The interface is part of the aggregate region.

Any router that is part of the aggregate region must have at least one interior interface and may have one or more exterior interfaces. Depending on the types of interfaces spanned by an IPv4 flow, a node can be an aggregator, a deaggregator, or an interior router with respect to that flow.

Aggregate RSVP DiffServ Integration Topology

RSVP aggregation further enhances RSVP scalability within an RSVP/DiffServ network as shown in the figure above by allowing the establishment of aggregate reservations across an aggregation region. This allows for aggregated connection admission control on core/interior device interfaces. Running RSVP on the core/interior devices allows for more predictable bandwidth use during normal and failure scenarios.

The voice gateways are running classic RSVP, which means RSVP is keeping a state per flow and also classifying, marking, and scheduling packets on a per-flow basis. The edge/aggregation devices are running RSVP with scalability enhancements for admission control on the exterior interfaces connected to the voice gateways and running RSVP aggregation on the interfaces connected to core/interior devices 1 and 3. The

core/interior devices in the RSVP/DiffServ network are running RSVP for the establishment of the aggregate reservations. The edge and core/interior devices inside the RSVP/DiffServ network also implement a specific per hop behavior (PHB) for a collection of flows that have the same DSCP.

The voice gateways identify voice data packets and set the appropriate DSCP in their IP headers so that the packets are classified into the priority class in the edge/aggregation devices and in core/interior devices 1, 2, 3 or 1, 4, 3.

The interior interfaces on the edge/aggregation/deaggregation devices (labeled A and B) connected to core/interior devices 1 and 3 are running RSVP aggregation. They are performing admission control only per flow against the RSVP bandwidth of the aggregate reservation for the corresponding DSCP.

Admission control is performed at the deaggregator because it is the first edge node to receive the returning E2E RSVP RESV message. CBWFQ is performing the classification, policing, and scheduling functions on all nodes within the RSVP/DiffServ network including the edge devices.

Aggregate reservations are dynamically established over an aggregation region when an E2E reservation enters an aggregation region by crossing from an exterior to an interior interface; for example, when voice gateway C initiates an E2E reservation to voice gateway D. The aggregation is accomplished by "hiding" the E2E RSVP messages from the RSVP nodes inside the aggregation region. This is achieved with a new IP protocol, RSVP-E2E-IGNORE, that replaces the standard RSVP protocol in E2E PATH, PATHTEAR, and RESVCONF messages. This protocol change to RSVP-E2E-IGNORE is performed by the aggregator when the message enters the aggregation region and later restored back to RSVP by the deaggregator when the message exits the aggregation region. Thus, the aggregator and deaggregator pairs for a given flow are dynamically discovered during the E2E PATH establishment.

The deaggregator device 2 is responsible for mapping the E2E PATH onto an aggregate reservation per the configured policy. If an aggregate reservation with the corresponding aggregator device 1 and a DSCP is established, the E2E PATH is forwarded. Otherwise a new aggregate at the requisite DSCP is established, and then the E2E PATH is forwarded. The establishment of this new aggregate is for the fixed bandwidth parameters configured at the deaggregator device 2. Aggregate PATH messages are sent from the aggregator to the deaggregator using RSVP’s normal IP protocol. Aggregate RESV messages are sent back from the deaggregator to the aggregator, thus establishing an aggregate reservation on behalf of the set of E2E flows that use this aggregator and deaggregator. All RSVP capable interior nodes process the aggregate reservation request following normal RSVP processing including any configured local policy.

The RSVP-E2E-IGNORE messages are ignored by the core/interior devices, no E2E reservation states are created, and the message is forwarded as IP. As a consequence, the previous hop/next hop (PHOP/ NHOP) for each RSVP-E2E-IGNORE message received at the deaggregator or aggregator is the aggregator or deaggregator node. Therefore, all messages destined to the next or previous hop (RSVP error messages, for example) do not require the protocol to be changed when they traverse the aggregation region.

By setting up a small number of aggregate reservations on behalf of a large number of E2E flows, the number of states stored at core/interior devices and the amount of signal processing within the aggregation region is reduced.

In addition, by using differentiated services mechanisms for classification and scheduling of traffic supported by aggregate reservations rather than performing per aggregate reservation classification and scheduling, the amount of classification and scheduling state in the aggregation region is further reduced. This reduction is independent of the number of E2E reservations and the number of aggregate reservations in the aggregation region. One or more RSVP/DiffServ DSCPs are used to identify the traffic covered by aggregate reservations, and one or more RSVP/DiffServ per hop behaviors (PHBs) are used to offer the required forwarding treatment to this traffic. There may be more than one aggregate reservation between the same pair of devices, each representing different classes of traffic and each using a different DSCP and a different PHB.

Integration with RSVP Features

RSVP aggregation has been integrated with many RSVP features, including the following:

- RSVP Fast Local Repair
- RSVP Local Policy Support
- RSVP Refresh Reduction and Reliable Messaging

Benefits of RSVP Aggregation

Enhanced Scalability

Aggregating a large number of small reservations into one reservation requires fewer resources for signaling, setting, and maintaining the reservation thereby increasing scalability.

Enhanced Bandwidth Usage within RSVP/DiffServ Core Network

Aggregate reservations across an RSVP/DiffServ network allow for more predictable bandwidth use of core links across RSVP/DiffServ PHBs. Aggregate reservations can use RSVP fast local repair and local policy preemption features for determining bandwidth use during failure scenarios.

How to Configure RSVP Aggregation

Configuring RSVP Scalability Enhancements

Perform these tasks on all nodes within the aggregation region including aggregators, deaggregators, and interior nodes.

Enabling RSVP on an Interface

Perform this task to enable RSVP on all the interfaces along the path from the sender to the receiver.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip routing**
4. **ip vrf** *vrf-name*
5. **exit**
6. **interface** *type number*
7. **ip vrf forwarding** *vrf-name*
8. **ip rsvp bandwidth** [*interface-kbps*] [*single-flow-kbps*]
9. Repeat the previous step for each interface that you want to enable.
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip routing** **Example:** `Device(config)# ip routing`	Enables IP routing.
Step 4	**ip vrf** *vrf-name* **Example:** `Device(config)# ip vrf vrf1`	Defines a VRF instance and enters VRF configuration mode.
Step 5	**exit** **Example:** `Device(config-vrf)# exit`	Exits VRF configuration mode and enters global configuration mode.
Step 6	**interface** *type number* **Example:** `Device(config)# interface Ethernet0/0`	Configures the interface type and enters interface configuration mode.
Step 7	**ip vrf forwarding** *vrf-name* **Example:** `Device(config-if)# ip vrf forwarding vrf1`	Associates a VRF instance with an interface or subinterface.
Step 8	**ip rsvp bandwidth** [*interface-kbps*] [*single-flow-kbps*] **Example:** `Device(config-if)# ip rsvp bandwidth 1158 100`	Enables RSVP bandwidth on an interface. • The optional *interface-kbps* and *single-flow-kbps* arguments specify the amount of bandwidth that can be allocated by RSVP flows or to a single flow, respectively. Values are from 1 to 10000000. **Note** Repeat this command for each interface that you want to enable.
Step 9	Repeat the previous step for each interface that you want to enable.	--

	Command or Action	Purpose
Step 10	**end** **Example:** `Device(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Setting the Resource Provider

Note Resource provider was formerly called QoS provider.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp bandwidth** [*interface-kbps* [*single-flow-kbps*[**bc1** *kbps* | **sub-pool** *kbps*]*]*| **percent** *percent-bandwidth* [*single-flow-kbps*]]
4. **ip rsvp resource-provider** [**none** | **wfq-interface** | **wfq-pvc**]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp bandwidth** [*interface-kbps* [*single-flow-kbps*[**bc1** *kbps*	**sub-pool** *kbps*]*]*	**percent** *percent-bandwidth* [*single-flow-kbps*]] **Example:** `Router(config-if)# ip rsvp bandwidth 500 500`	Configures the interface type and enters interface configuration mode.
Step 4	**ip rsvp resource-provider** [**none**	**wfq-interface**	**wfq-pvc**] **Example:** `Router(config-if)# ip rsvp resource-provider none`	Sets the resource provider. • Enter the optional **none** keyword to set the resource provider to none regardless of whether one is configured on the interface.

	Command or Action	Purpose
		Note Setting the resource provider to **none** instructs RSVP to *not* associate any resources, such as weighted fair queueing (WFQ) queues or bandwidth, with a reservation. • Enter the optional **wfq-interface** keyword to specify WFQ as the resource provider on the interface. • Enter the optional **wfq-pvc** keyword to specify WFQ as the resource provider on the permanent virtual circuit (PVC) or connection.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Disabling Data Packet Classification

Note Disabling data packet classification instructs RSVP not to process every packet, but to perform admission control only.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / subslot / port*
4. **ip rsvp data-packet classification none**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**interface** *type slot / subslot / port* **Example:** `Router(config)# interface gigabitEthernet 0/0/0`	Configures the interface type and enters interface configuration mode.
Step 4	**ip rsvp data-packet classification none** **Example:** `Router(config-if)# ip rsvp data-packet classification none`	Disables data packet classification.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Configuring Class and Policy Maps

To configure class and policy maps, use the following commands, beginning in global configuration mode:

SUMMARY STEPS

1. Device(config)# **class-map** *class-map-name*
2. Device(config)# **policy-map** *policy-map-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Device(config)# **class-map** *class-map-name*	Specifies the name of the class for which you want to create or modify class map match criteria.
Step 2	Device(config)# **policy-map** *policy-map-name*	Specifies the name of the policy map to be created, added to, or modified before you can configure policies for classes whose match criteria are defined in a class map.

Attaching a Policy Map to an Interface

Note If at the time you configure the RSVP scalability enhancements, there are existing reservations that use classic RSVP, no additional marking, classification, or scheduling is provided for these flows. You can also delete these reservations after you configure the RSVP scalability enhancements.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **interface** *type slot / subslot / port*
4. **service-policy** [**type access-control**] {**input** | **output**} *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / subslot / port* **Example:** `Router(config)# interface gigabitEthernet 0/0/0`	Configures the interface type and enters interface configuration mode.
Step 4	**service-policy** [**type access-control**] {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# service-policy output POLICY-ATM`	Specifies the name of the policy map to be attached to the input or output direction of the interface. **Note** Policy maps can be attached in the input or output direction of an interface. The direction and the router to which the policy map should be attached vary according to the network configuration. When using the **service-policy** command to attach the policy map to an interface, be sure to choose the router and the interface direction that are appropriate for the network configuration. • The optional **type access-control** keywords determine the exact pattern to look for in the protocol stack of interest. • Enter the *policy-map name*.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Configuring Interfaces with Aggregation Role

Perform this task on aggregator and deaggregators to specify which interfaces are facing the aggregation region.

Note You do not need to perform this task on interior routers; that is, nodes having interior interfaces only.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / subslot / port*
4. **ip rsvp aggregation role interior**
5. Repeat Step 4 as needed to configure additional aggregator and deaggregator interfaces.
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / subslot / port* **Example:** `Router(config)# interface gigabitEthernet 0/0/0`	Configures the interface type and enters interface configuration mode.
Step 4	**ip rsvp aggregation role interior** **Example:** `Router(config-if)# ip rsvp aggregation role interior`	Enables RSVP aggregation on an aggregator or deaggregator's interface.
Step 5	Repeat Step 4 as needed to configure additional aggregator and deaggregator interfaces.	Configures additional aggregator and deaggregator interfaces.
Step 6	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Configuring Aggregation Mapping on a Deaggregator

Note Typically, an edge router acts as both an aggregator and deaggregator because of the unidirectional nature of RSVP reservations. Most applications require bidirectional reservations. Therefore, these parameters are used by a deaggregator when mapping E2E reservations onto aggregates during the dynamic aggregate reservation process.

Before you begin

You should configure an access control list (ACL) to define a group of RSVP endpoints whose reservations will be aggregated onto a single aggregate reservation session identified by the specified DSCP. Then for each ACL, define a map configuration.

Note In classic (unaggregated) RSVP, a session is identified in the reservation message session object by the destination IP address and protocol information. In RSVP aggregation, a session is identified by the destination IP address and DSCP within the session object of the aggregate RSVP message. E2E reservations are mapped onto a particular aggregate RSVP session identified by the E2E reservation session object alone or a combination of the session object and sender template or filter spec.

Extended ACLs

The ACLs used within the **ip rsvp aggregation ip map** command match the RSVP message objects as follows for an extended ACL:

- Source IP address and port match the RSVP PATH message sender template or RSVP RESV message filter spec; this is the IP source or the RSVP sender.
- Destination IP address and port match the RSVP PATH/RESV message session object IP address; this is the IP destination address or the RSVP receiver.
- Protocol matches the RSVP PATH/RESV message session object protocol; if protocol = IP, then it matches the source or destination address as above.

Standard ACLs

The ACLs used within the **ip rsvp aggregation ip map** command match the RSVP message objects as follows for a standard ACL:

- IP address matches the RSVP PATH message sender template or RSVP RESV message filter spec; this is the IP source address or the RSVP sender.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp aggregation ip map** {**access-list** {*acl-number*} | **any**} **dscp** *value*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp aggregation ip map** {**access-list** {*acl-number*} | **any**} **dscp** *value* **Example:** `Router(config)# ip rsvp aggregation ip map any dscp af41`	Configures RSVP aggregation rules that tell a router how to map E2E reservations onto aggregate reservations. • The keywords and arguments specify additional information such as DSCP values.
Step 4	**end** **Example:** `Router(config)# end`	(Optional) Returns to privileged EXEC mode.

Configuring Aggregate Reservation Attributes on a Deaggregator

Perform this task on a deaggregator to configure the aggregate reservation attributes (also called token bucket parameters) on a per-DSCP basis.

Note Typically, an edge device acts as both an aggregator and deaggregator because of the unidirectional nature of RSVP reservations. Most applications require bidirectional reservations. Therefore, these parameters are used by a deaggregator when mapping E2E reservations onto aggregates during the dynamic aggregate reservation process.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp aggregation ip reservation dscp** *value* [**aggregator** *agg-ip-address*] **traffic-params static rate** *data-rate* [**burst** *burst-size*] [**peak** *peak-rate*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp aggregation ip reservation dscp** *value* [**aggregator** *agg-ip-address*] **traffic-params static rate** *data-rate* [**burst** *burst-size*] [**peak** *peak-rate*] **Example:** `Device(config)# ip rsvp aggregation ip reservation dscp af11 aggregator 10.10.10.10 traffic-params static rate 10 burst 8 peak 10`	Configures RSVP aggregate reservation attributes (also called token bucket parameters) on a per-DSCP basis. • The keywords and arguments specify additional information.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Returns to privileged EXEC mode.

Configuring an RSVP Aggregation Device ID

Perform this task on aggregators and deaggregators to configure an RSVP aggregation device ID.

Note Both aggregators and deaggregators need to be identified with a stable and routable IP address. This is the RFC 3175 device ID, which is also the IP address of the loopback interface with the lowest number. If there is no loopback interface configured or all those configured are down, then there will be no device ID assigned for the aggregating/deaggregating function and aggregate reservations will not be established.

Note The device ID may change if the associated loopback interface goes down or its IP address is removed. In this case, the E2E and aggregate sessions are torn down. If a new device ID is determined, new E2E and aggregate sessions will use the new device ID.

SUMMARY STEPS

1. **enable**

2. **configure terminal**
3. **interface loopback** *number*
4. **ip address** *ip-address subnet-mask/prefix*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface loopback** *number* **Example:** `Device(config)# interface loopback 1`	Creates a loopback interface and enters interface configuration mode. • Enter a value for the *number* argument. The range is 0 to 2147483647.
Step 4	**ip address** *ip-address subnet-mask/prefix* **Example:** `Device(config-if)# ip address 192.168.50.1 255.255.255.0`	Configures an IP address and subnet mask or prefix on the loopback interface.
Step 5	**end** **Example:** `Device(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Enabling RSVP Aggregation

Perform this task on aggregators and deaggregators to enable RSVP aggregation globally after you have completed all the previous aggregator and deaggregator configurations.

Note This task registers a device to receive RSVP-E2E-IGNORE messages. It is not necessary to perform this task on interior devices because they are only processing RSVP aggregate reservations. If you do so, you may decrease performance because the interior device will then unnecessarily process all the RSVP-E2E-IGNORE messages.

Note If you enable RSVP aggregation globally on an interior device, then you should configure all interfaces as interior.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp aggregation ip**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp aggregation ip** **Example:** `Device(config)# ip rsvp aggregation ip`	Enables RSVP aggregation globally on an aggregator or deaggregator.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Returns to privileged EXEC mode.

Configuring RSVP Local Policy

Perform this task to apply a local policy to an RSVP aggregate reservation.

Note In classic (unaggregated) RSVP, a session is identified in the reservation message session object by the destination IP address and protocol information. In RSVP aggregation, a session is identified by the destination IP address and DSCP within the session object of the aggregate RSVP message. The **dscp-ip** keyword matches the DSCP within the session object.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp policy local** {**acl** *acl1*[*acl2...acl8*] | **dscp-ip** *value1* [*value2 ... value8*] | **default** | **identity** *alias1* [*alias2...alias4*] | **origin-as** *as1*[*as2...as8*]}
4. {**accept** | **forward** [**all** | **path**| **path-error** | **resv**| **resv-error**] | **default** | **exit** | **fast-reroute** | **local-override** | **maximum** {**bandwidth** [**group** *x*] [**single** *y*] | **senders** *n*}| **preempt-priority** [**traffic-eng** *x*] *setup-priority* [*hold-priority*]}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp policy local** {**acl** *acl1*[*acl2...acl8*]	**dscp-ip** *value1* [*value2 ... value8*]	**default**	**identity** *alias1* [*alias2...alias4*]	**origin-as** *as1*[*as2...as8*]} **Example:** `Router(config)# ip rsvp policy local dscp-ip 46`	Creates a local policy to determine how RSVP resources are used in a network and enters local policy configuration mode. • Enter the **dscp-ip** *value*keyword and argument combination to specify a DSCP for matching the session object DCSP within the aggregate reservations. Values can be the following: • 0 to 63--Numerical. The default value is 0. • af11 to af43--Assured forwarding (AF). • cs1 to cs7--Type of service (ToS) precedence. • default--Default DSCP. • ef--Expedited Forwarding (EF). **Note** You must associate at least one DSCP with a DSCP-based policy. However, you can associate as many as eight.
Step 4	{**accept**	**forward** [**all**	**path**| **path-error**	**resv**| **resv-error**]	**default**	**exit**	**fast-reroute**	**local-override**	**maximum** {**bandwidth** [**group** *x*] [**single** *y*]	**senders** *n*}| **preempt-priority** [**traffic-eng** *x*] *setup-priority* [*hold-priority*]} **Example:**	(Optional) Defines the properties of the dscp-ip local policy that you are creating. (These are the submode commands.) **Note** This is an optional step. An empty policy rejects everything, which may be desired in some cases.

	Command or Action	Purpose
	`Router(config-rsvp-policy-local)# forward all`	See the **ip rsvp policy local** command for more detailed information on submode commands.
Step 5	**end** **Example:** `Router(config-rsvp-policy-local)# end`	(Optional) Exits local policy configuration mode and returns to privileged EXEC mode.

Verifying the RSVP Aggregation Configuration

Note You can use the following **show** commands in user EXEC or privileged EXEC mode.

SUMMARY STEPS

1. **enable**
2. **show ip rsvp aggregation ip** [**endpoints** | **interface** [*if-name*] | **map** [**dscp** *value*]| **reservation** [**dscp** *value*[**aggregator** *ip-address*]]
3. **show ip rsvp aggregation ip endpoints** [**role**{**aggregator**| **deaggregator**}] [*ip-address*] [**dscp** *value*] [**detail**]
4. **show ip rsvp** [**atm-peak-rate-limit**| **counters**| **host**| **installed**| **interface**| **listeners**| **neighbor**| **policy**| **precedence**| **request**| **reservation**| **sbm**| **sender**| **signalling**| **tos**]
5. **show ip rsvp reservation** [**detail**] [**filter**[**destination** *ip-address* | *hostname*] [**dst-port** *port-number*] [**source** *ip-address* | *hostname*][**src-port** *port-number*]]
6. **show ip rsvp sender** [**detail**] [**filter**[**destination** *ip-address* | *hostname*] [**dst-port** *port-number*] [**source** *ip-address* | *hostname*][**src-port** *port-number*]]
7. **show ip rsvp installed** [*interface-type interface-number*] [**detail**]
8. **show ip rsvp interface** [**detail**] [*interface-type interface-number*]
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	(Optional) Enables privileged EXEC mode. • Enter your password if prompted. **Note** Skip this step if you are using the **show** commands in user EXEC mode.
Step 2	**show ip rsvp aggregation ip** [**endpoints**	**interface** [*if-name*]	**map** [**dscp** *value*]| **reservation** [**dscp** *value*[**aggregator** *ip-address*]] **Example:**	(Optional) Displays RSVP summary aggregation information. • The optional keywords and arguments display additional information.

	Command or Action	Purpose
	`Device# show ip rsvp aggregation ip`	
Step 3	**show ip rsvp aggregation ip endpoints** [**role**{**aggregator**| **deaggregator**}] [*ip-address*] [**dscp** *value*] [**detail**] **Example:** `Device# show ip rsvp aggregation ip endpooints`	(Optional) Displays RSVP information about aggregator and deaggregator devices for currently established aggregate reservations. • The optional keywords and arguments display additional information.
Step 4	**show ip rsvp** [**atm-peak-rate-limit**| **counters**| **host**| **installed**| **interface**| **listeners**| **neighbor**| **policy**| **precedence**| **request**| **reservation**| **sbm**| **sender**| **signalling**| **tos**] **Example:** `Device# show ip rsvp`	(Optional) Displays specific information for RSVP categories. • The optional keywords display additional information.
Step 5	**show ip rsvp reservation** [**detail**] [**filter**[**destination** *ip-address* | *hostname*] [**dst-port** *port-number*] [**source** *ip-address* | *hostname*][**src-port** *port-number*]] **Example:** `Device# show ip rsvp reservation detail`	(Optional) Displays RSVP-related receiver information currently in the database. • The optional keywords and arguments display additional information. **Note** The optional **filter** keyword is supported in Cisco IOS Releases 12.0S and 12.2S only.
Step 6	**show ip rsvp sender** [**detail**] [**filter**[**destination** *ip-address* | *hostname*] [**dst-port** *port-number*] [**source** *ip-address* | *hostname*][**src-port** *port-number*]] **Example:** `Device# show ip rsvp sender detail`	(Optional) Displays RSVP PATH-related sender information currently in the database. • The optional keywords and arguments display additional information. **Note** The optional **filter** keyword is supported in Cisco IOS Releases 12.0S and 12.2S only.
Step 7	**show ip rsvp installed** [*interface-type interface-number*] [**detail**] **Example:** `Device# show ip rsvp installed detail`	(Optional) Displays RSVP-related installed filters and corresponding bandwidth information. • The optional keywords and arguments display additional information.
Step 8	**show ip rsvp interface** [**detail**] [*interface-type interface-number*] **Example:** `Device# show ip rsvp interface detail`	(Optional) Displays RSVP-related interface information. • The optional keywords and arguments display additional information.
Step 9	**end** **Example:**	(Optional) Exits privileged EXEC mode and returns to user EXEC mode.

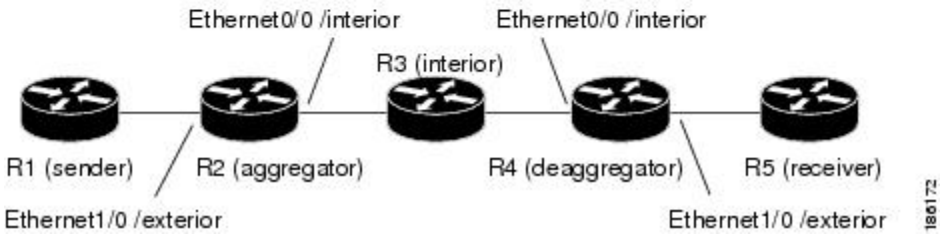
	Command or Action	Purpose
	`Device# end`	

Configuration Examples for RSVP Aggregation

Examples Configuring RSVP Aggregation

The figure below shows a five-router network in which RSVP aggregation is configured.

Figure 99: Sample RSVP Aggregation Network



Configuring RSVP and DiffServ Attributes on an Interior Router

The following example configures RSVP/DiffServ attributes on an interior router (R3 in the figure above).

- GigabitEthernet interface 0/0/0 is enabled for RSVP and the amount of bandwidth available for reservations is configured.
- A resource provider is configured and data packet classification is disabled because RSVP aggregation supports control plane aggregation only.

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface GigabitEthernet 0/0/0

Router(config-if)# ip rsvp bandwidth 400
```

```
Router(config-if)# ip rsvp resource-provider none

Router(config-if)# ip rsvp data-packet classification none

Router(config-if)# end
```

Configuring RSVP Aggregation on an Aggregator or Deaggregator

The following example configures RSVP aggregation attributes on an aggregator or deaggregator (R2 and R4 in the figure above):

- Loopback 1 is configured to establish an RSVP aggregation router ID.
- Ethernet interface 0/0 is enabled for RSVP and the amount of bandwidth available for reservations is configured.
- Ethernet interface 0/0 on an aggregator or deaggregator is configured to face an aggregation region.
- A resource provider is configured and data packet classification is disabled because RSVP aggregation supports control plane aggregation only.

```
Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.
Router(config)# interface Loopback 1
Router(config)# ip address 192.168.50.1 255.255.255.0
Router(config)# interface GigabitEthernet 0/0/0
Router(config-if)# ip rsvp bandwidth 400
Router(config-if)# ip rsvp aggregation role interior
Router(config-if)# ip rsvp resource-provider none
Router(config-if)# ip rsvp data-packet classification none
Router(config-if)# end
```

Configuring RSVP Aggregation Attributes and Parameters

The following example configures additional RSVP aggregation attributes, including a global rule for mapping all E2E reservations onto a single aggregate with DSCP AF41 and the token bucket parameters for aggregate reservations, because dynamic resizing is not supported. This configuration is only required on nodes performing the deaggregation function (R4 in the figure above).

```
Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)# ip rsvp aggregation ip map any dscp af41

Router(config)# ip rsvp aggregation ip reservation dscp af41 aggregator
10.10.10.10 traffic-params static rate 10 burst 8 peak 10

Router(config)# end
```

Configuring an Access List for a Deaggregator

In the following example, access list 1 is defined for all RSVP messages whose RSVP PATH message sender template source address is in the 10.1.0.0 subnet so that the deaggregator (R4 in the figure above) maps those reservations onto an aggregate reservation for the DSCP associated with the AF41 PHB:

```
Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)# access-list 1 permit 10.1.0.0 0.0.255.255

Router(config)# ip rsvp aggregation ip map access-list 1 dscp af41

Router(config)# end
```

Configuring RSVP Aggregation

After you configure your RSVP aggregation attributes, you are ready to enable aggregation globally.

When you enable aggregation on a router, the router can act as an aggregator or a deaggregator. To perform aggregator and deaggregator functions, the RSVP process must see messages with the RSVP-E2E-IGNORE protocol type (134) on a router; otherwise, the messages are forwarded as data by the router's data plane. The **ip rsvp aggregation ip** command enables RSVP to identify messages with the RSVP-E2E-IGNORE protocol.

Note This registers a router to receive RSVP-E2E-IGNORE messages. It is not necessary to configure this command on interior nodes that are only processing RSVP aggregate reservations and forwarding RSVP-E2E-IGNORE messages as IP datagrams). Since the router is loaded with an image that supports aggregation, the router will process aggregate (RFC 3175 formatted) messages correctly. Enabling aggregation on an interior mode may decrease performance because the interior node will then unnecessarily process all RSVP-E2E-IGNORE messages.

Note If you enable aggregation on an interior node, you must configure all its interfaces as interior. Otherwise, all the interfaces have the exterior role, and any E2E PATH (E2E-IGNORE) messages arriving at the router are discarded.

In summary, there are two options for an interior router (R3 in the figure above):

- No RSVP aggregation configuration commands are entered.
- RSVP aggregation is enabled and all interfaces are configured as interior.

Configuring RSVP Local Policy

You can configure a local policy optionally on any RSVP capable node. In this example, a local policy is configured on a deaggregator to set the preemption priority values within the RSVP RESV aggregate messages based upon matching the DSCP within the aggregate RSVP messages session object. This allows the bandwidth available for RSVP reservations to be used first by reservations of DSCP EF over DSCP AF41 on interior or

aggregation nodes. Any aggregate reservation for another DSCP will have a preemption priority of 0, the default.

Note Within the RSVP RESV aggregate message at the deaggregator, this local policy sets an RFC 3181 "Signaled Preemption Priority Policy Element" that can be used by interior nodes or the aggregator that has **ip rsvp preemption** enabled.

The following example sets the preemption priority locally for RSVP aggregate reservations during establishment on an interior router (R3 in the figure above):

```
Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)# ip rsvp policy local dscp-ip ef

Router(config-rsvp-local-policy)# 5 5

Router(config-rsvp-local-policy)# exit

Router(config)# ip rsvp policy local dscp-ip af41

Router(config-rsvp-local-policy)# 2 2

Router(config-rsvp-local-policy)# end
```

Example Verifying the RSVP Aggregation Configuration

Verifying RSVP Aggregation and Configured Reservations

The following example verifies that RSVP aggregation is enabled and displays information about the reservations currently established and configured map and reservation policies:

```
Router# show ip rsvp aggregation ip
RFC 3175 Aggregation:  Enabled
  Level: 1
  Default QoS service: Controlled-Load
  Number of signaled aggregate reservations:  2
  Number of signaled E2E reservations:        8
  Number of configured map commands:          4
  Number of configured reservation commands:  1
```

Verifying Configured Interfaces and Their Roles

The following example displays the configured interfaces and whether they are interior or exterior in regard to the aggregation region:

```
Router# show ip rsvp aggregation ip interface
```

```
Interface Name       Role
-------------------- --------
Ethernet0/0          interior
Serial2/0            exterior
Serial3/0            exterior
```

Verifying Aggregator and Deaggregator Reservations

The following example displays information about the aggregators and deaggregators when established reservations are present:

```
Router# show ip rsvp aggregation ip endpoints detail
Role  DSCP Aggregator      Deaggregator    State  Rate    Used    QBM PoolID
----- ---- --------------- --------------- ------ ------- ------- ----------
Agg   46   10.3.3.3         10.4.4.4         ESTABL 100K    100K    0x00000003
   Aggregate Reservation for the following E2E Flows (PSBs):
To             From          Pro DPort Sport  Prev Hop       I/F      BPS
10.4.4.4       10.1.1.1       UDP 1     1     10.23.20.3     Et1/0    100K
   Aggregate Reservation for the following E2E Flows (RSBs):
To             From          Pro DPort Sport  Next Hop       I/F      Fi Serv BPS
10.4.4.4       10.1.1.1       UDP 1     1     10.4.4.4       Se2/0    FF RATE 100K
   Aggregate Reservation for the following E2E Flows (Reqs):
To             From          Pro DPort Sport  Next Hop       I/F      Fi Serv BPS
10.4.4.4       10.1.1.1       UDP 1     1     10.23.20.3     Et1/0    FF RATE 100K
```

Additional References

The following sections provide references related to the RSVP Application ID Support feature.

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
QoS configuration tasks related to RSVP	"Configuring RSVP" module
Cisco United Communications Manager (CallManager) and related features	"Overview of Cisco Unified Communications Manager and Cisco IOS Interoperability" module
Regular expressions	"Using the Cisco IOS Command-Line Interface" module
Cisco IOS commands	Cisco IOS Master Commands List, All Releases

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2205	Resource ReSerVation Protocol (RSVP)
RFC 2872	Application and Sub Application Identity Policy Element for Use with RSVP
RFC 3181	Signaled Preemption Priority Policy Element
RFC 3182	Identity Representation for RSVP

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for RSVP Aggregation

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 77: Feature Information for RSVP Aggregation

Feature Name	Releases	Feature Information
RSVP Aggregation	Cico IOS XE Release 2.6 Cisco IOS XE Release 3.8S	The RSVP Aggregation feature allows the Resource Reservation Protocol (RSVP) state to be reduced within an RSVP/DiffServ network by aggregating many smaller reservations into a single, larger reservation at the edge. The following commands were introduced or modified: **debug ip rsvp aggregation**, **debug qbm**, **ip rsvp aggregation ip**, **ip rsvp aggregation ip map**, **ip rsvp aggregation**, **ip reservation dscp traffic-params static rate**, **ip rsvp aggregation ip role interior**, **ip rsvp policy local**, **show ip rsvp**, **show ip rsvp aggregation ip**, **show ip rsvp aggregation ip endpoints**, **show ip rsvp installed**, **show ip rsvp interface**, **show ip rsvp policy local**, **show ip rsvp request**, **show ip rsvp reservation**, **show ip rsvp sender**, **show qbm client**, **show qbm pool**. In Cisco IOS XE Release 3.8S, support was added for the Cisco ASR 903 Router.

Glossary

admission control --The process by which an RSVP reservation is accepted or rejected on the basis of end-to-end available network resources.

aggregate --AnRSVP flow that represents multiple end-to-end (E2E) flows; for example, a Multiprotocol Label Switching Traffic Engineering (MPLS-TE) tunnel may be an aggregate for many E2E flows.

aggregation region --An area where E2E flows are represented by aggregate flows, with aggregators and deaggregators at the edge; for example, an MPLS-TE core, where TE tunnels are aggregates for E2E flows. An aggregation region contains a connected set of nodes that are capable of performing RSVP aggregation.

aggregator --The device that processes the E2E PATH message as it enters the aggregation region. This device is also called the TE tunnel head-end device; it forwards the message from an exterior interface to an interior interface.

bandwidth --The difference between the highest and lowest frequencies available for network signals. The term is also used to describe the rated throughput capacity of a given network medium or protocol.

deaggregator --The device that processes the E2E PATH message as it leaves the aggregation region. This device is also called the TE tunnel tail-end device; it forwards the message from an interior interface to an exterior interface.

E2E --end-to-end. An RSVP flow that crosses an aggregation region, and whose state is represented in aggregate within this region, such as a classic RSVP unicast flow crossing an MPLS-TE core.

LSP --label-switched path. A configured connection between two devices, in which label switching is used to carry the packets. The purpose of an LSP is to carry data packets.

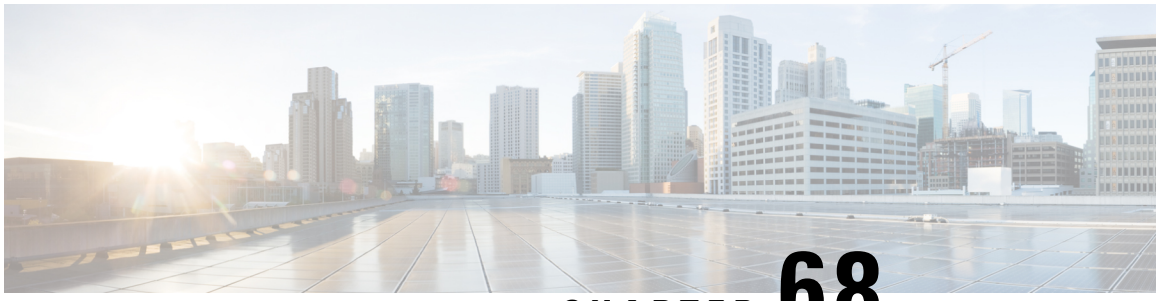
QoS --quality of service. A measure of performance for a transmission system that reflects its transmission quality and service availability.

RSVP --Resource Reservation Protocol. A protocol that supports the reservation of resources across an IP network. Applications running on IP end systems can use RSVP to indicate to other nodes the nature (bandwidth, jitter, maximum burst, and so on) of the packet streams that they want to receive.

state --Information that a device must maintain about each LSP. The information is used for rerouting tunnels.

TE --traffic engineering. The techniques and processes used to cause routed traffic to travel through the network on a path other than the one that would have been chosen if standard routing methods had been used.

tunnel --Secure communications path between two peers, such as two devices.



CHAPTER 68

RSVP Application ID Support

The RSVP Application ID Support feature introduces application-specific reservations, which enhance the granularity for local policy match criteria so that you can manage quality of service (QoS) on the basis of application type.

Prerequisites for RSVP Application ID Support

You must configure Resource Reservation Protocol (RSVP) on one or more interfaces on at least two neighboring routers that share a link within the network.

Restrictions for RSVP Application ID Support

- RSVP policies apply only to PATH, RESV, PATHERROR, and RESVERROR messages.
- Merging of global and interface-based local policies is not supported; therefore, you cannot match on multiple policies.

Information About RSVP Application ID Support

Feature Overview of RSVP Application ID Support

How RSVP Functions

Multiple applications such as voice and video need RSVP support. RSVP admits requests until the bandwidth limit is reached. RSVP does not differentiate between the requests and is not aware of the type of application for which the bandwidth is requested.

As a result, RSVP can exhaust the allowed bandwidth by admitting requests that represent just one type of application, causing all subsequent requests to be rejected because of unavailable bandwidth. For example, a few video calls could prevent all or most of the voice calls from being admitted because the video calls require a large amount of bandwidth and not enough bandwidth remains to accommodate the voice calls. With this limitation, you would probably not deploy RSVP for multiple applications especially if voice happens to be one of the applications for which RSVP is required.

The solution is to allow configuration of separate bandwidth limits for individual applications or classes of traffic. Limiting bandwidth per application requires configuring a bandwidth limit per application and having each reservation flag the application to which the reservation belongs so that it can be admitted against the appropriate bandwidth limit.

Application and Sub Application Identity Policy Element for Use with RSVP (Internet Engineering Task Force (IETF) RFC 2872) allows for creation of separate bandwidth reservation pools. For example, an RSVP reservation pool can be created for voice traffic, and a separate RSVP reservation pool can be created for video traffic. This prevents video traffic from overwhelming voice traffic.

Note Before the introduction of the RSVP Application ID Support feature, provision was made to create Access Control Lists (ACLs) that matched on the differentiated services code points (DSCPs) of the IP header in an RSVP message. However, multiple applications could use the same DSCP; therefore, you could not uniquely identify applications in order to define separate policies for them.

Sample Solution

The figure below shows a sample solution in which application ID support is used. In this example, bandwidth is allocated between the voice and video sessions that are being created by Cisco Unified Communications Manager (CUCM). Video requires much more bandwidth than voice, and if you do not separate the reservations, the video traffic could overwhelm the voice traffic.

CUCM uses the RSVP Application ID Support feature. In this example, when CUCM makes the RSVP reservation, CUCM can specify whether the reservation should be made against a video RSVP bandwidth pool or a voice RSVP bandwidth pool. If not enough bandwidth remains in the requested pool, even though there is enough bandwidth in the total RSVP allocation, RSVP signals CUCM that there is a problem with the reservation. The figure below shows some of the signaling and data traffic that is sent during the session setup.

IMAGE MISSING; embedded not referenced

In this scenario, the IP phones and IP video devices do not directly support RSVP. In order to allow RSVP to reserve the bandwidth for these devices, the RSVP agent component in the Cisco IOS router creates the reservation. While setting up the voice or video session, CUCM communicates with the RSVP agent and sends the parameters to reserve the necessary bandwidth.

When you want to make a voice or video call, the device signals CUCM. CUCM signals the RSVP agent, specifying the RSVP application ID that corresponds to the type of call, which is voice or video in this example. The RSVP agents establish the RSVP reservation across the network and communicate to CUCM that the reservation has been made. CUCM then completes the session establishment, and the Real-Time Transport Protocol (RTP) traffic streams flow between the phones (or video devices). If the RSVP agents are unable to create the bandwidth reservations for the requested application ID, they communicate that information back to CUCM, which signals this information back to you.

Global and per-Interface RSVP Policies

You can configure RSVP policies globally and on a per-interface basis. You can also configure multiple global policies and multiple policies per interface.

Global RSVP policies restrict how much RSVP bandwidth a router uses regardless of the number of interfaces. You should configure a global policy if your router has CPU restrictions, one interface, or multiple interfaces that do not require different bandwidth limits.

Per-interface RSVP policies allow you to configure separate bandwidth pools with varying limits so that no one application, such as video, can consume all the RSVP bandwidth on a specified interface at the expense of other applications, such as voice, which would be dropped. You should configure a per-interface policy when you need greater control of the available bandwidth.

How RSVP Policies Are Applied

RSVP searches for policies whenever an RSVP message is processed. The policy informs RSVP if any special handling is required for that message.

If your network configuration has global and per-interface RSVP policies, the per-interface policies are applied first; that is, the RSVP looks for policy-match criteria in the order in which the policies were configured. RSVP searches for policy-match criteria in the following order:

- Nondefault interface policies
- Default interface policy
- Nondefault global policies
- Global default policy

If RSVP finds no policy-match criteria, it accepts all incoming messages. To change this decision from accept to reject, use the **ip rsvp policy default-reject** command.

Preemption

Preemption happens when one reservation receives priority over another because there is insufficient bandwidth in an RSVP pool. There are two types of RSVP bandwidth pools: local policy pools and interface pools. Local policies can be global or interface-specific. RSVP performs admission control against these pools when a RESV message arrives.

If an incoming reservation request matches an RSVP local policy that has an RSVP bandwidth limit (as configured with the **maximum bandwidth group** submode command) and that limit has been reached, RSVP

tries to preempt other lower-priority reservations admitted by that policy. When there are too few of these lower-priority reservations, RSVP rejects the incoming reservation request. Then RSVP looks at the interface bandwidth pool that you configured by using the **ip rsvp bandwidth** command. If that bandwidth limit has been reached, RSVP tries to preempt other lower-priority reservations on that interface to accommodate the new reservation request. At this point, RSVP does not consider which local policies admitted the reservations. When not enough bandwidth on that interface pool can be preempted, RSVP rejects the new reservation even though the new reservation was able to obtain bandwidth from the local policy pool.

Preemption can also happen when you manually reconfigure an RSVP bandwidth pool of any type to a lower value such that the existing reservations using that pool no longer fit in the pool.

How Preemption Priorities Are Assigned and Signaled

If a received RSVP PATH or RESV message contains preemption priorities (signaled with an IETF RFC 3181 preemption priority policy element inside an IETF RFC 2750 POLICY_DATA object) and the priorities are higher than those contained in the matching local policy (if any), the offending message is rejected and a PATHERROR or RESVERROR message is sent in response. If the priorities are approved by the local policy, they are stored with the RSVP state in the device and forwarded to its neighbors.

If a received RSVP PATH or RESV message does not contain preemption priorities (as previously described) and you issued a global **ip rsvp policy preempt** command, and the message matches a local policy that contains a **preempt-priority** command, a POLICY_DATA object with a preemption priority element that contains the local policy's priorities is added to the message as part of the policy decision. These priorities are then stored with the RSVP state in the device and forwarded to neighbors.

Controlling Preemption

The **ip rsvp policy preempt** command controls whether a router preempts any reservations when required. When you issue this command, a RESV message that subsequently arrives on an interface can preempt the bandwidth of one or more reservations on that interface if the assigned setup priority of the new reservation is higher than the assigned hold priorities of the installed reservations.

Benefits of RSVP Application ID Support

The RSVP Application ID Support feature provides the following benefits:

- Allows RSVP to identify applications uniquely and to separate bandwidth pools to be created for different applications so that one application cannot consume all the available bandwidth, thereby forcing others to be dropped.
- Integrates with the RSVP agent and CUCM to provide a solution for call admission control (CAC) and QoS for VoIP and video conferencing applications in networks with multitiered, meshed topologies using signaling protocols such as Signaling Connection Control Part (SCCP) to ensure that a single application does not overwhelm the available reserved bandwidth.
- Functions with any endpoint that complies with RFC 2872 or RFC 2205.

How to Configure RSVP Application ID Support

You can configure application IDs and local policies to use with RSVP-aware software programs such as CUCM or to use with non-RSVP-aware applications such as static PATH and RESV messages.

Configuring RSVP Application ID for RSVP-Aware Software Programs

Configuring an RSVP Application ID

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp policy identity** *alias* **policy-locator** *locator*
4. Repeat Step 3 as needed to configure additional application IDs.
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp policy identity** *alias* **policy-locator** *locator* **Example:** `Router(config)# ip rsvp policy identity rsvp-voice policy-locator APP=Voice`	Defines RSVP application IDs to use as match criteria for local policies. • Enter a value for the *alias*argument, which is a string used within the router to reference the identity in RSVP configuration commands and show displays. The string can have as many as 64 printable characters (in the range 0x20 to 0x7E). **Note** If you use the " " or ? characters as part of the alias or locator string itself, you must type the CTRL-V key sequence before entering the embedded " " or ? character. The alias is never transmitted to other routers. • Enter a value for the *locator* argument, which is a string that is signaled in RSVP messages and contains application IDs usually in X.500 Distinguished Name (DN) format. This can also be a regular expression.
Step 4	Repeat Step 3 as needed to configure additional application IDs.	Defines additional application IDs.

	Command or Action	Purpose
Step 5	**end** **Example:** `Router(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

What to Do Next

Configure a local policy globally, or on an interface, or both.

Configuring a Local Policy Globally

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp policy local** {**acl** *acl1*[*acl2...acl8*] | **dscp-ip** *value1*[*value2...value8*]| **default** | **identity** *alias1* [*alias2...alias4*] | **origin-as** *as1*[*as2...as8*]}
4. Repeat Step 3 as needed to configure additional local policies.
5. Enter the submode commands as required.
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp policy local** {**acl** *acl1*[*acl2...acl8*]	**dscp-ip** *value1*[*value2...value8*]| **default**	**identity** *alias1* [*alias2...alias4*]	**origin-as** *as1*[*as2...as8*]} **Example:** `Router(config)# ip rsvp policy local identity rsvp-voice`	Creates a local policy to determine how RSVP resources are used in a network and enters local policy configuration mode. • Enter the **identity** *alias1* keyword and argument combination to specify an application ID alias.
Step 4	Repeat Step 3 as needed to configure additional local policies.	(Optional) Configures additional local policies.
Step 5	Enter the submode commands as required.	(Optional) Defines the properties of the local policy that you are creating.

	Command or Action	Purpose
		Note This is an optional step. An empty policy rejects everything, which may be desired in some cases. • See the **ip rsvp policy local** command for detailed information on submode commands.
Step 6	**end** **Example:** `Router(config-rsvp-policy-local)# end`	Exits local policy configuration mode and returns to privileged EXEC mode.

Configuring a Local Policy on an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / subslot / port*
4. Repeat Step 3 as needed to configure a local policy on additional interfaces.
5. **ip rsvp bandwidth** [*interface-kbps* [*single-flow-kbps*[**bc1** *kbps* | **sub-pool** *kbps*]*]*| **percent** *percent-bandwidth* [*single-flow-kbps*]]
6. Repeat Step 5 as needed to configure bandwidth for additional interfaces.
7. **ip rsvp policy local** {**acl** *acl1*[*acl2...acl8*] | **dscp-ip** *value1*[*value2...value8*]| **default** | **identity** *alias1* [*alias2...alias4*] | **origin-as** *as1*[*as2...as8*]}
8. Repeat Step 7 as needed to configure additional local policies.
9. Enter the submode commands as required.
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / subslot / port* **Example:**	Configures the interface type and number and enters interface configuration mode.

	Command or Action	Purpose
	`Router(config)# interface gigabitEthernet 0/0/0`	
Step 4	Repeat Step 3 as needed to configure a local policy on additional interfaces.	(Optional) Configures additional interfaces.
Step 5	**ip rsvp bandwidth** [*interface-kbps* [*single-flow-kbps*[**bc1** *kbps* | **sub-pool** *kbps*]*]*| **percent** *percent-bandwidth* [*single-flow-kbps*]] **Example:** `Router(config-if)# ip rsvp bandwidth 500 500`	Enables RSVP on an interface. • The optional *interface-kbps* and *single-flow-kbps* arguments specify the amount of bandwidth that can be allocated by RSVP flows or to a single flow, respectively. Values are from 1 to 1000000.
Step 6	Repeat Step 5 as needed to configure bandwidth for additional interfaces.	(Optional) Configures bandwidth for additional interfaces.
Step 7	**ip rsvp policy local** {**acl** *acl1*[*acl2...acl8*] | **dscp-ip** *value1*[*value2...value8*]| **default** | **identity** *alias1* [*alias2...alias4*] | **origin-as** *as1*[*as2...as8*]} **Example:** `Router(config-if)# ip rsvp policy local identity rsvp-voice`	Creates a local policy to determine how RSVP resources are used in a network. • Enter the **identity** *alias1* keyword argument combination to specify an application ID alias.
Step 8	Repeat Step 7 as needed to configure additional local policies.	(Optional) Configures additional local policies.
Step 9	Enter the submode commands as required.	(Optional) Defines the properties of the local policy that you are creating and enters local policy configuration mode. **Note** This is an optional step. An empty policy rejects everything, which may be desired in some cases. • See the **ip rsvp policy local**command for detailed information on submode commands.
Step 10	**end** **Example:** `Router(config-rsvp-policy-local)# end`	Exits local policy configuration mode and returns to privileged EXEC mode.

Configuring RSVP Application ID for Non-RSVP-Aware Software Programs

Configuring an Application ID

Refer to the Configuring an RSVP Application ID, on page 865.

Configuring a Static RSVP Sender with an Application ID

Perform this task to configure a static RSVP sender with an application ID to make the router proxy an RSVP PATH message containing an application ID on behalf of an RSVP-unaware sender application.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp sende r-host** *session-ip-address sender-ip-address* {*ip-protocol* |**tcp** | **udp**} *session-dest-port sender-source-port bandwidth burst-size*[**identity** *alias*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp sende r-host** *session-ip-address sender-ip-address* {*ip-protocol* |**tcp**	**udp**} *session-dest-port sender-source-port bandwidth burst-size*[**identity** *alias*] **Example:** `Router(config)# ip rsvp sender-host 10.0.0.7 10.0.0.1 udp 1 1 10 10 identity rsvp-voice`	Enables a router to simulate a host generating RSVP PATH messages. • The optional **identity** *alias* keyword and argument combination specifies an application ID alias. The string can have as many as 64 printable characters (in the range 0x20 to 0x7E). **Note** If you use the " " or ? character as part of the alias string itself, you must type the CTRL-V key sequence before entering the embedded " " or ? character. The alias is never transmitted to other routers.
Step 4	**end** **Example:** `Router(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring a Static RSVP Receiver with an Application ID

Perform this task to configure a static RSVP receiver with an application ID to make the router proxy an RSVP RESV message containing an application ID on behalf of an RSVP-unaware receiver application.

Note You can also configure a static listener to use with an application ID. If an incoming PATH message contains an application ID and/or a preemption priority value, the listener includes them in the RESV message sent in reply. See the Feature Information for RSVP Application ID Support, on page 878for more information.

Note Use the **ip rsvp reservation-host** command if the router is the destination, or the **ip rsvp reservation** command to have the router proxy on behalf of a downstream host.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. Do one of the following:
   - **ip rsvp reservation-host** *session-ip-address sender-ip-address* {*ip-protocol*| **tcp** | **udp**} *session-dest-port sender-source-port*{**ff** | **se** | **wf**} {**load** | **rate**} *bandwidth burst-size*[**identity** *alias*]
   -
   - **ip rsvp reservation** *session-ip-address sender-ip-address* {*ip-protocol* | **tcp** | **udp**} *session-dest-port sender-source-port next-hop-ip-address next-hop-interface*{**ff** | **se** | **wf**} {**load** | **rate**} *bandwidth burst-size*[**identity** *alias*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	Do one of the following: • **ip rsvp reservation-host** *session-ip-address sender-ip-address* {*ip-protocol*| **tcp**	**udp**} *session-dest-port sender-source-port*{**ff**	**se**	**wf**} {**load**	**rate**} *bandwidth burst-size*[**identity** *alias*] • • **ip rsvp reservation** *session-ip-address sender-ip-address* {*ip-protocol*	**tcp**	**udp**}	Enables a router to simulate a host generating RSVP RESV messages. • The optional **identity** *alias* keyword and argument combination specifies an application ID alias. The string can have as many as 64 printable characters (in the range 0x20 to 0x7E).

	Command or Action	Purpose

	session-dest-port sender-source-port *next-hop-ip-address next-hop-interface*{**ff** | **se** | **wf**}{**load** | **rate**} *bandwidth burst-size*[**identity** *alias*]
	Example:
	`Router(config)# ip rsvp reservation-host 10.1.1.1 10.30.1.4 udp 20 30 se load 100 60 identity rsvp-voice`
	Example:
	Example:
	`Router(config)# ip rsvp reservation 10.1.1.1 0.0.0.0 udp 20 0 172.16.4.1 Ethernet1 wf rate 350 65 identity xyz`
		Note If you use the " " or ? character as part of the alias string itself, you must type the CTRL-V key sequence before entering the embedded " " or ? character. The alias is never transmitted to other routers.
		Note Use the **ip rsvp reservation-host** command if the router is the destination or the **ip rsvp reservation** command to have the router proxy on behalf of a downstream host.

Step 4	**end** **Example:** `Router(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Verifying the RSVP Application ID Support Configuration

Note You can use the following commands in user EXEC or privileged EXEC mode, in any order.

SUMMARY STEPS

1. **enable**
2. **show ip rsvp host** {**receivers**| **senders**}[*hostname* | *group-address*]
3. **show ip rsvp policy identity** [*regular-expression*]
4. **show ip rsvp policy local** [**detail**] [**interface** *type slot* / *subslot* / *port*] [**acl** *acl-number*| **dscp-ip** *value*| **default** | **identity** *alias* | **origin-as** *as*]
5. **show ip rsvp reservation** [**detail**] [**filter** [**destination** *address*] [**dst-port** *port-number*] [**source** *address*] [**src-port** *port-number*]]
6. **show ip rsvp sender** [**detail**] [**filter** [**destination** *address*] [**dst-port** *port-number*] [**source** *address*] [**src-port** *port-number*]]
7. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	(Optional) Enables privileged EXEC mode.

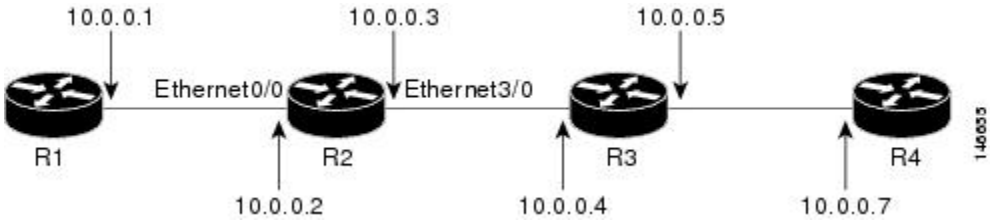
	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted. **Note** Skip this step if you are using the commands in user EXEC mode.
Step 2	**show ip rsvp host** {**receivers**| **senders**}[*hostname* | *group-address*] **Example:** `Router# show ip rsvp host senders`	Displays specific information for an RSVP host. **Note** Use this command only on routers from which PATH and RESV messages originate.
Step 3	**show ip rsvp policy identity** [*regular-expression*] **Example:** `Router# show ip rsvp policy identity voice100`	Displays selected RSVP identities in a router configuration. • The optional *regular-expression* argument allows pattern matching on the alias strings of the RSVP identities to be displayed.
Step 4	**show ip rsvp policy local** [**detail**] [**interface** *type slot* / *subslot* / *port*] [**acl** *acl-number*| **dscp-ip** *value*| **default** | **identity** *alias* | **origin-as** *as*] **Example:** `Router# show ip rsvp policy local identity voice100`	Displays the local policies currently configured. • The optional **detail** keyword and the optional **interface** *type slot* / *subslot* / *port* keyword and argument combination can be used with any of the match criteria.
Step 5	**show ip rsvp reservation** [**detail**] [**filter** [**destination** *address*] [**dst-port** *port-number*] [**source** *address*] [**src-port** *port-number*]] **Example:** `Router# show ip rsvp reservation detail`	Displays RSVP-related receiver information currently in the database. • The optional **detail** keyword displays additional output with information about where the policy originated and which application ID was signaled in the RESV message.
Step 6	**show ip rsvp sender** [**detail**] [**filter** [**destination** *address*] [**dst-port** *port-number*] [**source** *address*] [**src-port** *port-number*]] **Example:** `Router# show ip rsvp sender detail`	Displays RSVP PATH-related sender information currently in the database. • The optional **detail** keyword displays additional output with information that includes which application ID was signaled in the PATH message.
Step 7	**end** **Example:** `Router# end`	Exits privileged EXEC mode and returns to user EXEC mode.

Configuration Examples for RSVP Application ID Support

Example Configuring RSVP Application ID Support

The configurations for four-router network shown in the figure below are in the following sections:

Figure 100: Sample Network with Application Identities and Local Policies



Configuring a Proxy Receiver on R4

The following example configures R4 with a proxy receiver to create an RESV message to match the PATH message for the destination 10.0.0.7:

```
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# ip rsvp listener 10.0.0.7 any any reply

Device(config)# end
```

Configuring an Application ID and a Global Local Policy on R3

The following example configures R3 with an application ID called video and a global local policy in which all RSVP messages are being accepted and forwarded:

```
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# ip rsvp policy identity video policy-locator video
Device(config)# ip rsvp policy local identity video
Device(config-rsvp-policy-local)# forward all
Device(config-rsvp-policy-local)# end
```

Configuring an Application ID and Separate Bandwidth Pools on R2 for per-Interface Local Policies

The following example configures R2 with an application ID called video, which is a wildcard regular expression to match any application ID that contains the substring video:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# ip rsvp policy identity video policy-locator .*Video.*
Router(config-rsvp-id)# end
```

The following example configures R2 with a local policy on ingress Gigabit Ethernet interface 0/0/0:

```
Router# configure terminal
```

```
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface gigabitEthernet 0/0/0
Router(config-if)# ip address 10.0.0.2 255.0.0.0
Router(config-if)# no cdp enable
Router(config-if)# ip rsvp bandwidth 200
Router(config-if)# ip rsvp policy local identity video
Router(config-rsvp-policy-local)# maximum senders 10
Router(config-rsvp-policy-local)# maximum bandwidth group 100
Router(config-rsvp-policy-local)# maximum bandwidth single 10
Router(config-rsvp-policy-local)# forward all
Router(config-rsvp-policy-local)# end
```

The following example configures R2 with a local policy on egress Gigabit Ethernet interface 3/0/0:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface gigabitEthernet 3/0/0
Router(config-if)# ip address 10.0.0.3 255.0.0.0
Router(config-if)# no cdp enable
Router(config-if)# ip rsvp bandwidth 200
Router(config-if)# ip rsvp policy local identity video
Router(config-rsvp-policy-local)# maximum senders 10
Router(config-rsvp-policy-local)# maximum bandwidth group 100
Router(config-rsvp-policy-local)# maximum bandwidth single 10
Router(config-rsvp-policy-local)# forward all
Router(config-rsvp-policy-local)# end
```

Note PATH messages arrive on ingress Gigabit Ethernet interface 0/0/0 and RESV messages arrive on egress Gigabit Ethernet interface 3/0/0.

Configuring an Application ID and a Static Reservation from R1 to R4

The following example configures R1 with an application ID called video and initiates a host generating a PATH message with that application ID:

```
Device# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Device(config)# ip rsvp policy identity video policy-locator "GUID=www.cisco.com, APP=Video,
 VER=1.0"
Device(config)# ip rsvp sender-host 10.0.0.7 10.0.0.1 udp 1 1 10 10 identity video
Device(config)# end
```

Example Verifying RSVP Application ID Support

Verifying the Application ID and the Global Local Policy on R3

The following example verifies that a global local policy has been configured on R3 with an application ID called Video:

```
Router# show ip rsvp policy local detail
Global:
  Policy for ID(s): Video
    Preemption Scope: Unrestricted.
    Local Override:   Disabled.
```

```
    Fast ReRoute:     Accept.
    Handle:           23000404.
                           Accept               Forward
    Path:                  Yes                  Yes
    Resv:                  Yes                  Yes
    PathError:             Yes                  Yes
    ResvError:             Yes                  Yes
                           Setup Priority       Hold Priority
    TE:                    N/A                  N/A
    Non-TE:                N/A                  N/A
                           Current              Limit
    Senders:               1                    N/A
    Receivers:             1                    N/A
    Conversations:         1                    N/A
    Group bandwidth (bps): 10K                  N/A
    Per-flow b/w (bps):    N/A                  N/A


Generic policy settings:
    Default policy: Accept all
    Preemption:     Disabled
```

Verifying the Application ID and the per-Interface Local Policies on R2

The following example verifies that an application ID called Video has been created on R2:

```
Router# show ip rsvp policy identity
Alias: Video
  Type:    Application ID
  Locator: .*Video.*
```

The following example verifies that per-interface local policies have been created on Gigabit Ethernet interface 0/0/0 and Gigabit Ethernet interface 3/0/0 on R2:

```
Router# show ip rsvp policy local detail
gigabitEthernet 0/0/0:
  Policy for ID(s): Video
    Preemption Scope: Unrestricted.
    Local Override:   Disabled.
    Fast ReRoute:     Accept.
    Handle:           26000404.
                           Accept               Forward
    Path:                  Yes                  Yes
    Resv:                  Yes                  Yes
    PathError:             Yes                  Yes
    ResvError:             Yes                  Yes
                           Setup Priority       Hold Priority
    TE:                    N/A                  N/A
    Non-TE:                N/A                  N/A
                           Current              Limit
    Senders:               1                    10
    Receivers:             0                    N/A
    Conversations:         0                    N/A
    Group bandwidth (bps): 0                    100K
    Per-flow b/w (bps):    N/A                  10K

giabitEthernet 3/0/0:
  Policy for ID(s): Video
    Preemption Scope: Unrestricted.
    Local Override:   Disabled.
    Fast ReRoute:     Accept.
    Handle:           5A00040A.
```

```
                          Accept               Forward
    Path:                 Yes                  Yes
    Resv:                 Yes                  Yes
    PathError:            Yes                  Yes
    ResvError:            Yes                  Yes
                          Setup Priority       Hold Priority
    TE:                   N/A                  N/A
    Non-TE:               N/A                  N/A
                          Current              Limit
    Senders:              0                    10
    Receivers:            1                    N/A
    Conversations:        1                    N/A
    Group bandwidth (bps): 10K                 100K
    Per-flow b/w (bps):   N/A                  10K
Generic policy settings:
    Default policy: Accept all
    Preemption:     Disabled
```

Note Notice in the display that the ingress interface has only its senders counter incremented because the PATH message is checked there. However, the egress interface has its receivers, conversations, and group bandwidth counters incremented because the reservation is checked on the incoming interface, which is the egress interface on R2.

Verifying the Application ID and the Reservation on R1

The following example verifies that a PATH message containing the application ID called Video has been created on R1:

```
Router# show ip rsvp sender detail
PATH Session address: 10.0.0.7, port: 1. Protocol: UDP
  Sender address: 10.0.0.1, port: 1
     Inbound from: 10.0.0.1 on interface:
  Traffic params - Rate: 10K bits/sec, Max. burst: 10K bytes
                   Min Policed Unit: 0 bytes, Max Pkt Size 4294967295 bytes
  Path ID handle: 02000402.
  Incoming policy: Accepted. Policy source(s): Default
    Application ID: 'GUID=www.cisco.com, APP=Video, VER=1.0'
  Status: Proxied
  Output on gigabitEthernet 0/0/0. Policy status: Forwarding. Handle: 01000403
    Policy source(s): Default
```

Note You can use the **debug ip rsvp dump path** and the **debug ip rsvp dump resv** commands to get more information about a sender and the application ID that it is using.

The following example verifies that a reservation with the application ID called Video has been created on R1:

```
Router# show ip rsvp reservation detail

RSVP Reservation. Destination is 10.0.0.7, Source is 10.0.0.1,
  Protocol is UDP, Destination port is 1, Source port is 1
  Next Hop is 10.0.0.2, Interface is gigabitEthernet 0/0/0
  Reservation Style is Fixed-Filter, QoS Service is Guaranteed-Rate
  Resv ID handle: 01000405.
```

```
Created: 10:07:35 EST Thu Jan 12 2006
Average Bitrate is 10K bits/sec, Maximum Burst is 10K bytes
Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
Status:
Policy: Forwarding. Policy source(s): Default
  Application ID: 'GUID=www.cisco.com, APP=Video, VER=1.0'
```

Additional References

The following sections provide references related to the RSVP Application ID Support feature.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
QoS configuration tasks related to RSVP	"Configuring RSVP" module
Cisco United Communications Manager (CallManager) and related features	"Overview of Cisco Unified Communications Manager and Cisco IOS Interoperability" module
Regular expressions	"Using the Cisco IOS Command-Line Interface" module

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2205	*Resource ReSerVation Protocol (RSVP)*

RFC	Title
RFC 2872	*Application and Sub Application Identity Policy Element for Use with RSVP*
RFC 3181	*Signaled Preemption Priority Policy Element*
RFC 3182	*Identity Representation for RSVP*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for RSVP Application ID Support

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 78: Feature Information for RSVP Application ID Support

Feature Name	Releases	Feature Information
RSVP Application ID Support	Cisco IOS XE Release 2.6 Cisco IOS XE Release 3.8S	The RSVP Application ID Support feature introduces application-specific reservations, which enhance the granularity for local policy-match criteria so that you can manage QoS on the basis of application type. The following commands were introduced or modified: **ip rsvp listener**, **ip rsvp policy identity**, **ip rsvp policy local**, **ip rsvp reservation**, **ip rsvp reservation-host**, **ip rsvp sender**, **ip rsvp sender-host**, **maximum**(local policy), **show ip rsvp host**, **show ip rsvp policy identity**, **show ip rsvp policy local**. In Cisco IOS XE Release 3.8S, support was added for the Cisco ASR 903 Router.

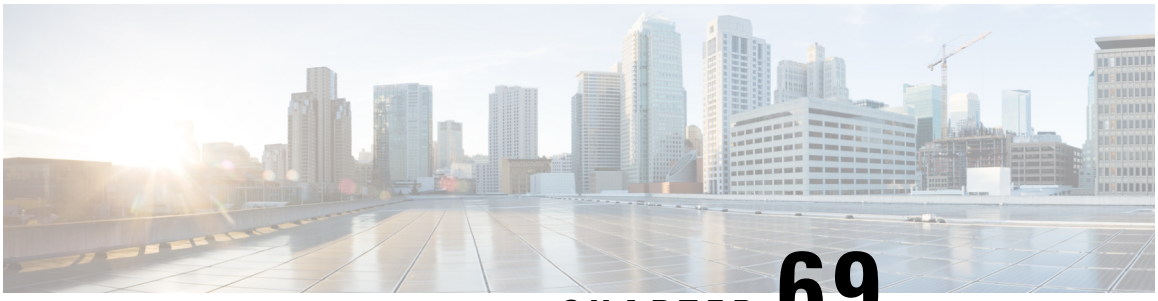
Glossary

QoS --quality of service. A measure of performance for a transmission system that reflects its transmission quality and service availability.

RSVP --Resource Reservation Protocol. A protocol that supports the reservation of resources across an IP network. Applications running on IP end systems can use RSVP to indicate to other nodes the nature (bandwidth, jitter, maximum burst, and so on) of the packet streams that they want to receive.

RSVP Agent --Implements a Resource Reservation Protocol (RSVP) agent on Cisco IOS voice gateways that support Unified CM.

Unified Communcations Manager (CM)--The software-based, call-processing component of the Cisco IP telephony solution. The software extends enterprise telephony features and functions to packet telephony network devices such as IP phones, media processing devices, voice-over-IP (VoIP) gateways, and multimedia applications.



CHAPTER 69

RSVP Fast Local Repair

The RSVP Fast Local Repair feature provides quick adaptation to routing changes occurring in global and Virtual Routing and Forwarding (VRF) domains, without the overhead of the refresh period to guarantee the quality of service (QoS) for data flows. With fast local repair (FLR), Resource Reservation Protocol (RSVP) speeds up its response to routing changes from 30 seconds to a few seconds.

Prerequisites for RSVP FLR

You must configure RSVP on one or more interfaces on at least two neighboring devices that share a link within the network.

Restrictions for RSVP FLR

- RSVP FLR applies only when RSVP is used to set up resource reservations for IPv4 unicast flows; IPv4 multicast flows are not supported.
- RSVP FLR does not apply to traffic engineering (TE) tunnels and, therefore, does not affect TE sessions.
- RSVP FLR does not support message bundling.

Information About RSVP FLR

Feature Overview of RSVP FLR

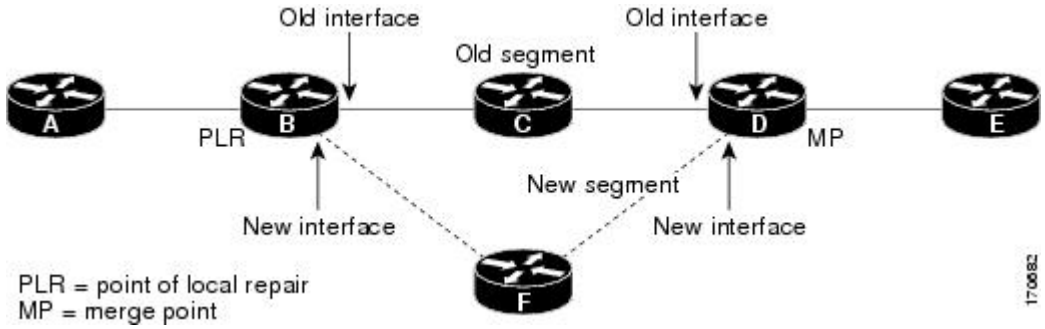
RSVP FLR provides for dynamic adaptation when routing changes occur in global or VRF routing domains. When a route changes, the next PATH and RESV message refreshes establish path and reservation states along the new route. Depending on the configured refresh interval, this reroute happens in tens of seconds. However, during this time, the QoS of flows is not guaranteed because congestion may occur while data packets travel over links where reservations are not yet in place.

In order to provide faster adaptation to routing changes, without the overhead of a refresh period, RSVP registers with the Routing Information Base (RIB) and receives notifications when routes change, thereby triggering state refreshes for the affected destinations. These triggered refreshes use the new route information and, as a result, install reservations over the new path.

When routes change, RSVP has to reroute all affected paths and reservations. Without FLR, the reroute happens when refresh timers expire for the path states. With real-time applications such as VoIP and video on demand (VoD), the requirement changes and the reroute must happen, within three seconds from the triggering event such as link down or link up.

The figure below illustrates the FLR process.

Figure 101: Overview of RSVP FLR



Initial RSVP states are installed for an IPv4 unicast flow over Routers A, B, C, D, and E. Router A is the source or headend, and Router E is the destination or tailend. The data packets are destined to an address of Router E. Assume that a route change occurs, and the new path taken by the data packets is from Router A to Router B to Router F to Router D to Router E; therefore, the old and new paths differ on the segments between Routers B and D. The Router B to Router C to Router D segment is the old segment, and the Router B to Router F to Router D segment is the new segment.

A route may change because of a link or node failure, or if a better path becomes available.

RSVP at Router B detects that the route change affects the RSVP flow and initiates the FLR procedure. The node that initiates an FLR repair procedure, Router B in the figure above, is the point of local repair (PLR). The node where the new and old segments meet, Router D in the figure above, is the merge point (MP). The interfaces at the PLR and the MP that are part of the old segment are the old interfaces, and the interfaces that are part of the new segment are the new interfaces.

If a route has changed because of a failure, the PLR may not be the node that detects the failure. For example, it is possible that the link from Router C to Router D fails, and although Router C detects the failure, the route

change at Router B is the trigger for the FLR procedure. Router C, in this case, is also referred to as the node that detects the failure.

The support for FLR in VRF domains means that RSVP can get a route change notification, even if there is a route change in any VRF domains, because RSVP FLR was previously supported only in the global routing domain.

Benefits of RSVP FLR

Faster Response Time to Routing Changes

FLR reduces the time that it takes for RSVP to determine that a physical link has gone down and that the data packets have been rerouted. Without FLR, RSVP may not recognize the link failure for 30 seconds when all of the sessions are impacted by having too much traffic for the available bandwidth. With FLR, this time can be significantly reduced to a few seconds.

After detecting the failure, RSVP recomputes the admission control across the new link. If the rerouted traffic fits on the new link, RSVP reserves the bandwidth and guarantees the QoS of the new traffic.

If admission control fails on the new route, RSVP does not explicate tear down the flow, but instead sends a RESVERROR message toward the receiver. If a proxy receiver is running, then RSVP sends a PATHERROR message toward the headend, in response to the RESVERROR message, indicating the admission failure. In both cases, with and without a proxy receiver, the application tears down the failed session either at the headend or at the final destination.

Until this happens, the data packets belonging to this session still flow over the rerouted segment although admission has failed and QoS is affected.

The support of FLR in VRF domains means that if there is a route change in any routing domain, RSVP can use FLR to adapt to the routing change, because RSVP FLR was previously supported only in the global routing domain.

How to Configure RSVP FLR

You can configure the RSVP FLR parameters in any order that you want.

Configuring the RSVP FLR Wait Time

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / subslot / port*
4. **ip rsvp bandwidth** [*interface-kbps* [*single-flow-kbps*[**bc1** *kbps* | **sub-pool** *kbps*]*]*| **percent** *percent-bandwidth* [*single-flow-kbps*]]
5. **ip rsvp signalling fast-local-repair wait-time** *interval*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / subslot / port* **Example:** `Router(config)# interface gigabitEthernet 0/0/0`	Configures the interface type and enters interface configuration mode.
Step 4	**ip rsvp bandwidth** [*interface-kbps* [*single-flow-kbps*[**bc1** *kbps* | **sub-pool** *kbps*]]| **percent** *percent-bandwidth* [*single-flow-kbps*]] **Example:** `Router(config-if)# ip rsvp bandwidth 7500 7500`	Enables RSVP on an interface. • The optional *interface-kbps* and *single-flow-kbps* arguments specify the amount of bandwidth that can be allocated by RSVP flows or to a single flow, respectively. Values are from 1 to 10000000. • The optional **sub-pool**and *kbps*keyword and argument specify subpool traffic and the amount of bandwidth that can be allocated by RSVP flows. Values are from 1 to 10000000. **Note** Repeat this command for each interface on which you want to enable RSVP.
Step 5	**ip rsvp signalling fast-local-repair wait-time** *interval* **Example:** `Router(config-if)# ip rsvp signalling fast-local-repair wait-time 100`	Configures the delay that RSVP uses before starting an FLR procedure. • Values for the *interval* argument are 1 to 2500 milliseconds (ms); the default is 0.
Step 6	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Configuring the RSVP FLR Repair Rate

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp signalling fast-local-repair rate** rate
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp signalling fast-local-repair rate** rate **Example:** `Router(config)# ip rsvp signalling fast-local-repair rate 100`	Configures the repair rate that RSVP uses for an FLR procedure. • Values for the *rate* argument are 1 to 2500 messages per second; the default is 400.
Step 4	**exit** **Example:** `Router(config)# exit`	(Optional) Returns to privileged EXEC mode.

Configuring the RSVP FLR Notifications

Perform this task to configure the number of RSVP FLR notifications.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp signalling fast-local-repair notifications** *number*
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp signalling fast-local-repair notifications** *number* **Example:** `Router(config)# ip rsvp signalling fast-local-repair notifications 100`	Configures the number of per flow notifications that RSVP processes during an FLR procedure before it suspends. • Values for the *number*argument are 10 to 10000; the default is 1000.
Step 4	**exit** **Example:** `Router(config)# exit`	(Optional) Returns to privileged EXEC mode.

Verifying the RSVP FLR Configuration

Perform this task to verify the RSVP FLR configuration. You can use these commands in any order.

Note You can use the following **show** commands in user EXEC or privileged EXEC mode.

SUMMARY STEPS

1. **enable**
2. **show ip rsvp signalling fast-local-repair** [**statistics** [**detail**]]
3. **show ip rsvp interface** [**detail**] [*interface-type interface-number*]
4. **show ip rsvp**
5. **show ip rsvp sender** [**detail**] [**filter** [**destination** *ip-address* | *hostname*] [**dst-port** *port-number*] [**source** *ip-address* | *hostname*] [**src-port** *port-number*]]
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	(Optional) Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted. **Note** Omit this step if you are using the **show** commands in user EXEC mode.
Step 2	**show ip rsvp signalling fast-local-repair** [**statistics** [**detail**]] **Example:** `Router# show ip rsvp signalling fast-local-repair statistics detail`	Displays FLR-specific information that RSVP maintains. • The optional **statistics** and **detail** keywords display additional information about the FLR parameters.
Step 3	**show ip rsvp interface** [**detail**] [*interface-type interface-number*] **Example:** `Router# show ip rsvp interface gigabitethernet 0/0/0`	Displays RSVP-related information. • The optional **detail** keyword displays additional information including FLR parameters.
Step 4	**show ip rsvp** **Example:** `Router# show ip rsvp`	Displays general RSVP-related information.
Step 5	**show ip rsvp sender** [**detail**] [**filter** [**destination** *ip-address* | *hostname*] [**dst-port** *port-number*] [**source** *ip-address* | *hostname*] [**src-port** *port-number*]] **Example:** `Router# show ip rsvp sender detail`	Displays RSVP PATH-related sender information currently in the database. • The optional **detail** keyword displays additional output including the FLR parameters.
Step 6	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode and returns to user EXEC mode.

Configuration Examples for RSVP FLR

Example Configuring RSVP FLR

The configuration options for RSVP FLR are the following:

- Wait time
- Number of notifications

- Repair rate

Note You can configure these options in any order.

Configuring the Wait Time

The following example configures gigabitEthernet interface 0/0/0 with a bandwidth of 200 kbps and a wait time of 1000 ms:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface gigabitethernet 0/0/0
Router(config-if)# ip rsvp bandwidth 200
Router(config-if)# ip rsvp signalling fast-local-repair wait-time 1000
Router(config-if)# end
```

Configuring the Number of Notifications

The following example configures the number of flows that are repaired before suspending to 100:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# ip rsvp signalling fast-local-repair notifications 100
Router(config)# exit
```

Configuring the Repair Rate

The following example configures a repair rate of 100 messages per second:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# ip rsvp signalling fast-local-repair rate 100
Router(config)# exit
```

Example Verifying the RSVP FLR Configuration

Verifying the Details for FLR Procedures

The following example displays detailed information about FLR procedures:

```
Router# show ip rsvp signalling fast-local-repair statistics detail
Fast Local Repair: enabled
  Max repair rate (paths/sec): 10
  Max processed   (paths/run): 10
FLR Statistics:
  FLR 1: DONE
    Start Time: 05:18:54 IST Mon Nov 5 2007
    Number of PSBs repaired:          2
    Used Repair Rate (msgs/sec):      10
    RIB notification processing time: 0(us).
    Time of last PSB refresh:         5025(ms).
    Time of last Resv received:       6086(ms).
```

```
    Time of last Perr received:       0(us).
    Suspend count: 0
    FLR Pacing Unit: 100 msec.
    Affected neighbors:
      Nbr Address   Interface    Relative Delay Values (msec)      VRF
      10.1.2.12         Et0/3          [5000  ,..., 5000  ]        vrf1
      10.1.2.12         Et1/3          [5000  ,..., 5000  ]        vrf2
```

Verifying Configuration Details for a Specific Interface

The following example from the **show ip rsvp interface detail** command displays detailed information, including FLR, for the gigabitEthernet 0/0/0 interface:

```
Router# show ip rsvp interface detail gigabitethernet 0/0/0
  Et1/0:
    RSVP: Enabled
    Interface State: Up
    Bandwidth:
      Curr allocated: 9K bits/sec
      Max. allowed (total): 300K bits/sec
      Max. allowed (per flow): 300K bits/sec
      Max. allowed for LSP tunnels using sub-pools (pool 1): 0 bits/sec
      Set aside by policy (total): 0 bits/sec
    Traffic Control:
      RSVP Data Packet Classification is ON via CEF callbacks
    Signalling:
      DSCP value used in RSVP msgs: 0x30
      Number of refresh intervals to enforce blockade state: 4
    FLR Wait Time (IPv4 flows):
      Repair is delayed by 1000 msec.
    Authentication: disabled
      Key chain:   <none>
      Type:        md5
      Window size: 1
      Challenge:   disabled
    Hello Extension:
      State: Disabled
```

Verifying Configuration Details Before During and After an FLR Procedure

The following is sample output from the **showiprsvpsenderdetail** command before an FLR procedure has occurred:

```
Router# show ip rsvp sender detail
PATH:
   Destination 192.168.101.21, Protocol_Id 17, Don't Police , DstPort 1
   Sender address: 10.10.10.10, port: 1
   Path refreshes:
     arriving: from PHOP 172.3.31.34 on Et0/0 every 30000 msecs
   Traffic params - Rate: 9K bits/sec, Max. burst: 9K bytes
     Min Policed Unit: 0 bytes, Max Pkt Size 2147483647 bytes
   Path ID handle: 01000401.
   Incoming policy: Accepted. Policy source(s): Default
   Status:
   Output on gigabitEthernet 0/0/0. Policy status: Forwarding. Handle: 02000400
     Policy source(s): Default
   Path FLR: Never repaired
```

The following is sample output from the **showiprsvpsenderdetail** command at the PLR during an FLR procedure:

```
Router# show ip rsvp sender detail
PATH:
   Destination 192.168.101.21, Protocol_Id 17, Don't Police , DstPort 1
   Sender address: 10.10.10.10, port: 1
   Path refreshes:
     arriving: from PHOP 172.16.31.34 on Et0/0 every 30000 msecs
   Traffic params - Rate: 9K bits/sec, Max. burst: 9K bytes
     Min Policed Unit: 0 bytes, Max Pkt Size 2147483647 bytes
   Path ID handle: 01000401.
   Incoming policy: Accepted. Policy source(s): Default
   Status:
   Path FLR: PSB is currently being repaired...try later
   PLR - Old Segments: 1
    Output on Ethernet1/0, nhop 172.5.36.34
    Time before expiry: 2 refreshes
    Policy status: Forwarding. Handle: 02000400
       Policy source(s): Default
```

The following is sample output from the **showiprsvpsenderdetail** command at the MP during an FLR procedure:

```
Router# show ip rsvp sender detail
PATH:
   Destination 192.168.101.21, Protocol_Id 17, Don't Police , DstPort 1
   Sender address: 10.10.10.10, port: 1
   Path refreshes:
     arriving: from PHOP 172.16.37.35 on Et1/0 every 30000 msecs
Traffic params - Rate: 9K bits/sec, Max. burst: 9K bytes
     Min Policed Unit: 0 bytes, Max Pkt Size 2147483647 bytes
   Path ID handle: 09000406.
   Incoming policy: Accepted. Policy source(s): Default
   Status: Proxy-terminated
   Path FLR: Never repaired
   MP - Old Segments: 1
    Input on Serial2/0, phop 172.16.36.35
    Time before expiry: 9 refreshes
```

The following is sample output from the **showiprsvpsenderdetail** command at the PLR after an FLR procedure:

```
Router# show ip rsvp sender detail
PATH:
   Destination 192.168.101.21, Protocol_Id 17, Don't Police , DstPort 1
   Sender address: 10.10.10.10, port: 1
   Path refreshes:
     arriving: from PHOP 172.16.31.34 on Et0/0 every 30000 msecs
   Traffic params - Rate: 9K bits/sec, Max. burst: 9K bytes
     Min Policed Unit: 0 bytes, Max Pkt Size 2147483647 bytes
   Path ID handle: 05000401.
   Incoming policy: Accepted. Policy source(s): Default
   Status:
   Output on Serial3/0. Policy status: Forwarding. Handle: 3B000406
     Policy source(s): Default
   Path FLR: Started 12:56:16 EST Thu Nov 16 2006, PSB repaired 532(ms) after.
          Resv/Perr: Received 992(ms) after.
```

Additional References

The following sections provide references related to the Control Plane DSCP Support for RSVP feature.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
RSVP Commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Quality of service overview	"Quality of Service Overview" module

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
RFC 2206 (RSVP Management Information Base using SMIv2)	To locate and download MIBs for selected platforms, software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2205	*Resource Reservation Protocol*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for RSVP FLR

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 79: Feature Information for RSVP FLR

Feature Name	Releases	Feature Information
RSVP Fast Local Repair	Cisco IOS XE Release 2.6 Cisco IOS XE Release 3.8S	The RSVP Fast Local Repair feature provides quick adaptation to routing changes without the overhead of the refresh period to guarantee QoS for data flows. With FLR, RSVP speeds up its response to routing changes from 30 seconds to a few seconds. The following commands were introduced or modified: **clear ip rsvp signalling fast-local-repair statistics**, **ip rsvp signalling fast-local-repair notifications**, **ip rsvp signalling fast-local-repair rate**, **ip rsvp signalling fast-local-repair wait-time**, **show ip rsvp**, **show ip rsvp interface**, **show ip rsvp sender**, **show ip rsvp signalling fast-local-repair**. In Cisco IOS XE Release 3.8S, support was added for the Cisco ASR 903 Router.

Glossary

admission control --The process by which an RSVP reservation is accepted or rejected on the basis of end-to-end available network resources.

bandwidth --The difference between the highest and lowest frequencies available for network signals. The term is also used to describe the rated throughput capacity of a given network medium or protocol.

message pacing-- A system for managing volume and timing that permits messages from multiple sources to be spaced apart over time. RSVP message pacing maintains, on an outgoing basis, a count of the messages that it has been forced to drop because the output queue for the interface used for the message pacing was full.

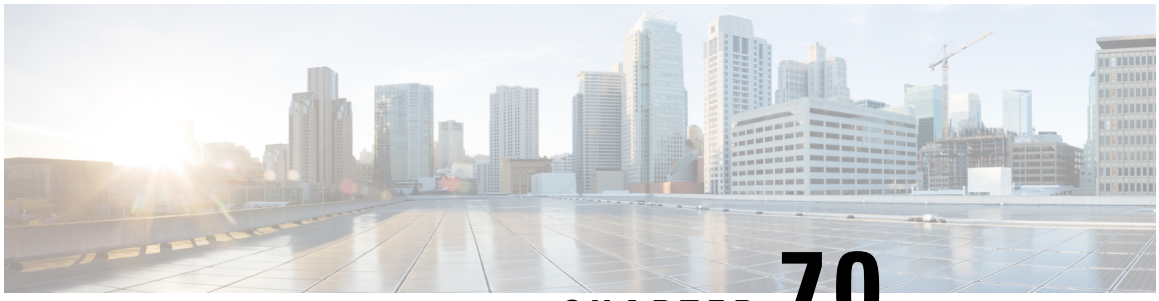
MP --merge point. The node where the new and old FLR segments meet.

PLR --point of local repair. The node that initiates an FLR procedure.

QoS --quality of service. A measure of performance for a transmission system that reflects its transmission quality and service availability.

RSVP --Resource Reservation Protocol. A protocol that supports the reservation of resources across an IP network. Applications running on IP end systems can use RSVP to indicate to other nodes the nature (bandwidth, jitter, maximum burst, and so on) of the packet streams that they want to receive.

VRF--virtual routing and forwarding. VRF is a VPN routing and forwarding instance. A VRF consists of an IP routing table, a derived forwarding table, a set of interfaces that use the forwarding table, and a set of rules and routing protocols that determine what goes into the forwarding table. In general, a VRF includes the routing information that defines a customer VPN site that is attached to a PE router.



CHAPTER 70

RSVP Interface-Based Receiver Proxy

The RSVP Interface-Based Receiver Proxy feature lets you configure a proxy router by outbound interface instead of configuring a destination address for each flow going through the same interface.

Prerequisites for RSVP Interface-Based Receiver Proxy

You must configure an IP address and enable Resource Reservation Protocol (RSVP) on one or more interfaces on at least two neighboring routers that share a link within the network.

Restrictions for RSVP Interface-Based Receiver Proxy

- Filtering using access control lists (ACLs), application IDs, or other mechanisms is not supported.
- A provider edge (PE) router cannot switch from being a proxy node to a transit node for a given flow during the lifetime of the flow.

Information About RSVP Interface-Based Receiver Proxy

Feature Overview of RSVP Interface-Based Receiver Proxy

The RSVP Interface-Based Receiver Proxy feature allows you to use RSVP to signal reservations and guarantee bandwidth on behalf of a receiver that does not support RSVP by terminating the PATH message and generating a RESV message in the upstream direction on an RSVP-capable router on the path to the endpoint. An example

is a video-on-demand flow from a video server to a set-top box, which is a computer that acts as a receiver and decodes the incoming video signal from the video server.

Because set-top boxes may not support RSVP natively, you cannot configure end-to-end RSVP reservations between a video server and a set-top box. Instead, you can enable the RSVP interface-based receiver proxy on the router that is closest to that set-top box.

The router terminates the end-to-end sessions for many set-top boxes and performs admission control on the outbound (or egress) interface of the PATH message, where the receiver proxy is configured, as a proxy for Call Admission Control (CAC) on the router-to-set-top link. The RSVP interface-based receiver proxy determines which PATH messages to terminate by looking at the outbound interface to be used by the traffic flow.

You can configure an RSVP interface-based receiver proxy to terminate PATH messages going out a specified interface with a specific action (reply with RESV, or reject). The most common application is to configure the receiver proxy on the edge of an administrative domain on interdomain interfaces. The router then terminates PATH messages going out the administrative domain while still permitting PATH messages transitioning through the router within the same administrative domain to continue downstream.

The router terminates the end-to-end sessions for many set-top boxes, with the assumption that the links further downstream (for example, from the DSLAM to the set-top box) never become congested or, more likely, in the case of congestion, that the voice and video traffic from the router gets the highest priority and access to the bandwidth.

Benefits of RSVP Interface-Based Receiver Proxy

Before the RSVP Interface-Based Receiver Proxy feature was introduced, you had to configure a receiver proxy for every separate RSVP stream or set-top box. The RSVP Interface-Based Receiver Proxy feature allows you to configure the proxy by outbound interface. For example, if there were 100 set-top boxes downstream from the proxy router, you had to configure 100 proxies. With this enhancement, you configure only the outbound interfaces. In addition, the receiver proxy is guaranteed to terminate the reservation only on the last hop within the core network. Nodes that may function as transit nodes for some PATH messages but should proxy others depending on their placement in the network can perform the correct functions on a flow-by-flow basis.

How to Configure RSVP Interface-Based Receiver Proxy

Enabling RSVP on an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. interface *type number*
4. **ip rsvp bandwidth** [*interface-kbps*][*single-flow-kbps*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	interface *type number* **Example:** `Device(config)# interface Ethernet0/0`	Configures the interface type and enters interface configuration mode.
Step 4	**ip rsvp bandwidth** [*interface-kbps*][*single-flow-kbps*] **Example:** `Device(config-if)# ip rsvp bandwidth 7500`	Enables RSVP bandwidth on an interface. • The optional *interface-kbps* and *single-flow-kbps* arguments specify the amount of bandwidth that can be allocated by RSVP flows or to a single flow, respectively. Values are from 1 to 10000000. **Note** Repeat this command for each interface that you want to enable.
Step 5	**end** **Example:** `Device(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Configuring a Receiver Proxy on an Outbound Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface type** *slot* / *subslot* / *port*
4. **ip rsvp listener outbound** {**reply** | **reject**}
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface type** *slot* / *subslot* / *port* **Example:** `Router(config)# interface gigabitEthernet 0/0/0`	Configures the interface type and enters interface configuration mode.
Step 4	**ip rsvp listener outbound** {**reply** | **reject**} **Example:** `Router(config-if)# ip rsvp listener outbound reject`	Configures an RSVP router to listen for PATH messages sent through a specified interface. • Enter the **reply** keyword or the **reject** keyword to specify the response that you want to PATH messages.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Verifying the RSVP Interface-Based Receiver Proxy Configuration

Perform the following task to verify the configuration. You can use these commands in any order.

Note You can use the following **show** commands in user EXEC or privileged EXEC mode.

SUMMARY STEPS

1. **enable**
2. **show ip rsvp listeners** [*ip-address*| **any**] [**udp** | **tcp** | **any** | *protocol*][*dst-port* | **any**]
3. **show ip rsvp sender** [**detail**] [**filter** [**destination** *address*] [**dst-port** *port-number*] [**source** *address*] [**src-port** *port-number*]]
4. **show ip rsvp reservation** [**detail**] [**filter** [**destination** *address*] [**dst-port** *port-number*] [**source** *address*] [**src-port** *port-number*]]
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	(Optional) Enables privileged EXEC mode.

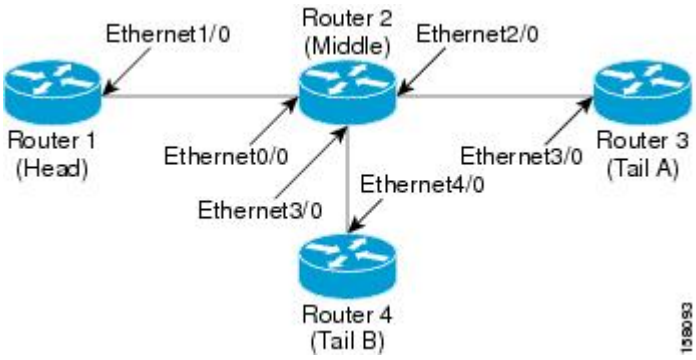
	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted. **Note** Omit this step if you are using the **show** commands in user EXEC mode.
Step 2	**show ip rsvp listeners** [*ip-address*| **any**] [**udp** | **tcp** | **any** | *protocol*][*dst-port* | **any**] **Example:** `Router# show ip rsvp listeners`	Displays RSVP listeners for a specified port or protocol.
Step 3	**show ip rsvp sender** [**detail**] [**filter** [**destination** *address*] [**dst-port** *port-number*] [**source** *address*] [**src-port** *port-number*]] **Example:** `Router# show ip rsvp sender detail`	Displays RSVP PATH-related sender information currently in the database.
Step 4	**show ip rsvp reservation** [**detail**] [**filter** [**destination** *address*] [**dst-port** *port-number*] [**source** *address*] [**src-port** *port-number*]] **Example:** `Router# show ip rsvp reservation detail`	Displays RSVP-related receiver information currently in the database.
Step 5	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode and returns to user EXEC mode.

Configuration Examples for RSVP Interface-Based Receiver Proxy

Examples Configuring RSVP Interface-Based Receiver Proxy

The four-router network in the figure below contains the configurations for the examples shown in the following sections:

Figure 102: Sample Network with an Interface-Based Receiver Proxy Configured



Configuring a Receiver Proxy on a Middle Router on Behalf of Tailend Routers

The following example configures a receiver proxy, also called a listener, on the middle router (Router 2) on behalf of the two tailend routers (Routers 3 and 4):

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface gigabitEthernet 2/0/0
Router(config-if)# ip rsvp listener outbound reply
Router(config-if)# exit
Router(config)# interface gigabitethernet 3/0/0
Router(config-if)# ip rsvp listener outbound reject
Router(config-if)# end
```

Configuring PATH Messages from a Headend Router to Tailend Routers to Test the Receiver Proxy

Note If you do not have another headend router generating RSVP PATH messages available, configure one in the network for the specific purpose of testing RSVP features such as the receiver proxy. Note that these commands are not expected (or supported) in a final deployment.

The following example configures four PATH messages from the headend router (Router 1) to the tailend routers (Routers 3 and 4):

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# ip rsvp sender-host 10.0.0.5 10.0.0.1 TCP 2 2 100 10
Router(config)# ip rsvp sender-host 10.0.0.5 10.0.0.1 UDP 1 1 100 10
Router(config)# ip rsvp sender-host 10.0.0.7 10.0.0.1 TCP 4 4 100 10
Router(config)# ip rsvp sender-host 10.0.0.7 10.0.0.1 UDP 3 3 100 10
Router(config)# end
```

Examples Verifying RSVP Interface-Based Receiver Proxy

This section contains the following verification examples:

Verifying the PATH Messages in the Database

The following example verifies that the PATH messages you configured are in the database:

```
Router# show ip rsvp sender
To              From            Pro DPort Sport Prev Hop        I/F      BPS
10.0.0.5        10.0.0.1        TCP 2     2     none            none     100K
10.0.0.5        10.0.0.1        UDP 1     1     none            none     100K
10.0.0.7        10.0.0.1        TCP 4     4     none            none     100K
10.0.0.7        10.0.0.1        UDP 3     3     none            none     100K
```

The following example verifies that a PATH message has been terminated by a receiver proxy configured to reply.

Note A receiver proxy that is configured to reject does not cause any state to be stored in the RSVP database; therefore, this **show** command does not display these PATH messages. Only one PATH message is shown.

```
Router# show ip rsvp sender detail
PATH:
  Destination 10.0.0.5, Protocol_Id 17, Don't Police , DstPort 1
  Sender address: 10.0.0.1, port: 1
  Path refreshes:
    arriving: from PHOP 10.1.2.1 on Et0/0 every 30000 msecs
  Traffic params - Rate: 100K bits/sec, Max. burst: 10K bytes
    Min Policed Unit: 0 bytes, Max Pkt Size 2147483647 bytes
  Path ID handle: 01000402.
  Incoming policy: Accepted. Policy source(s): Default
  Status: Proxy-terminated
  Output on Ethernet2/0. Policy status: NOT Forwarding. Handle: 02000401
    Policy source(s):
  Path FLR: Never repaired
```

Verifying the Running Configuration

The following example verifies the configuration for GigabitEthernet interface 2/0/0:

```
Router# show running-config interface gigbitEthernet 2/0/0
Building configuration...
Current configuration : 132 bytes
!
interface gigabitEthernet2/0/0
 ip address 172.16.0.1 255.0.0.0
 no cdp enable
 ip rsvp bandwidth 2000
 ip rsvp listener outbound reply
end
```

The following example verifies the configuration for GigabitEthernet interface 3/0/0:

```
Router# show running-config interface gigbitEthernet 3/0/0
Building configuration...
Current configuration : 133 bytes
!
interface gigabitEthernet3/0/0
 ip address 172.16.0.2 255.0.0.0
 no cdp enable
 ip rsvp bandwidth 2000
```

```
 ip rsvp listener outbound reject
end
```

Verifying the Listeners

The following example verifies the listeners (proxies) that you configured on the middle router (Router 2) on behalf of the two tailend routers (Routers 3 and 4):

```
Router# show ip rsvp listener
To                Protocol   DPort   Description                 Action    OutIf
10.0.0.0          0          0       RSVP Proxy                  reply     Et2/0
10.0.0.0          0          0       RSVP Proxy                  reject    Et3/0
```

Verifying the Reservations

The following example displays reservations established by the middle router (Router 2) on behalf of the tailend routers (Routers 3 and 4) as seen from the headend router (Router 1):

```
Router# show ip rsvp reservation
To            From          Pro DPort Sport Next Hop      I/F      Fi Serv BPS
10.0.0.7      10.0.0.1      TCP 4     4     10.0.0.2      Gi1/0    FF RATE 100K
10.0.0.7      10.0.0.1      UDP 3     3     10.0.0.2      Gi1/0    FF RATE 100K
```

The following example verifies that a reservation is locally generated (proxied). Only one reservation is shown:

```
Router# show ip rsvp reservation detail
RSVP Reservation. Destination is 10.0.0.7, Source is 10.0.0.1,
  Protocol is UDP, Destination port is 1, Source port is 1
  Next Hop: 10.2.3.3 on GigabitEthernet2/0/0
  Reservation Style is Fixed-Filter, QoS Service is Guaranteed-Rate
  Resv ID handle: 01000405.
  Created: 09:24:24 EST Fri Jun 2 2006
  Average Bitrate is 100K bits/sec, Maximum Burst is 10K bytes
  Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Status: Proxied
  Policy: Forwarding. Policy source(s): Default
```

Verifying CAC on an Outbound Interface

The following example verifies that the proxied reservation performed CAC on the local outbound interface:

```
Router# show ip rsvp installed
RSVP: GigabitEthernet2/0/0 has no installed reservations
RSVP: GigabitEthernet3/0/0
BPS    To              From            Protoc DPort  Sport
100K   10.0.0.7        10.0.0.1        UDP    1       1
```

Additional References

The following sections provide references related to the RSVP Interface-Based Receiver Proxy feature.

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
QoS configuration tasks related to RSVP	"Configuring RSVP" module
Internet draft	*RSVP Proxy Approaches* , Internet draft, October 2006 [draft-lefaucheur-tsvwg-rsvp-proxy-00.txt]

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2205	Resource ReSerVation Protocol (RSVP)

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for RSVP Interface-Based Receiver Proxy

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 80: Feature Information for RSVP Interface-Based Receiver Proxy

Feature Name	Releases	Feature Information
RSVP Interface-Based Receiver Proxy	Cisco IOS XE Release 2.6 Cisco IOS XE Release 3.8S	The RSVP Interface-Based Receiver Proxy feature lets you configure a proxy router by outbound interface instead of configuring a destination address for each flow going through the same interface. The following commands were introduced or modified: **ip rsvp bandwidth**, **ip rsvp listener outbound**, **show ip rsvp listeners**, **show ip rsvp reservation**, **show ip rsvp sender**. In Cisco IOS XE Release 3.8S, support was added for the Cisco ASR 903 Router.

Glossary

flow --A stream of data traveling between two endpoints across a network (for example, from one LAN station to another). Multiple flows can be transmitted on a single circuit.

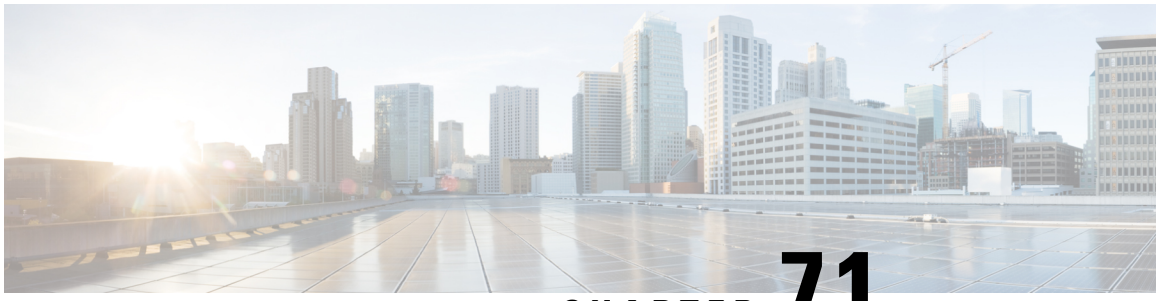
PE router --provider edge router. A router that is part of a service provider's network and is connected to a customer edge (CE) router.

proxy --A component of RSVP that manages all locally originated and terminated state.

receiver proxy --A configurable feature that allows a router to proxy RSVP RESV messages for local or remote destinations.

RSVP --Resource Reservation Protocol. A protocol for reserving network resources to provide quality of service guarantees to application flows.

set-top box--A computer that acts as a receiver and decodes the incoming signal from a satellite dish, a cable network, or a telephone line.



CHAPTER 71

RSVP Scalability Enhancements

This document describes the Cisco Resource Reservation Protocol (RSVP) scalability enhancements. It provides an overview of the feature, includes configuration tasks and examples, and lists related Cisco IOS command-line interface (CLI) commands.

Prerequisites for RSVP Scalability Enhancements

The network must support the following Cisco IOS XE features before the RSVP scalability enhancements are enabled:

- Resource Reservation Protocol (RSVP)
- Class-based weighted fair queueing (CBWFQ)

Restrictions for RSVP Scalability Enhancements

- Sources should not send marked packets without an installed reservation.
- Sources should not send marked packets that exceed the reserved bandwidth.
- Sources should not send marked packets to a destination other than the reserved path.

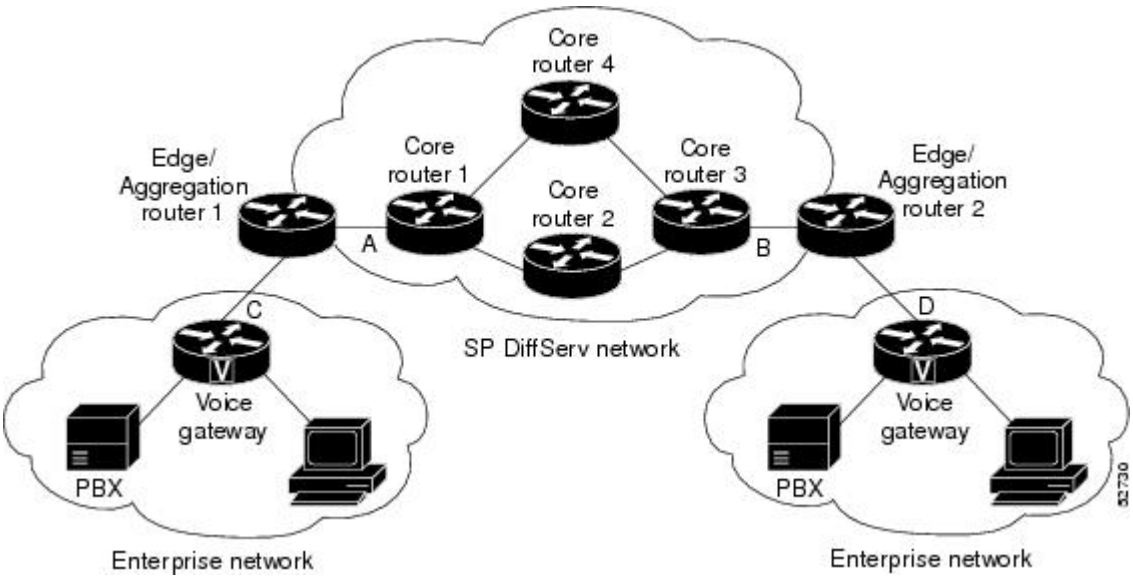
Information About RSVP Scalability Enhancements

RSVP typically performs admission control, classification, policing, and scheduling of data packets on a per-flow basis and keeps a database of information for each flow. RSVP scalability enhancements let you select a resource provider (formerly called a quality of service (QoS) provider) and disable data packet classification so that RSVP performs admission control only. This facilitates integration with service provider (differentiated services (DiffServ)) networks and enables scalability across enterprise networks.

CBWFQ provides the classification, policing, and scheduling functions. CBWFQ puts packets into classes based on the differentiated services code point (DSCP) value in the packet's Internet Protocol (IP) header, thereby eliminating the need for per-flow state and per-flow processing.

The figure below shows two enterprise networks interconnected through a service provider (SP) network. The SP network has an IP backbone configured as a DiffServ network. Each enterprise network has a voice gateway connected to an SP edge/aggregation router via a wide area network (WAN) link. The enterprise networks are connected to a private branch exchange (PBX).

Figure 103: RSVP/DiffServ Integration Topology



The voice gateways are running classic RSVP, which means RSVP is keeping a state per flow and also classifying, marking, and scheduling packets on a per flow basis. The edge/aggregation routers are running classic RSVP on the interfaces (labeled C and D) connected to the voice gateways and running RSVP for admission control only on the interfaces connected to core routers 1 and 3. The core routers in the DiffServ network are not running RSVP, but are forwarding the RSVP messages to the next hop. The core routers inside the DiffServ network implement a specific per hop behavior (PHB) for a collection of flows that have the same DSCP value.

The voice gateways identify voice data packets and set the appropriate DSCP in their IP headers such that the packets are classified into the priority class in the edge/aggregation routers and in core routers 1, 2, 3 or 1, 4, 3.

The interfaces or the edge/aggregation routers (labeled A and B) connected to core routers 1 and 3 are running RSVP, but are doing admission control only per flow against the RSVP bandwidth pool configured on the

DiffServ interfaces of the edge/aggregation routers. CBWFQ is performing the classification, policing, and scheduling functions.

Benefits of RSVP Scalability Enhancements

Enhanced Scalability

RSVP scalability enhancements handle similar flows on a per-class basis instead of a per-flow basis. Since fewer resources are required to maintain per-class QoS guarantees, the RSVP scability enhancements provide faster processing results, thereby enhancing scalability.

Improved Router Performance

RSVP scalability enhancements improve router performance by reducing the cost for data-packet classification and scheduling, which decrease CPU resource consumption. The saved resources can then be used for other network management functions.

How to Configure RSVP Scalability Enhancements

Configuring the Resource Provider

Note The resource provider was formerly called the QoS provider.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / subslot / port*
4. **ip rsvp bandwidth** [*interface-kbps* [*single-flow-kbps*[**bc1** *kbps* | **sub-pool** *kbps*]*]*| **percent** *percent-bandwidth* [*single-flow-kbps*]]
5. **ip rsvp resource-provider none**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**interface** *type slot* / *subslot* / *port* **Example:** `Router(config)# interface gigabitEthernet 0/0/0`	Configures the interface type and enters interface configuration mode.
Step 4	**ip rsvp bandwidth** [*interface-kbps* [*single-flow-kbps*[**bc1** *kbps* | **sub-pool** *kbps*]]| **percent** *percent-bandwidth* [*single-flow-kbps*]] **Example:** `Router(config-if)# ip rsvp bandwidth 7500 7500`	Enables RSVP on an interface. • The optional *interface-kbps* and *single-flow-kbps* arguments specify the amount of bandwidth that can be allocated by RSVP flows or to a single flow, respectively. Range is from 1 to 10000000. • The optional **sub-pool**and *kbps*keyword and argument specify subpool traffic and the amount of bandwidth that can be allocated by RSVP flows. Range is from 1 to 10000000. **Note** Repeat this command for each interface on which you want to enable RSVP. **Note** The bandwidth that you configure on the interface must match the bandwidth that you configure for the CBWFQ priority queue.
Step 5	**ip rsvp resource-provider none** **Example:** `Router(config-if)# ip rsvp resource-provider none`	Sets the resource provider to none. **Note** Setting the resource provider to none instructs RSVP to not associate any resources, such as WFQ queues or bandwidth, with a reservation.
Step 6	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Disabling Data Packet Classification

Perform the following task to disable data packet classification. Disabling data packet classification instructs RSVP not to process every packet, but to perform admission control only.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot* / *subslot* / *port*
4. **ip rsvp data-packet classification none**

5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / subslot / port* **Example:** `Router(config)# interface gigabitEthernet0/0/0`	Configures the interface type and enters interface configuration mode.
Step 4	**ip rsvp data-packet classification none** **Example:** `Router(config-if)# ip rsvp data-packet classification none`	Disables data packet classification.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Configuring Class Maps and Policy Maps

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** *class-map-name*
4. **exit**
5. **policy-map** *policy-map-name*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Router> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** *class-map-name* **Example:** `Router(config)# class-map class1`	Specifies the name of the class for which you want to create or modify class-map match criteria and enters the class map configuration mode.
Step 4	**exit** **Example:** `Router(config-cmap)# exit`	Returns to the global configuration mode.
Step 5	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy1`	Specifies the name of the policy map to be created, added to, or modified before you can configure policies for classes whose match criteria are defined in a class map.
Step 6	**end** **Example:** `Router(config-control-policymap)# end`	(Optional) Returns to privileged EXEC mode.

Attaching a Policy Map to an Interface

Perform the following task to attach a policy map to an interface. If at the time you configure the RSVP scalability enhancements, there are existing reservations that use classic RSVP, no additional marking, classification, or scheduling is provided for these flows. You can also delete these reservations after you configure the RSVP scalability enhancements.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / subslot / port*
4. **service-policy** {**input** | **output**} *policy-map-name*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / subslot / port* **Example:** `Router(config)# interface gigabitEthernet 0/0/0`	Configures the interface type and enters interface configuration mode.
Step 4	**service-policy** {**input** | **output**} *policy-map-name* **Example:** `Router(config-if)# service-policy input policy1`	Attaches a single policy map to one or more interfaces to specify the service policy for those interfaces.
Step 5	**end** **Example:** `Router(config-if)# end`	(Optional) Returns to privileged EXEC mode.

Verifying RSVP Scalability Enhancements Configuration

SUMMARY STEPS

1. Enter the **show ip rsvp interface detail**command to display information about interfaces, subinterfaces, resource providers, and data packet classification. The output in the following example shows that the ATM interface 6/0 has resource provider none configured and that data packet classification is turned off:
2. Enter the **show ip rsvp installed detail**command to display information about interfaces, subinterfaces, their admitted reservations, bandwidth, resource providers, and data packet classification.
3. Wait for a while, then enter the **show ip rsvp installed detail**command again. In the following output, notice there is no increment in the number of packets classified:

DETAILED STEPS

Step 1 Enter the **show ip rsvp interface detail**command to display information about interfaces, subinterfaces, resource providers, and data packet classification. The output in the following example shows that the ATM interface 6/0 has resource provider none configured and that data packet classification is turned off:

Example:

```
Router# show ip rsvp interface detail
 AT6/0:
   Bandwidth:
     Curr allocated: 190K bits/sec
     Max. allowed (total): 112320K bits/sec
     Max. allowed (per flow): 112320K bits/sec
   Neighbors:
     Using IP encap: 1.  Using UDP encaps: 0
   DSCP value used in Path/Resv msgs: 0x30
   RSVP Data Packet Classification is OFF
   RSVP resource provider is: none
```

Note The last two lines in the preceding output verify that the RSVP scalability enhancements (disabled data packet classification and resource provider none) are present.

Step 2 Enter the **show ip rsvp installed detail**command to display information about interfaces, subinterfaces, their admitted reservations, bandwidth, resource providers, and data packet classification.

Example:

```
Router# show ip rsvp installed detail
RSVP: GigabitEthernet0/0/0 has no installed reservations
RSVP: ATM6/0 has the following installed reservations
RSVP Reservation. Destination is 10.20.20.212, Source is 10.10.10.211,
  Protocol is UDP, Destination port is 14, Source port is 14
  Reserved bandwidth: 50K bits/sec, Maximum burst: 1K bytes, Peak rate: 50K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 0 packets (0 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 54 seconds
  Long-term average bitrate (bits/sec): 0M reserved, 0M best-effort
RSVP Reservation. Destination is 10.20.20.212, Source is 10.10.10.211,
  Protocol is UDP, Destination port is 10, Source port is 10
  Reserved bandwidth: 20K bits/sec, Maximum burst: 1K bytes, Peak rate: 20K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 0 packets (0 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 80 seconds
  Long-term average bitrate (bits/sec): 0M reserved, 0M best-effort
```

Step 3 Wait for a while, then enter the **show ip rsvp installed detail**command again. In the following output, notice there is no increment in the number of packets classified:

Example:

```
Router# show ip rsvp installed detail
RSVP: Ethernet3/3 has no installed reservations
RSVP: ATM6/0 has the following installed reservations
RSVP Reservation. Destination is 10.20.20.212, Source is 10.10.10.211,
  Protocol is UDP, Destination port is 14, Source port is 14
  Reserved bandwidth: 50K bits/sec, Maximum burst: 1K bytes, Peak rate: 50K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 0 packets (0 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 60 seconds
```

```
  Long-term average bitrate (bits/sec): 0 reserved, OM best-effort
RSVP Reservation. Destination is 10.20.20.212, Source is 10.10.10.211,
  Protocol is UDP, Destination port is 10, Source port is 10
  Reserved bandwidth: 20K bits/sec, Maximum burst: 1K bytes, Peak rate: 20K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 0 packets (0 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 86 seconds
  Long-term average bitrate (bits/sec): OM reserved, 0M best-effort
```

Monitoring and Maintaining RSVP Scalability Enhancements

To monitor and maintain RSVP scalability enhancements, use the following commands in EXEC mode. The following commands can be entered in any order.

Command	Purpose
`Router# show ip rsvp installed`	Displays information about interfaces and their admitted reservations.
`Router# show ip rsvp installed detail`	Displays additional information about interfaces and their admitted reservations.
`Router# show ip rsvp interface`	Displays RSVP-related interface information.
`Router# show ip rsvp interface detail`	Displays additional RSVP-related interface information.
`Router# show queueing [custom | fair | priority | random-detect [interface serial-number]]`	Displays all or selected configured queueing strategies and available bandwidth for RSVP reservations.

Configuration Examples for RSVP Scalability Enhancements

Examples Configuring the Resource Provider as None with Data Classification Turned Off

Following is output from the **showiprsvpinterfacedetail** command before a resource provider is configured as none and data-packet classification is turned off:

```
Router# show ip rsvp interface detail
 AT6/0:
   Bandwidth:
     Curr allocated: 190K bits/sec
     Max. allowed (total): 112320K bits/sec
```

```
    Max. allowed (per flow): 112320K bits/sec
  Neighbors:
    Using IP encap: 1.  Using UDP encaps: 0
  DSCP value used in Path/Resv msgs: 0x30
```

Following is the output from the **showqueueing**command before a resource provider is configured as none and data packet classification is turned off:

```
Router# show queueing int atm6/0
  Interface ATM6/0 VC 200/100
  Queueing strategy: weighted fair
  Output queue: 63/512/64/3950945 (size/max total/threshold/drops)
     Conversations  2/5/64 (active/max active/max total)
     Reserved Conversations 0/0 (allocated/max allocated)
     Available Bandwidth 450 kilobits/sec
```

Note New reservations do not reduce the available bandwidth (450 kilobits/sec shown above). Instead RSVP performs admission control only using the bandwidth limit configured in the **iprsvpbandwidth** command. The bandwidth configured in this command should match the bandwidth configured in the CBWFQ class that you set up to handle the reserved traffic.

The following example shows how to configure resource provider as none:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface atm6/0
Router(config-if)# ip rsvp resource-provider none

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

The following example shows how to turn off the data packet classification:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)# interface atm6/0
Router(config-if)# ip rsvp data-packet classification none

Router(config-if)# end
```

Following is the output from the **showiprsvpinterfacedetail** command after resource provider has been configured as none and data packet classification has been turned off:

```
Router# show ip rsvp interface detail
 AT6/0:
   Bandwidth:
     Curr allocated: 190K bits/sec
     Max. allowed (total): 112320K bits/sec
     Max. allowed (per flow): 112320K bits/sec
   Neighbors:
     Using IP encap: 1.  Using UDP encaps: 0
   DSCP value used in Path/Resv msgs: 0x30
   RSVP Data Packet Classification is OFF
   RSVP resource provider is: none
```

The following output from the **showiprsvpinstalleddetail** command verifies that resource provider none is configured and data packet classification is turned off:

```
Router# show ip rsvp installed detail
RSVP: ATM6/0 has the following installed reservations
RSVP Reservation. Destination is 10.20.20.212, Source is 10.10.10.211,
  Protocol is UDP, Destination port is 14, Source port is 14
  Reserved bandwidth: 50K bits/sec, Maximum burst: 1K bytes, Peak rate: 50K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 3192 packets (1557696 bytes)
  Data given best-effort service: 42 packets (20496 bytes)
  Reserved traffic classified for 271 seconds
  Long-term average bitrate (bits/sec): 45880 reserved, 603 best-effort
RSVP Reservation. Destination is 10.20.20.212, Source is 10.10.10.211,
  Protocol is UDP, Destination port is 10, Source port is 10
  Reserved bandwidth: 20K bits/sec, Maximum burst: 1K bytes, Peak rate: 20K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 1348 packets (657824 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 296 seconds
  Long-term average bitrate (bits/sec): 17755 reserved, 0M best-effort
```

The following output shows no increments in packet counts after the source sends data packets that match the reservation:

```
Router# show ip rsvp installed detail
RSVP: GigabitEthernet3/3 has no installed reservations
RSVP: ATM6/0 has the following installed reservations
RSVP Reservation. Destination is 10.20.20.212, Source is 10.10.10.211,
  Protocol is UDP, Destination port is 14, Source port is 14
  Reserved bandwidth: 50K bits/sec, Maximum burst: 1K bytes, Peak rate: 50K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 3192 packets (1557696 bytes)
  Data given best-effort service: 42 packets (20496 bytes)
  Reserved traffic classified for 282 seconds
  Long-term average bitrate (bits/sec): 44051 reserved, 579 best-effort
RSVP Reservation. Destination is 145.20.20.212, Source is 145.10.10.211,
  Protocol is UDP, Destination port is 10, Source port is 10
  Reserved bandwidth: 20K bits/sec, Maximum burst: 1K bytes, Peak rate: 20K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 1348 packets (657824 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 307 seconds
  Long-term average bitrate (bits/sec): 17121 reserved, 0M best-effort
```

The following output verifies that data packet classification is occurring:

```
Router# show ip rsvp installed detail
Enter configuration commands, one per line.  End with CNTL/Z.
RSVP: ATM6/0 has the following installed reservations
RSVP Reservation. Destination is 10.20.20.212, Source is 10.10.10.211,
  Protocol is UDP, Destination port is 14, Source port is 14
  Reserved bandwidth: 50K bits/sec, Maximum burst: 1K bytes, Peak rate: 50K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 3683 packets (1797304 bytes)
```

```
  Data given best-effort service: 47 packets (22936 bytes)
  Reserved traffic classified for 340 seconds
  Long-term average bitrate (bits/sec): 42201 reserved, 538 best-effort
RSVP Reservation. Destination is 10.20.20.212, Source is 10.10.10.211,
  Protocol is UDP, Destination port is 10, Source port is 10
  Reserved bandwidth: 20K bits/sec, Maximum burst: 1K bytes, Peak rate: 20K bits/sec
  Min Policed Unit: 0 bytes, Max Pkt Size: 1514 bytes
  Resource provider for this flow: None
  Conversation supports 1 reservations
  Data given reserved service: 1556 packets (759328 bytes)
  Data given best-effort service: 0 packets (0 bytes)
  Reserved traffic classified for 364 seconds
  Long-term average bitrate (bits/sec): 16643 reserved, 0M best-effort
```

Note You can use **debugiprsvptraffic-control** and **debugiprsvpwfq** simultaneously. Use the**showdebug** command to see which debugging commands are enabled.

Additional References

The following sections provide references related to the RSVP Scalability Enhancements feature.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
QoS configuration tasks related to RSVP	"Configuring RSVP" module

Standards

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

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2205	Resource Reservation Protocol
RFC 2206	RSVP Management Information Base using SMIv2

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for RSVP Scalability Enhancements

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 81: Feature Information for RSVP Scalability Enhancements

Feature Name	Releases	Feature Information
RSVP Scalability Enhancements	Cisco IOS XE Release 2.6 Cisco IOS XE Release 3.8S	RSVP scalability enhancements let you select a resource provider (formerly called a QoS provider) and disable data packet classification so that RSVP performs admission control only. This facilitates integration with service provider (DiffServ) networks and enables scalability across enterprise networks. The following commands were introduced or modified: **debug ip rsvp traffic-control**, **debug ip rsvp wfq**, **ip rsvp data-packet classification none**, **ip rsvp resource-provider**, **show ip rsvp installed**, **show ip rsvp interface**, **show queueing**. In Cisco IOS XE Release 3.8S, support was added for the Cisco ASR 903 Router.

Glossary

admission control --The process by which an RSVP reservation is accepted or rejected based on end-to-end available network resources.

aggregate --A collection of packets with the same DSCP.

bandwidth --The difference between the highest and lowest frequencies available for network signals. This term also describes the rated throughput capacity of a given network medium or protocol.

CBWFQ -- class-based weighted fair queueing. A queueing mechanism that extends the standard WFQ functionality to provide support for user-defined traffic classes.

DiffServ --differentiated services. An architecture based on a simple model where traffic entering a network is classified and possibly conditioned at the boundaries of the network. The class of traffic is then identified with a DS code point or bit marking in the IP header. Within the core of the network, packets are forwarded according to the per-hop behavior associated with the DS code point.

DSCP --differentiated services code point. The six most significant bits of the 1-byte IP type of service (ToS) field. The per-hop behavior represented by a particular DSCP value is configurable. DSCP values range between 0 and 63.

enterprise network --A large and diverse network connecting most major points in a company or other organization.

flow --A stream of data traveling between two endpoints across a network (for example, from one LAN station to another). Multiple flows can be transmitted on a single circuit.

packet --A logical grouping of information that includes a header containing control information and (usually) user data. Packets most often refer to network-layer units of data.

PBX --private branch exchange. A digital or analog telephone switchboard located on the subscriber premises and used to connect private and public telephone networks.

PHB --per-hop behavior. A DiffServ concept that specifies how specifically marked packets are to be treated by each DiffServ router.

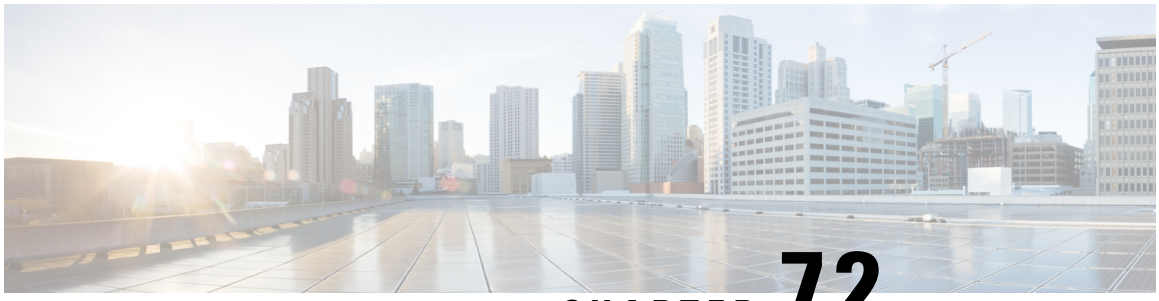
QoS --quality of service. A measure of performance for a transmission system that reflects its transmission quality and service availability.

RSVP --Resource Reservation Protocol. A protocol for reserving network resources to provide quality of service guarantees to application flows.

Voice over IP --See VoIP.

VoIP --Voice over IP. The ability to carry normal telephony-style voice over an IP-based internet maintaining telephone-like functionality, reliability, and voice quality.

WFQ --weighted fair queueing. A queue management algorithm that provides a certain fraction of link bandwidth to each of several queues, based on the relative bandwidth applied to each of the queues.



CHAPTER 72

Control Plane DSCP Support for RSVP

This document describes the Cisco Control Plane DSCP Support for RSVP feature.

Prerequisites for Control Plane DSCP Support for RSVP

The network must support Resource Reservation Protocol (RSVP) before the Control Plane DSCP Support for RSVP feature is enabled.

Restrictions for Control Plane DSCP Support for RSVP

Control plane DSCP support for RSVP can be configured on interfaces and subinterfaces only. It affects all RSVP messages that are sent out on the interface or that are present on any logical circuit of the interface, including subinterfaces, permanent virtual circuits (PVCs), and switched virtual circuits (SVCs).

Information About Control Plane DSCP Support for RSVP

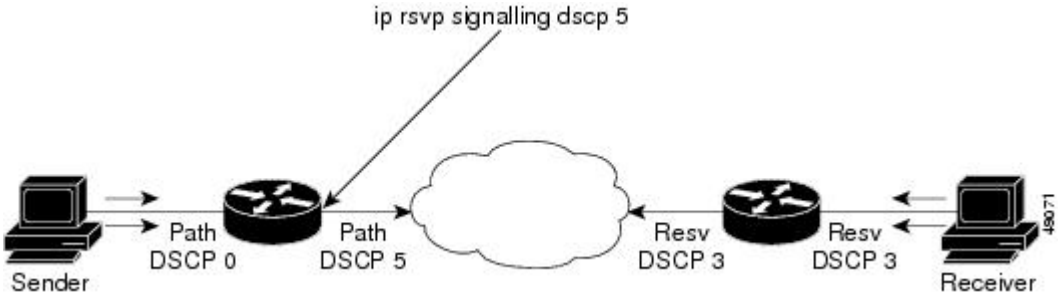
Typically, networks operate on a best-effort delivery basis, which means that all traffic has equal priority and an equal chance of being delivered in a timely manner. When congestion occurs, all traffic has an equal chance of being dropped.

Before traffic can be handled according to its unique requirements, it must be identified or labeled. There are numerous classification techniques for doing this. These include Layer 3 schemes such as IP precedence or differentiated services code point (DSCP), Layer 2 schemes such as 802.1P, and implicit characteristics of the data itself, such as the traffic type using the Real-Time Transport Protocol (RTP) and a defined port range.

The Control Plane DSCP Support for RSVP feature allows you to set the priority value in the type of service (ToS) byte or differentiated services (DiffServ) field in the IP header for RSVP messages. The IP header functions with resource providers such as weighted fair queueing (WFQ), so that voice frames have priority over data fragments and data frames. When packets arrive in a router's output queue, the voice packets are placed ahead of the data frames.

The figure below shows a path message originating from a sender with a DSCP value of 0 (the default), which is changed ito 5 to give the message a higher priority, and it shows a reservation (resv) message originating from a receiver with a DSCP of 3.

Figure 104: Control Plane DSCP Support for RSVP



Raising the DSCP value reduces the possibility of packets being dropped, thereby improving call setup time in VoIP environments.

Benefits of Control Plane DSCP Support for RSVP

Faster Call Setup Time

The Control Plane DSCP Support for RSVP feature allows you to set the priority for RSVP messages. In a DiffServ QoS environment, higher-priority packets get serviced before lower-priority packets, thereby improving the call setup time for RSVP sessions.

Improved Message Delivery

During periods of congestion, routers drop lower-priority traffic before they drop higher-priority traffic. Since RSVP messages can now be marked with higher priority, the likelihood of these messages being dropped is significantly reduced.

Faster Recovery After Failure Conditions

When heavy congestion occurs, many packets are dropped. Network resources attempt to retransmit almost instantaneously, resulting in further congestion. This leads to a considerable reduction in throughput.

Previously, RSVP messages were marked best effort and subject to being dropped by congestion avoidance mechanisms such as weighted random early detection (WRED). However, with the Control Plane DSCP Support for RSVP feature, RSVP messages are likely to be dropped later, if at all, thereby providing faster recovery of RSVP reservations.

How to Configure Control Plane DSCP Support for RSVP

Enabling RSVP on an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / subslot / port*
4. **ip rsvp bandwidth** [*interface-kbps*] [*single-flow-kbps*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / subslot / port* **Example:** `Router(config)# interface gigbitEthernet 0/0/0`	Enters interface configuration mode for a specific interface.
Step 4	**ip rsvp bandwidth** [*interface-kbps*] [*single-flow-kbps*] **Example:** `Router(config-if)# ip rsvp bandwidth 23 43`	Enables RSVP on an interface.

Specifying the DSCP

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / subslot / port*
4. **ip rsvp signalling dscp** *value*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / subslot / port* **Example:** `Router(config)# interface gigbitEthernet 0/1/0`	Enters interface configuration mode for a specific interface.
Step 4	**ip rsvp signalling dscp** *value* **Example:** `Router(config-if)# ip rsvp signalling dscp 10`	Specifies the DSCP to be used on all RSVP messages that are transmitted on an interface.

Verifying Control Plane DSCP Support for RSVP Configuration

SUMMARY STEPS

1. Enter the **show running-config** command to verify the configuration.
2. Enter the **show ip rsvp interface detail**command to display RSVP-related interface information. The following is sample output from the **show ip rsvp interface detail**command. Interfaces that are not configured for DSCP do not show the DSCP value, which is 0 by default.

DETAILED STEPS

Step 1 Enter the **show running-config** command to verify the configuration.

Step 2 Enter the **show ip rsvp interface detail**command to display RSVP-related interface information. The following is sample output from the **show ip rsvp interface detail**command. Interfaces that are not configured for DSCP do not show the DSCP value, which is 0 by default.

Example:

```
Router# show
 ip rsvp interface detail
Gi0/0/0:
   RSVP: Disabled
   Interface State: N/A
   Bandwidth:
     Curr allocated: 0 bits/sec
     Max. allowed (total): 0 bits/sec
```

```
  Max. allowed (per flow): 0 bits/sec
  Max. allowed for LSP tunnels using sub-pools (pool 1): 0 bits/sec
  Set aside by policy (total): 0 bits/sec
Traffic Control:
  RSVP Data Packet Classification is ON
Signalling:
  DSCP value used in RSVP msgs: 0x17
  Number of refresh intervals to enforce blockade state: 4
Authentication: disabled
  Key chain:   <none>
  Type:        md5
  Window size: 1
  Challenge:   disabled
FRR Extension:
  Backup Path: Not Configured
BFD Extension:
  State: Disabled
  Interval: Not Configured
RSVP Hello Extension:
  State: Disabled
RFC 3175 Aggregation: Disabled
  Role: exterior.
```

Configuration Examples for Control Plane DSCP Support for RSVP

The following example shows how to enable RSVP on an interface, specify the DSCP, and verify the control plane DSCP support for RSVP.

```
Router> enable
Router# config terminal
Router(config)# interface gigabitethernet 3/1/0
Router(config-if)# ip rsvp bandwidth 7500 7500
Router(config-if)# ip rsvp signalling dscp 48
Router(config-if)# end
```

The following example shows how to display the RSVP-related information.

```
Router# show running-config interface gigabitEthernet 0/0/0
interface gigabitEthernet 0/0/0
ip address 10.10.10.1 255.255.255.0
fair-queue 64 256 235
ip rsvp signalling dscp 48
ip rsvp bandwidth 7500 7500
```

Additional References

The following sections provide references related to the Control Plane DSCP Support for RSVP feature.

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
RSVP Commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Quality of service overview	"Quality of Service Overview" module

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
None	To locate and download MIBs for selected platforms, software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2205	*Resource Reservation Protocol*
RFC 2206	*RSVP Management Information Base using SMIv2*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Control Plane DSCP Support for RSVP

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 82: Feature Information for Control Plane DSCP Support for RSVP

Feature Name	Releases	Feature Information
Control Plane DSCP Support for RSVP	Cisco IOS XE Release 2.6	The Control Plane DSCP Support for RSVP feature allows you to set the priority value in ToS byte or DiffServ field in the IP header for RSVP messages. The following commands were introduced or modified: **ip rsvp signalling dscp**, **show ip rsvp interface**.

Glossary

CBWFQ -- class-based weighted fair queueing. A queueing mechanism that extends the standard WFQ functionality to provide support for user-defined traffic classes.

DiffServ --differentiated services. An architecture based on a simple model where traffic that is entering a network is classified and possibly conditioned at the boundaries of the network. The class of traffic is then identified with a DS code point or bit marking in the IP header. Within the core of the network, packets are forwarded according to the per-hop behavior associated with the DS code point.

DSCP --differentiated services code point. The six most significant bits of the 1-byte IP type of service (ToS) field. The per-hop behavior represented by a particular DSCP value is configurable. DSCP values range between 0 and 63.

IP precedence --The three most significant bits of the 1-byte type of service (ToS) field. IP precedence values range between 0 for low priority and 7 for high priority.

latency --The delay between the time when a device receives a packet and the time when the packet is forwarded out the destination port.

marking --The process of setting a Layer 3 DSCP value in a packet.

QoS --quality of service. A measure of performance for a transmission system that reflects its transmission quality and service availability.

RSVP --Resource Reservation Protocol. A protocol for reserving network resources to provide quality of service guarantees to application flows.

ToS --type of service. An 8-bit value in the IP header field.

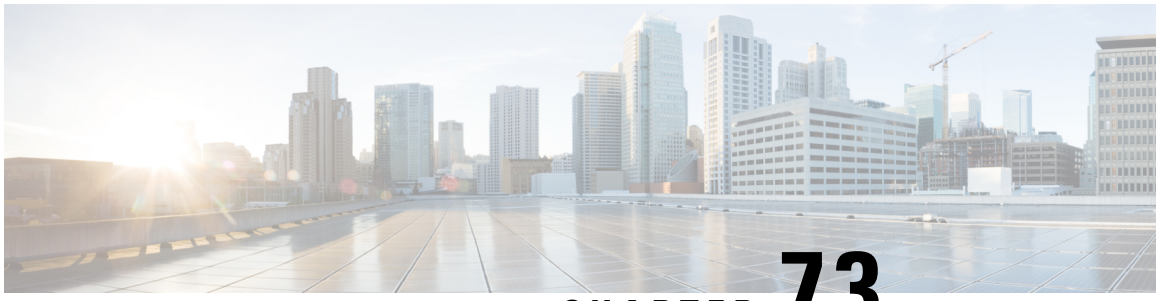
type of service --See ToS.

Voice over IP --See VoIP.

VoIP --Voice over IP. The ability to carry normal telephony-style voice over an IP-based internet while maintaining telephone-like functionality, reliability, and voice quality.

WFQ --weighted fair queueing. A queue management algorithm that provides a certain fraction of link bandwidth to each of several queues, based on relative bandwidth applied to each of the queues.

WRED --weighted random early detection. A congestion avoidance mechanism that slows traffic by randomly dropping packets when there is congestion.



CHAPTER 73

MPLS TE - Tunnel-Based Admission Control

The MPLS TE--Tunnel-Based Admission Control (TBAC) feature enables classic Resource Reservation Protocol (RSVP) unicast reservations that are traveling across a Multiprotocol Label Switching traffic engineering (MPLS TE) core to be aggregated over an MPLS TE tunnel.

Prerequisites for MPLS TE - Tunnel-Based Admission Control

- You must configure an MPLS TE tunnel in the network.
- You must configure RSVP on one or more interfaces on at least two neighboring routers that share a link within the network.

Restrictions for MPLS TE - Tunnel-Based Admission Control

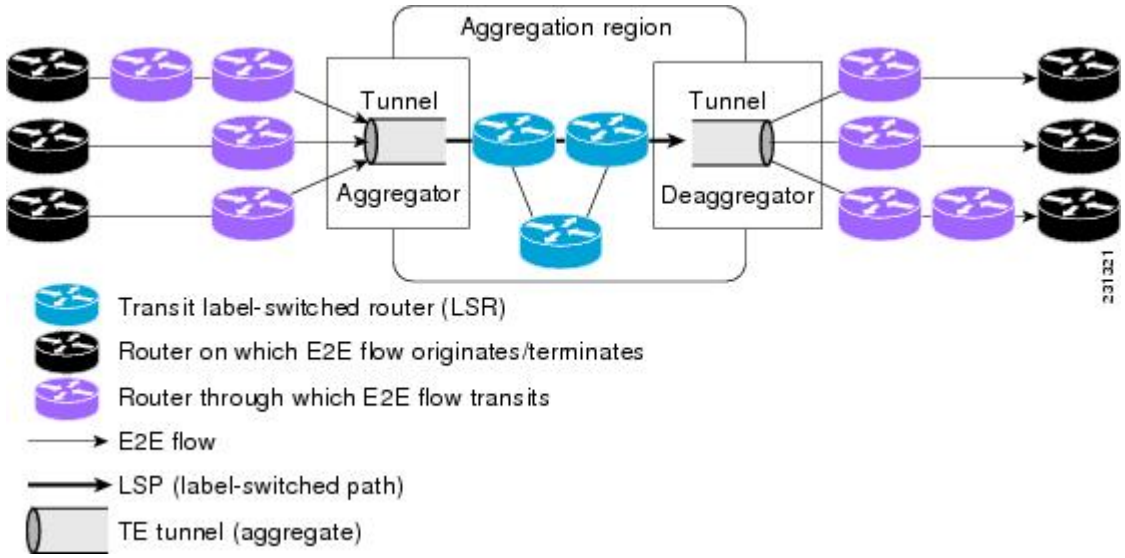
- Only IPv4 unicast RSVP flows are supported.
- Primary, one-hop tunnels are not supported. The TE tunnel cannot be a member of a class-based tunnel selection (CBTS) bundle.
- Multitopology Routing (MTR) is not supported.
- Only preestablished aggregates are supported. They can be configured statically or dynamically using command-line interface (CLI) commands.

Information About MPLS TE - Tunnel-Based Admission Control

Feature Overview of MPLS TE - Tunnel-Based Admission Control

TBAC aggregates reservations from multiple, classic RSVP sessions over different forms of tunneling technologies that include MPLS TE tunnels, which act as aggregate reservations in the core. The figure below gives an overview of TBAC.

Figure 105: TBAC Overview



The figure below shows three RSVP end-to-end (E2E) flows that originate at routers on the far left, and terminate on routers at the far right. These flows are classic RSVP unicast flows, meaning that RSVP is maintaining a state for each flow. There is nothing special about these flows, except that along their path, these flows encounter an MPLS-TE core, where there is a desire to avoid creating a per-flow RSVP state.

When the E2E flows reach the edge of the MPLS-TE core, they are aggregated onto a TE tunnel. This means that when transiting through the MPLS-TE core, their state is represented by a single state; the TE tunnel is within the aggregation region, and their packets are forwarded (label-switched) by the TE tunnel. For example, if 100 E2E flows traverse the same aggregator and deaggregator, rather than creating 100 RSVP states (PATH and RESV messages) within the aggregation region, a single RSVP-TE state is created, that of the aggregate, which is the TE tunnel, to allocate and maintain the resources used by the 100 E2E flows. In particular, the bandwidth consumed by E2E flows within the core is allocated and maintained in aggregate by the TE tunnel. The bandwidth of each E2E flow is normally admitted into the TE tunnel at the headend, just as any E2E flow's bandwidth is admitted onto an outbound link in the absence of aggregation.

Benefits of MPLS TE - Tunnel-Based Admission Control

To understand the benefits of TBAC, you should be familiar with how Call Admission Control (CAC) works for RSVP and Quality of Service (QoS).

Cost Effective

Real-time traffic is very sensitive to loss and delay. CAC avoids QoS degradation for real-time traffic because CAC ensures that the accepted load always matches the current network capacity. As a result, you do not have to overprovision the network to compensate for absolute worst peak traffic or for reduced capacity in case of failure.

Improved Accuracy

CAC uses RSVP signaling, which follows the same path as the real-time flow, and routers make a CAC decision at every hop. This ensures that the CAC decision is very accurate and dynamically adjusts to the current conditions such as a reroute or an additional link. Also, RSVP provides an explicit CAC response (admitted or rejected) to the application, so that the application can react appropriately and fast; for example, sending a busy signal for a voice call, rerouting the voice call on an alternate VoIP route, or displaying a message for video on demand.

RSVP and MPLS TE Combined

TBAC allows you to combine the benefits of RSVP with those of MPLS TE. Specifically, you can use MPLS TE inside the network to ensure that the transported traffic can take advantage of Fast Reroute protection (50-millisecond restoration), Constraint Based Routing (CBR), and aggregate bandwidth reservation.

Seamless Deployment

TBAC allows you to deploy IPv4 RSVP without any impact on the MPLS part of the network because IPv4 RSVP is effectively tunneled inside MPLS TE tunnels that operate unchanged as per regular RSVP TE. No upgrade or additional protocol is needed in the MPLS core.

Enhanced Scaling Capability

TBAC aggregates multiple IPv4 RSVP reservations ingressing from the same MPLS TE headend router into a single MPLS TE tunnel and egressing from the same MPLS TE tailend router.

How to Configure MPLS TE - Tunnel-Based Admission Control

Enabling RSVP QoS

Perform this task to enable RSVP QoS globally on a device.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp qos**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp qos** **Example:** `Device(config)# ip rsvp qos`	Enables RSVP QoS globally on a device.
Step 4	**end** **Example:** `Device(config)# end`	(Optional) Returns to privileged EXEC mode.

Enabling MPLS TE

Perform this task to enable MPLS TE. This task enables MPLS TE globally on a router that is running RSVP QoS.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls traffic-eng tunnels**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**mpls traffic-eng tunnels** **Example:** `Router(config)# mpls traffic-eng tunnels`	Enables MPLS TE globally on a router.
Step 4	**end** **Example:** `Router(config)# end`	(Optional) Returns to privileged EXEC mode.

Configuring an MPLS TE Tunnel Interface

Before you begin

You must configure an MPLS-TE tunnel in your network before you can proceed. For detailed information, see the "MPLS Traffic Engineering (TE)--Automatic Bandwidth Adjustment for TE Tunnels" module.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *number*
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *number* **Example:** `Router(config)# interface tunnel 1`	Specifies a tunnel interface and enters interface configuration mode.
Step 4	**end** **Example:**	(Optional) Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Router(config-if)# end`	

Configuring RSVP Bandwidth on an MPLS TE Tunnel Interface

Perform this task to configure RSVP bandwidth on the MPLS TE tunnel interface that you are using for the aggregation.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface tunnel** *number*
4. **ip rsvp bandwidth** [*interface-kbps*] [*single-flow-kbps*]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface tunnel** *number* **Example:** `Router(config)# interface tunnel 1`	Specifies a tunnel interface and enters interface configuration mode.
Step 4	**ip rsvp bandwidth** [*interface-kbps*] [*single-flow-kbps*] **Example:** `Router(config-if)# ip rsvp bandwidth 7500`	Enables RSVP bandwidth on an interface. • The optional *interface-kbps* and *single-flow-kbps* arguments specify the amount of bandwidth that can be allocated by RSVP flows or to a single flow, respectively. Values are from 1 to 10000000. **Note** You must enter a value for the *interface-kbps*argument on a tunnel interface.
Step 5	**end** **Example:**	(Optional) Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Router(config-if)# end`	

Verifying the TBAC Configuration

Note You can use the following **show** commands in user EXEC or privileged EXEC mode, in any order.

SUMMARY STEPS

1. **enable**
2. **show ip rsvp**
3. **show ip rsvp reservation** [**detail**] [**filter** [**destination** {*ip-address* | *hostname*}] [**dst-port** *port-number*] [**source** {*ip-address* | *hostname*}] [**src-port** *port-number*]]
4. **show ip rsvp sender** [**detail**] [**filter** [**destination** *ip-address* | *hostname*] [**dst-port** *port-number*] [**source** *ip-address* | *hostname*] [**src-port** *port-number*]]
5. **show mpls traffic-eng link-management bandwidth-allocation** [**summary**] [*interface-type interface-number*]
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	(Optional) Enables privileged EXEC mode. • Enter your password if prompted. **Note** Omit this step if you are using the **show** commands in user EXEC mode.
Step 2	**show ip rsvp** **Example:** `Router# show ip rsvp`	Displays specific information for RSVP categories.
Step 3	**show ip rsvp reservation** [**detail**] [**filter** [**destination** {*ip-address* | *hostname*}] [**dst-port** *port-number*] [**source** {*ip-address* | *hostname*}] [**src-port** *port-number*]] **Example:** `Router# show ip rsvp reservation detail`	Displays RSVP-related receiver information currently in the database.
Step 4	**show ip rsvp sender** [**detail**] [**filter** [**destination** *ip-address* | *hostname*] [**dst-port** *port-number*] [**source** *ip-address* | *hostname*] [**src-port** *port-number*]]	Displays RSVP PATH-related sender information currently in the database.

	Command or Action	Purpose
	Example: `Router# show ip rsvp sender detail`	
Step 5	**show mpls traffic-eng link-management bandwidth-allocation** [**summary**] [*interface-type interface-number*] **Example:** `Router# show mpls traffic-eng link-management bandwidth-allocation`	Displays current local link information.
Step 6	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode and returns to user EXEC mode.

Configuration Examples for MPLS TE - Tunnel-Based Admission Control

Example Configuring TBAC

Note You must have an MPLS TE tunnel already configured in your network. For detailed information, see the "MPLS Traffic Engineering (TE)--Automatic Bandwidth Adjustment for TE Tunnels" module.

The following example enables RSVP and MPLS TE globally on a router and then configures a tunnel interface and bandwidth of 7500 kbps on the tunnel interface traversed by the RSVP flows:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.

Router(config)# ip rsvp qos

Router(config)# mpls traffic-eng tunnels

Router(config)# interface tunnel 1

Router(config-if)# ip rsvp bandwidth 7500

Router(config-if)# end
```

Example Configuring RSVP Local Policy on a Tunnel Interface

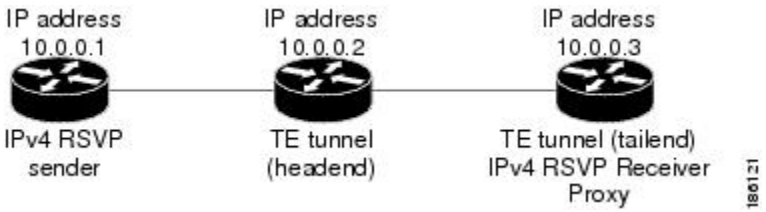
The following example configures an RSVP default local policy on a tunnel interface:

```
Router# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.

Router(config)# interface tunnel 1

Router(config-if)# ip rsvp policy local default

Router(config-rsvp-local-if-policy)# max bandwidth single 10

Router(config-rsvp-local-if-policy)# forward all

Router(config-rsvp-local-if-policy)# end
```

Example Verifying the TBAC Configuration

The figure below shows a network in which TBAC is configured.

Figure 106: Sample TBAC Network



The following example verifies that RSVP and MPLS TE are enabled and coexist on the headend router (10.0.0.2 in the figure above):

```
Router# show ip rsvp
RSVP: enabled (on 3 interface(s))
  RSVP QoS enabled <-------
  MPLS/TE signalling enabled <------
Signalling:
   Refresh interval (msec): 30000
   Refresh misses: 4
.
.
.
```

The following example verifies that RSVP and MPLS TE are enabled and coexist on the tailend router (10.0.0.3 in the figure above):

```
Router# show ip rsvp
RSVP: enabled (on 3 interface(s))
  RSVP QoS enabled <-------
  MPLS/TE signalling enabled <------
Signalling:
   Refresh interval (msec): 30000
```

```
   Refresh misses: 4
.
.
.
```

The following examples verify that an IPv4 flow is traveling through a TE tunnel (a label-switched path [LSP]) on the headend router (10.0.0.2 in the figure above):

```
Router# show ip rsvp sender
To              From          Pro DPort Sport Prev Hop       I/F      BPS
10.0.0.3        10.0.0.1      UDP 2     2     10.0.0.1       Et0/0    10K <-- IPv4 flow
10.0.0.3        10.0.0.2      0   1     11    none           none     100K <-- TE tunnel
```

```
Router# show ip rsvp reservation
To            From        Pro DPort Sport Next Hop     I/F    Fi Serv BPS
10.0.0.3      10.0.0.1    UDP 2     2     10.0.0.3     Tu1    SE RATE 10K <-- IPv4 flow
10.0.0.3      10.0.0.2    0   1     11    10.1.0.2     Et1/0  SE LOAD 100K <-- TE tunnel
```

The following examples verify that an IPv4 flow is traveling through a TE tunnel (LSP) on the tailend router (10.0.0.3 in the figure above):

```
Router# show ip rsvp sender
To              From          Pro DPort Sport Prev Hop       I/F      BPS
10.0.0.3        10.0.0.1      UDP 2     2     10.0.0.2       Et1/0    10K <-- IPv4 flow
10.0.0.3        10.0.0.2      0   1     11    10.1.0.1       Et1/0    100K <-- TE tunnel
```

```
Router# show ip rsvp reservation
To            From          Pro DPort Sport Next Hop   I/F    Fi Serv BPS
10.0.0.3      10.0.0.1      UDP 2     2     none       none   SE RATE 10K <-- IPv4 flow
10.0.0.3      10.0.0.2      0   1     11    none       none   SE LOAD 100K <-- TE tunnel
```

The following examples display detailed information about the IPv4 flow and the TE tunnel (LSP) on the headend router (10.0.0.2 in the figure above):

```
Router# show ip rsvp sender detail
PATH: <---------------------------------------------- IPv4 flow information begins here.
  Destination 10.0.0.3, Protocol_Id 17, Don't Police , DstPort 2
  Sender address: 10.0.0.1, port: 2
  Path refreshes:
    arriving: from PHOP 10.0.0.10 on Et0/0 every 30000 msecs. Timeout in 189 sec
  Traffic params - Rate: 10K bits/sec, Max. burst: 10K bytes
    Min Policed Unit: 0 bytes, Max Pkt Size 2147483647 bytes
  Path ID handle: 02000412.
  Incoming policy: Accepted. Policy source(s): Default
  Status:
  Output on Tunnel1, out of band. Policy status: Forwarding. Handle: 0800040E <--- TE tunnel
 verified
    Policy source(s): Default
  Path FLR: Never repaired
PATH: <---------------------------------------------- TE tunnel information begins here.
  Tun Dest:   10.0.0.3  Tun ID: 1  Ext Tun ID: 10.0.0.2
  Tun Sender: 10.0.0.2  LSP ID: 11
  Path refreshes:
    sent:     to   NHOP 10.1.0.2 on GigabitEthernet1/0/0
   .
   .

.

Router# show ip rsvp reservation detail
```

```
RSVP Reservation. Destination is 10.0.0.3, Source is 10.0.0.1,<--- IPv4 flow information
begins here.
  Protocol is UDP, Destination port is 2, Source port is 2
  Next Hop: 10.0.0.3 on Tunnel1, out of band <-------------------- TE tunnel verified
  Reservation Style is Shared-Explicit, QoS Service is Guaranteed-Rate
   .
   .
   .
Reservation: <-------------------------------------- TE Tunnel information begins here.
  Tun Dest:   10.0.0.3  Tun ID: 1  Ext Tun ID: 10.0.0.2
  Tun Sender: 10.0.0.2  LSP ID: 11
  Next Hop: 10.1.0.2 on GigabitEthernet1/0/0
  Label: 0 (outgoing)
  Reservation Style is Shared-Explicit, QoS Service is Controlled-Load
   .
   .
   .


Router# show ip rsvp installed detail

RSVP: GigabitEthernet0/0/0 has no installed reservations

RSVP: GigabitEthernet1/0/0 has the following installed reservations
RSVP Reservation. Destination is 10.0.0.3. Source is 10.0.0.2,
  Protocol is 0  , Destination port is 1, Source port is 11
  Traffic Control ID handle: 03000405
  Created: 04:46:55 EST Fri Oct 26 2007 <------ IPv4 flow information
  Admitted flowspec:
    Reserved bandwidth: 100K bits/sec, Maximum burst: 1K bytes, Peak rate: 100K bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 1500 bytes
  Resource provider for this flow: None
   .
   .
   .
RSVP: Tunnel1 has the following installed reservations <------ TE tunnel verified
RSVP Reservation. Destination is 10.0.0.3. Source is 10.0.0.1,
  Protocol is UDP, Destination port is 2, Source port is 2
  Traffic Control ID handle: 01000415
  Created: 04:57:07 EST Fri Oct 26 2007 <----- IPv4 flow information
  Admitted flowspec:
    Reserved bandwidth: 10K bits/sec, Maximum burst: 10K bytes, Peak rate: 10K bits/sec
    Min Policed Unit: 0 bytes, Max Pkt Size: 0 bytes
  Resource provider for this flow: None
   .
   .
   .


Router# show ip rsvp interface detail


 Et0/0:
   RSVP: Enabled
   Interface State: Up
   Bandwidth:
     Curr allocated: 0 bits/sec
     Max. allowed (total): 3M bits/sec
     Max. allowed (per flow): 3M bits/sec
   .
   .
   .
 Et1/0:
   RSVP: Enabled
   Interface State: Up
```

```
   Bandwidth:
     Curr allocated: 0 bits/sec
     Max. allowed (total): 3M bits/sec
     Max. allowed (per flow): 3M bits/sec
   .
   .
   .
Tu1: <--------------------------------- TE tunnel information begins here.
   RSVP: Enabled
   RSVP aggregation over MPLS TE: Enabled
   Interface State: Up
   Bandwidth:
     Curr allocated: 20K bits/sec
     Max. allowed (total): 3M bits/sec
     Max. allowed (per flow): 3M bits/sec
   .
   .
   .
```

The following examples display detailed information about the IPv4 flow and the TE tunnel (LSP) on the tailend router (10.0.0.3 in the figure above):

```
Router# show ip rsvp sender detail
PATH: <---------------------------------------------- IPv4 flow information begins here.
  Destination 10.0.0.3, Protocol_Id 17, Don't Police , DstPort 2
  Sender address: 10.0.0.1, port: 2
  Path refreshes:
    arriving: from PHOP 10.0.0.2 on Et1/0 every 30000 msecs, out of band. Timeout in 188
sec
  Traffic params - Rate: 10K bits/sec, Max. burst: 10K bytes
    Min Policed Unit: 0 bytes, Max Pkt Size 2147483647 bytes
   .
   .
   .
PATH: <------------------------------------------------ TE tunnel information begins here.
  Tun Dest:   10.0.0.3  Tun ID: 1  Ext Tun ID: 10.0.0.2
  Tun Sender: 10.0.0.2  LSP ID: 11
  Path refreshes:
    arriving: from PHOP 10.1.0.1 on Et1/0 every 30000 msecs. Timeout in 202 sec
   .
   .
   .


Router# show ip rsvp reservation detail

RSVP Reservation. Destination is 10.0.0.3, Source is 10.0.0.1, <--- IPv4 flow information
begins here.
  Protocol is UDP, Destination port is 2, Source port is 2
  Next Hop: none
  Reservation Style is Shared-Explicit, QoS Service is Guaranteed-Rate
   .
   .
   .

Reservation: <------------------------------------- TE tunnel information begins here.
  Tun Dest:   10.0.0.3  Tun ID: 1  Ext Tun ID: 10.0.0.2
  Tun Sender: 10.0.0.2  LSP ID: 11
  Next Hop: none
  Label: 1 (outgoing)
  Reservation Style is Shared-Explicit, QoS Service is Controlled-Load
   .
   .
   .
```

```
Router# show ip rsvp request detail

RSVP Reservation. Destination is 10.0.0.3, Source is 10.0.0.1,
  Protocol is UDP, Destination port is 2, Source port is 2
  Prev Hop: 10.0.0.2 on GigabitEthernet1/0/0, out of band <-------------- TE tunnel verified

  Reservation Style is Shared-Explicit, QoS Service is Guaranteed-Rate
  Average Bitrate is 10K bits/sec, Maximum Burst is 10K bytes
   .
   .
   .

Request: <----------------------------------- TE tunnel information begins here.
  Tun Dest:   10.0.0.3  Tun ID: 1  Ext Tun ID: 10.0.0.2
  Tun Sender: 10.0.0.2  LSP ID: 11
  Prev Hop: 10.1.0.1 on GigabitEthernet1/0/0
  Label: 0 (incoming)
  Reservation Style is Shared-Explicit, QoS Service is Controlled-Load
   .
   .
   .
```

Example Verifying the RSVP Local Policy Configuration

The following example verifies that a default local policy has been configured on tunnel interface 1:

```
Device# show run interface tunnnel 1
Building configuration...

Current configuration : 419 bytes
!
interface Tunnel1
 bandwidth 3000
 ip unnumbered Loopback0
 tunnel destination 10.0.0.3
 tunnel mode mpls traffic-eng
 tunnel mpls traffic-eng autoroute announce
 tunnel mpls traffic-eng priority 1 1
 tunnel mpls traffic-eng bandwidth 100
 tunnel mpls traffic-eng path-option 1 dynamic
 tunnel mpls traffic-eng fast-reroute
 ip rsvp policy local default <---------------- Local policy information begins here.
  max bandwidth single 10
  forward all
 ip rsvp bandwidth 3000
end


The following example provides additional information about the default local policy
configured on tunnel interface 1:
Device# show ip rsvp policy local detail
Tunnel1:
  Default policy:

    Preemption Scope: Unrestricted.
    Local Override:   Disabled.
    Fast ReRoute:     Accept.
    Handle:           BC000413.

                          Accept              Forward
    Path:                 Yes                 Yes
```

```
    Resv:                  Yes                 Yes
    PathError:             Yes                 Yes
    ResvError:             Yes                 Yes

                           Setup Priority      Hold Priority
    TE:                    N/A                 N/A
    Non-TE:                N/A                 N/A

                           Current             Limit
    Senders:               0                   N/A
    Receivers:             1                   N/A
    Conversations:         1                   N/A
    Group bandwidth (bps): 10K                 N/A
    Per-flow b/w (bps):    N/A                 10K

Generic policy settings:
    Default policy: Accept all
    Preemption:     Disabled
```

Additional References

The following sections provide references related to the RSVP--VRF Lite Admission Control feature.

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
VRF-related internet draft	*Support for RSVP in Layer 3 VPNs*, Internet draft, November 19, 2007 [draft-davie-tsvwg-rsvp-l3vpn-01.txt]
Cisco IOS commands	Cisco IOS Master Commands List, All Releases

Standards

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

MIBs

MIB	MIBs Link
No new or modified MIBs are supported by this feature, and support for existing MIBs has not been modified by this feature.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	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 Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for MPLS TE - Tunnel-Based Admission Control

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 83: Feature Information for MPLS TE--Tunnel-Based Admission Control (TBAC)

Feature Name	Releases	Feature Information
MPLS TE Tunnel-Based Admission Control	Cisco IOS XE Release 2.6	The MPLS TE--Tunnel-Based Admission Control feature enables classic Resource Reservation Protocol (RSVP) unicast reservations that are traveling across an MPLS TE core to be aggregated over an MPLS TE tunnel. The following commands were introduced or modified: **ip rsvp qos**, **show ip rsvp**, **show ip rsvp reservation**, **show ip rsvp sender**, **show mpls traffic-eng link-management bandwidth-allocation**.

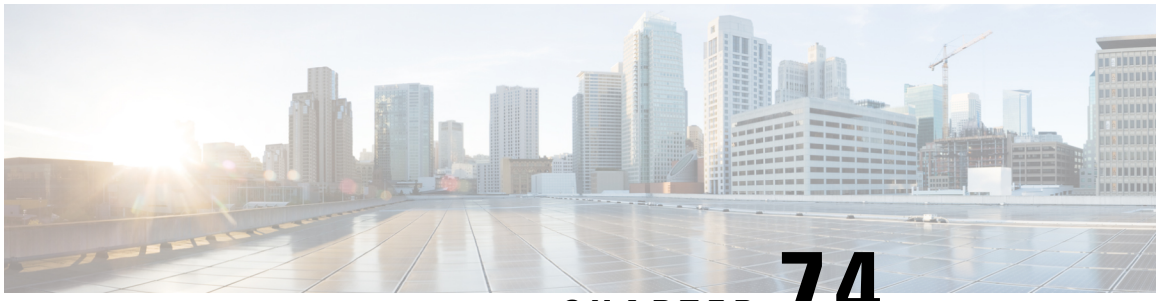
Glossary

admission control --The process by which an RSVP reservation is accepted or rejected on the basis of end-to-end available network resources.

QoS --quality of service. A measure of performance for a transmission system that reflects its transmission quality and service availability. Quality of service focuses on achieving appropriate network performance for networked applications; it is superior to best effort performance.

RSVP --Resource Reservation Protocol. A protocol that supports the reservation of resources across an IP network. Applications that run on IP end systems can use RSVP to indicate to other nodes the nature (bandwidth, jitter, maximum burst, and so on) of the packet streams that they want to receive.

VRF --virtual routing and forwarding. An extension of IP routing that provides multiple routing instances. A VRF consists of an IP routing table, a derived forwarding table, a set of interfaces that use the forwarding table, and a set of rules and routing protocols that determine what goes into the forwarding table. In general, a VRF includes the routing information that defines a customer VPN site that is attached to a provider edge (PE) device.



CHAPTER 74

PfR RSVP Control

The PfR RSVP Control feature introduces the ability to perform application-aware path selection for traffic that is controlled by Resource Reservation Protocol (RSVP). This feature allows RSVP flows to be learned by Performance Routing (PfR) and protocol Path messages to be redirected after the PfR primary controller determines the best exit using PfR policies.

Information About PfR RSVP Control

PfR and RSVP Control

The PfR RSVP Control feature introduces the ability for Performance Routing (PfR) to learn, monitor, and optimize Resource Reservation Protocol (RSVP) flows. PfR is an integrated Cisco IOS solution that allows you to monitor IP traffic flows and then define policies and rules based on traffic class performance, link load distribution, link bandwidth monetary cost, and traffic type. PfR provides active and passive monitoring systems, dynamic failure detection, and automatic path correction. Deploying PfR enables intelligent load distribution and optimal route selection in an enterprise network that uses multiple ISP or WAN connections at the network edge.

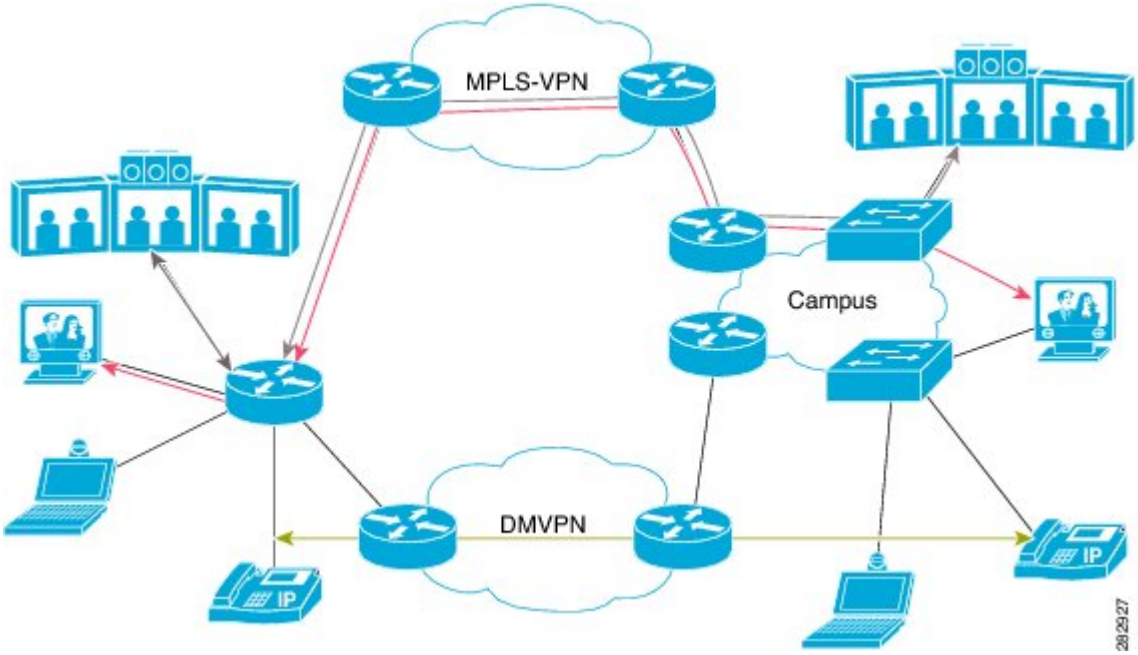
PfR can monitor and control applications and prefixes that are configured or learned by observing traffic that is flowing on the network. The controller is a centralized policy decision point at which policies are defined and applied to various traffic classes that traverse the border routers (BRs). The controller can be configured to learn and control traffic classes on the network. The controller makes exit selections and instructs the BRs to enforce the exit selection. While the current PfR implementation can be used to optimize voice/video traffic, the control exercised by PfR is not aware of technologies such as RSVP. The PfR RSVP integration will help RSVP leverage the application-specific control of routes that PfR can provide.

RSVP is a standards-based control protocol that allows for resources to be reserved to allow for better reliability for voice/video traffic. RSVP achieves this by signaling the traffic profile before the actual data flow to reserve resources for the data flow. Establishing end-to-end resource reservations along a media path allows RSVP to guarantee that resources are available when they are needed. RSVP consults the forwarding plane database (or CEF) in order to achieve path congruency with the media flow. The routes in the CEF database are mostly

dictated by the routing protocols where the only metric for determining the best route is the cumulative cost of the links on that path.

In the diagram shown below, there are two paths for the network on the left to reach the campus network on the right. One path uses the DMVPN cloud, and the other path uses the MPLS-VPN cloud. Depending on the speed and bandwidth required, it might make sense to route video applications over the MPLS-VPN network while routing voice applications over the DMVPN network. Such kind of application-aware path selection is not possible in CEF, but PfR can determine the best path for specific application traffic based on performance criteria.

Figure 107: Application-Aware Path Selection



With the RSVP integration, PfR will learn, monitor, and optimize RSVP flows. RSVP is included as a new learn source. PfR will learn RSVP flows that traverse internal and external interfaces. Each RSVP flow is learned as a PfR traffic class and is controlled independently of the other RSVP flows. While filtering of the learned flows is supported with prefix lists and route maps, aggregating RSVP flows is not advised. The PfR controller chooses a best exit based on the configured PfR policies and installs route maps to redirect traffic. If any of the RSVP flows enters an Out-of-Policy (OOP) condition, PfR will find and switch the RSVP flow to a new exit. RSVP will reinstall the reservation on the new path at the time of refresh (usually within a span of 30 seconds) or as a Fast Local Repair (FLR) case in less than 5 seconds.

The intent of the PfR RSVP Control feature is to identify and install route maps at the time the router receives an RSVP Path message. The route map captures the data traffic, while RSVP uses this path for the Path message.

RSVP flows are learned as PfR traffic classes defined as a single application flow that can be identified by the source address, source port, destination address, destination port and IP protocol. This microflow is optimized as an application by PfR, and a dynamic policy route is created by PfR to forward this traffic class over the selected exit.

All RSVP flows are optimized only after PfR checks that there is enough bandwidth on the exit that is being considered. This information is pushed periodically from the BRs to the MC. On the BR itself, RSVP notifies PfR every time the bandwidth pool on an interface changes.

Equivalent-Path Round-Robin Resolver

PfR introduced a new resolver with the PfR RSVP Control feature. PfR, by default, uses a random resolver to decide between equivalent paths, exits with the same cost determined by the PfR policies. When the round-robin resolver is configured using the **equivalent-path-round-robin** command, the next exit (next-hop interface) is selected and compared to the running PfR policy. The round-robin resolver is handed an array of equivalent exits from which it chooses in a round-robin fashion. Exits are pruned in the same fashion they are today by each resolver. If the exit matches the policy, the exit becomes the best exit. The round-robin resolver does not do any specific RSVP checking. To return to using the random resolver, enter the no form of the **equivalent-path-round-robin** command.

Any PfR traffiic class can use the round-robin resolver, and it provides a load-balancing scheme for multiple equivalent paths as determined by PfR policy.

RSVP Post Dial Delay Timer for Best Path Selection

In the PfR RSVP Control feature, the **rsvp post-dial-delay** command was introduced to set a value for the RSVP post dial delay timer that runs on the border routers when RSVP flow learning is enabled on a PfR controller. The timer is updated on the border routers at the start of every PfR learn cycle, and the timer determines the delay, in milliseconds, before the routing path is returned to RSVP. When the PfR and RSVP integration is enabled, PfR tries to locate a best path for any RSVP flows that are learned before the delay timer expires. If the current path is not the best path, PfR attempts to install the new path. RSVP reacts to this policy route injection as a case of Fast Local Repair (FLR) and resignals a new reservation path.

RSVP Signaling Retries for Alternative Reservation Path

The PfR RSVP Control feature introduced a new command, **rsvp signaling-retries**, which is configured on a controller and is used to instruct PfR to provide an alternate reservation path when an RSVP reservation returns an error condition. If an alternate path is provided by PfR, RSVP can resend the reservation signal. The default number of retries is set to 0; no signaling retries are to be permitted, and a reservation error message is sent when a reservation failure occurs.

Performance Statistics from PfR Commands

The PfR controller learns and monitors IP traffic that flows through the border routers, and the master controller selects the best exit for a traffic flow based on configured policies and the performance information received from the border routers. To view some of the performance data collected by the controller, use the following commands:

- **show pfr master active-probes**
- **show pfr master border**
- **show pfr master exits**
- **show pfr master statistics**
- **show pfr master traffic-class**
- **show pfr master traffic-class performance**

All these commands are entered at the controller, and some of the commands have keywords and arguments to filter the output. For detailed information about these commands, see the Cisco IOS Performance Routing Command Reference.

How to Configure PfR RSVP Control

Configuring PfR RSVP Control Using a Learn List

Perform this task on the controller to define a learn list that contains traffic classes that are automatically learned based on RSVP flows and filtered by a prefix list. In this task, the goal is to optimize all video traffic that is learned from RSVP flows.

The VIDEO traffic class is defined as any prefix that matches 10.100.0.0/16 or 10.200.0.0/16 and a PfR policy, named POLICY_RSVP_VIDEO, is created.

The learn lists are referenced in a PfR policy using a PfR map and are activated using the **policy-rules** (PfR) command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network/length* | **permit** *network/length*}
4. **pfr master**
5. **policy-rules** *map-name*
6. **rsvp signaling-retries** *number*
7. **rsvp post-dial-delay** *msecs*
8. **learn**
9. **list seq** *number* **refname** *refname*
10. **traffic-class prefix-list** *prefix-list-name* [**inside**]
11. **rsvp**
12. **exit**
13. Repeat Step 9 to Step 12 to configure additional learn lists.
14. **exit**
15. Use the **exit** command as necessary to return to global configuration mode.
16. **pfr-map** *map-name sequence-number*
17. **match pfr learn list** *refname*
18. **set mode route control**
19. **set resolve equivalent-path-round-robin**
20. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**ip prefix-list** *list-name* [**seq** *seq-value*] {**deny** *network/length* | **permit** *network/length*} **Example:** `Router(config)# ip prefix-list RSVP_VIDEO seq 10 permit 10.100.0.0/16`	Creates an IP prefix list to filter prefixes for learning. • An IP prefix list is used under learn list configuration mode to filter IP addresses that are learned. • The example creates an IP prefix list named RSVP_VIDEO for PfR to profile the prefix, 10.100.0.0/16.
Step 4	**pfr master** **Example:** `Router(config)# pfr master`	Enters the PfR master controller configuration mode to configure a Cisco router as the primary controller and to configure the master controller policy and the timer settings.
Step 5	**policy-rules** *map-name* **Example:** `Router(config-pfr-mc)# policy-rules POLICY_RSVP_VIDEO`	Selects a PfR map and applies the configuration under the PfR master controller configuration mode. • Use the *map-name* argument to specify the PfR map name to be activated. • The example applies the PfR map named POLICY_RSVP_VIDEO which includes the learn list configured in this task.
Step 6	**rsvp signaling-retries** *number* **Example:** `Router(config-pfr-mc)# rsvp signaling-retries 1`	Specifies the number of alternate paths that PfR provides for an RSVP reservation when a reservation error condition is detected. • Use the *number* argument to specify the number of alternate paths. • The example configured in this task shows how to configure PfR to set the number of alternate paths for RSVP signaling retries to 1.
Step 7	**rsvp post-dial-delay** *msecs* **Example:** `Router(config-pfr-mc)# rsvp post-dial-delay 100`	Configures the RSVP post dial delay timer to set the delay before PfR returns the routing path to RSVP. • Use the *msecs* argument to specify the delay, in milliseconds. • The example configured in this task shows how to configure PfR to set the RSVP post dial delay to 100 milliseconds.

	Command or Action	Purpose
Step 8	**learn** **Example:** `Router(config-pfr-mc)# learn`	Enters PfR Top Talker and Top Delay learning configuration mode to automatically learn traffic classes.
Step 9	**list seq** *number* **refname** *refname* **Example:** `Router(config-pfr-mc-learn)# list seq 10 refname LEARN_RSVP_VIDEO`	Creates a PfR learn list and enters learn list configuration mode. • Use the **seq** keyword and *number* argument to specify a sequence number used to determine the order in which learn list criteria are applied. • Use the **refname** keyword and *refname* argument to specify a reference name for the learn list. • The example creates a learn list named LEARN_RSVP_VIDEO.
Step 10	**traffic-class prefix-list** *prefix-list-name* [**inside**] **Example:** `Router(config-pfr-mc-learn-list)# traffic-class prefix-list RSVP_VIDEO`	Configures the primary controller to automatically learn traffic based only on destination prefixes. • Use the *prefix-list-name* argument to specify a prefix list. • The example defines a traffic class using the prefix list named RSVP_VIDEO.
Step 11	**rsvp** **Example:** `Router(config-pfr-mc-learn-list)# rsvp`	Configures the primary controller to learn the top prefixes based on RSVP flows. • When this command is enabled, the primary controller learns the top prefixes across all border routers according to the highest outbound throughput. • The example configures the primary controller to learn the top prefixes based on RSVP flows for the LEARN_RSVP_VIDEO learn list.
Step 12	**exit** **Example:** `Router(config-pfr-mc-learn-list)# exit`	Exits learn list configuration mode, and returns to PfR Top Talker and Top Delay learning configuration mode.
Step 13	Repeat Step 9 to Step 12 to configure additional learn lists.	--
Step 14	**exit** **Example:** `Router(config-pfr-mc-learn)# exit`	Exits PfR Top Talker and Top Delay learn configuration mode, and returns to the PfR master controller configuration mode.

	Command or Action	Purpose
Step 15	Use the **exit** command as necessary to return to global configuration mode.	--
Step 16	**pfr-map** *map-name sequence-number* **Example:** `Router(config)# pfr-map POLICY_RSVP_VIDEO 10`	Enters PfR map configuration mode to configure a PfR map. • The example creates a PfR map named POLICY_RSVP_VIDEO.
Step 17	**match pfr learn list** *refname* **Example:** `Router(config-pfr-map)# match pfr learn list LEARN_RSVP_VIDEO`	Creates a match clause entry in a PfR map to match PfR-learned prefixes. • Only one match clause can be configured for each PfR map sequence. • The example defines a traffic class using the criteria defined in the PfR learn list named LEARN_RSVP_VIDEO. **Note** Only the syntax relevant to this task is used here.
Step 18	**set mode route control** **Example:** `Router(config-pfr-map)# set mode route control`	Creates a set clause entry to configure route control for matched traffic. • In the control mode, the primary controller analyzes monitored prefixes and implements changes based on policy parameters.
Step 19	**set resolve equivalent-path-round-robin** **Example:** `Router(config-pfr-map)# set resolve equivalent-path-round-robin`	Creates a set clause entry to specify the use of the equivalent-path round-robin resolver. • In this task, the equivalent-path round-robin resolver is used to choose between equivalent paths instead of the random resolver.
Step 20	**end** **Example:** `Router(config-pfr-map)# end`	(Optional) Exits PfR map configuration mode and returns to privileged EXEC mode.

Displaying PfR RSVP Control Information

Although the PfR RSVP Control feature is configured on the primary controller, the border routers actually collect the performance information, and there are **show** and **debug** commands available to display the RSVP information for both the primary controller and border routers. The first few commands in this task are entered on the primary controller and, for the rest of the commands, there is a step to move to a border router through which the application traffic is flowing. These **show** and **debug** commands can be entered in any order.

SUMMARY STEPS

1. **enable**
2. **show pfr master traffic-class** [**rsvp**] [**active** | **passive** | **status**] [**detail**]
3. **show pfr master policy** [*sequence-number* | *policy-name* | **default** | **dynamic**]
4. **debug pfr master rsvp**
5. Move to a border router through which the RSVP traffic is flowing.
6. **enable**
7. **show pfr border rsvp**
8. **show pfr border routes rsvp-cache**
9. **debug pfr border rsvp**

DETAILED STEPS

Step 1 enable

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Router> enable
```

Step 2 show pfr master traffic-class [rsvp] [active | passive | status] [detail]

This command is used to display information about PfR traffic classes that are learned as RSVP traffic classes.

Example:

```
Router# show pfr master traffic-class rsvp

OER Prefix Statistics:
 Pas - Passive, Act - Active, S - Short term, L - Long term, Dly - Delay (ms),
 P - Percentage below threshold, Jit - Jitter (ms),
 MOS - Mean Opinion Score
 Los - Packet Loss (packets-per-million), Un - Unreachable (flows-per-million),
 E - Egress, I - Ingress, Bw - Bandwidth (kbps), N - Not applicable
 U - unknown, * - uncontrolled, + - control more specific, @ - active probe all
 # - Prefix monitor mode is Special, & - Blackholed Prefix
 % - Force Next-Hop, ^ - Prefix is denied

DstPrefix           Appl_ID Dscp Prot     SrcPort     DstPort SrcPrefix
          Flags             State     Time            CurrBR  CurrI/F Protocol
         PasSDly  PasLDly   PasSUn   PasLUn  PasSLos  PasLLos      EBw      IBw
         ActSDly  ActLDly   ActSUn   ActLUn  ActSJit  ActPMOS  ActSLos  ActLLos
------------------------------------------------------------------------------
10.1.0.10/32              N    N  tcp       75-75       75-75 10.1.0.12/32
                          INPOLICY       @0         10.1.0.24 Tu24            PBR
               U        U        0        0        0        0        0        0
               1        1        0        0        N        N        N        N
```

Step 3 show pfr master policy [sequence-number | policy-name | default | dynamic]

This command is used to display policy information. The following example uses the **dynamic** keyword to display the policies dynamically created by provider applications. Note the RSVP configuration commands.

Example:

```
Router# show pfr master policy dynamic

Dynamic Policies:

  proxy id 10.3.3.3
  sequence no. 18446744069421203465, provider id 1001, provider priority 65535
    host priority 65535, policy priority 101, Session id 9
  backoff 90 90 90
  delay relative 50
  holddown 90
  periodic 0
  probe frequency 56
  mode route control
  mode monitor both
  mode select-exit good
  loss relative 10
  jitter threshold 20
  mos threshold 3.60 percent 30
  unreachable relative 50
  next-hop not set
  forwarding interface not set
  resolve delay priority 11 variance 20
  resolve utilization priority 12 variance 20
  proxy id 10.3.3.3
  sequence no. 18446744069421269001, provider id 1001, provider priority 65535
    host priority 65535, policy priority 102, Session id 9
  backoff 90 90 90
  delay relative 50
  holddown 90
  periodic 0
  probe frequency 56
  mode route control
  mode monitor both
  mode select-exit good
  loss relative 10
  jitter threshold 20
  mos threshold 3.60 percent 30
  unreachable relative 50
  next-hop not set
  forwarding interface not set
  resolve delay priority 11 variance 20
  resolve utilization priority 12 variance 20
  proxy id 10.3.3.4
  sequence no. 18446744069421334538, provider id 1001, provider priority 65535
    host priority 65535, policy priority 103, Session id 10
  backoff 90 90 90
  delay relative 50
  holddown 90
  periodic 0
  probe frequency 56
  mode route control
  mode monitor both
  mode select-exit good
  loss relative 10
  jitter threshold 20
  mos threshold 3.60 percent 30
  unreachable relative 50
  next-hop not set
  forwarding interface not set
  resolve delay priority 11 variance 20
  resolve utilization priority 12 variance 20
```

Step 4 **debug pfr master rsvp**

Displays debugging information about PfR RSVP events on the PfR controller.

Example:

```
Router# debug pfr master rsvp

Jan 23 21:18:19.439 PST: PFR_MC_RSVP: recvd a RSVP flow
Jan 23 21:18:19.439 PST: PFR_MC_RSVP: Processing 1 rsvp flows
Jan 23 21:18:19.439 PST: PFR_MC_RSVP: Resolve: src: 10.1.0.12 dst: 10.1.25.19 pr
oto: 17 sport min: 1 sport max: 1 dport min: 1 dport max: 1 from BR 10.1.0.23
Jan 23 21:18:19.439 PST: PFR_MC_RSVP: Marking: 10.1.0.23, FastEthernet1/0
Jan 23 21:18:19.439 PST: %OER_MC-5-NOTICE: Uncontrol Prefix 10.1.25.19/32, Probe frequency changed
Jan 23 21:18:19.439 PST: PFR_MC_RSVP: Marked: 10.1.0.23, FastEthernet1/0 as current
Jan 23 21:18:19.467 PST: PFR_MC_RSVP: recv new pool size
Jan 23 21:18:19.467 PST: PFR_MC_RSVP: Update from 10.1.0.23, Fa1/0: pool 8999
Jan 23 21:18:20.943 PST: %OER_MC-5-NOTICE: Prefix Learning WRITING DATA
Jan 23 21:18:21.003 PST: %OER_MC-5-NOTICE: Prefix Learning STARTED
Jan 23 21:18:22.475 PST: PFR_MC_RSVP: RSVP resolver invoked
Jan 23 21:18:22.475 PST: PFR RSVP MC:  10.1.25.19/32 Appl 17 [1, 1][1, 1] 0:
        BR 10.1.0.23, Exit Fa1/0, is current exit
Jan 23 21:18:22.475 PST: PFR RSVP MC:  10.1.25.19/32 Appl 17 [1, 1][1, 1] 0:
        BR 10.1.0.23, Exit Fa1/0, is current exit
Jan 23 21:18:22.475 PST: PFR_MC_RSVP: BR:10.1.0.23 Exit:Fa1/0pool size : 8999
est : 8999 tc->tspec: 1, fit: 8999
Jan 23 21:18:22.475 PST: PFR_MC_RSVP: BR:10.1.0.24 Exit:Tu24pool size : 9000
est : 9000 tc->tspec: 1, fit: 8999
Jan 23 21:18:22.475 PST: PFR_MC_RSVP: BR:10.1.0.23 Exit:Fa1/1pool size : 9000
est : 9000 tc->tspec: 1, fit: 8999
```

Step 5 Move to a border router through which the RSVP traffic is flowing.

Step 6 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Router> enable
```

Step 7 **show pfr border rsvp**

The following example shows information about the current values for the RSVP post dial timeout timer and signaling retries on a PfR border router:

Example:

```
Router# show pfr border rsvp

PfR BR RSVP parameters:
       RSVP Signaling retries:         1
       Post-dial-timeout(msec):        0
```

Step 8 **show pfr border routes rsvp-cache**

This command is used to show all the RSVP paths that PfR is aware of.

Note Only syntax appropriate to this example is shown.

Example:

```
Router# show pfr border routes rsvp-cache
```

```
SrcIP       DstIP       Protocol Src_port Dst_port Nexthop       Egress I/F PfR/RIB
----------- ----------- -------- -------- -------- ------------- ---------- --------
10.1.25.19  10.1.35.5   UDP      1027     1027     10.1.248.5    Gi1/0       RIB*
10.1.0.12   10.1.24.10  UDP      48       48       10.1.248.24   Gi1/0       PfR*
10.1.0.12   10.1.42.19  UDP      23       23       10.1.248.24   Gi1/0       PfR*
10.1.0.12   10.1.18.10  UDP      12       12       172.16.43.2   Fa1/1       PfR*
```

Step 9 **debug pfr border rsvp**

Displays debugging information about PfR RSVP events on a PfR border router.

Example:

```
Router# debug pfr border rsvp

Jan 23 21:18:19.434 PST: PfR RSVP:RESOLVE called for src: 10.1.0.12 dst: 10.1.25.19
 proto: 17 sport: 1 dport: 1; tspec 1
Jan 23 21:18:19.434 PST: PfR RSVP:hash index = 618
Jan 23 21:18:19.434 PST: PfR RSVP:Searching flow: src: 10.1.0.12 dst: 10.1.25.19
 proto: 17 sport: 1 dport: 1
Jan 23 21:18:19.434 PST: PfR RSVP:Add flow: src: 10.1.0.12 dst: 10.1.25.19
 proto: 17 sport: 1 dport: 1
Jan 23 21:18:19.434 PST: PfR RSVP:hash index = 618
Jan 23 21:18:19.434 PST: PfR RSVP:Searching flow: src: 10.1.0.12 dst: 10.1.25.19
 proto: 17 sport: 1 dport: 1
Jan 23 21:18:19.434 PST: PfR RSVP:hash index = 618
Jan 23 21:18:19.434 PST: PfR RSVP:successfully added the flow to the db
Jan 23 21:18:19.434 PST: PfR RSVP:flow: src: 10.1.0.12 dst: 10.1.25.19
 proto: 17 sport: 1 dport: 1 lookup; topoid: 0
Jan 23 21:18:19.434 PST: PfR RSVP(det):ret nh: 10.185.252.1, idb: 35
Jan 23 21:18:19.434 PST: PfR RSVP:Adding new context
Jan 23 21:18:19.434 PST: PfR RSVP(det):Num contexts: 0
Jan 23 21:18:19.434 PST: PfR RSVP(det):Num contexts: 1
Jan 23 21:18:19.434 PST: PfR RSVP:flow src: 10.1.0.12 dst: 10.1.25.19
 proto: 17 sport: 1 dport: 1 now pending notify
Jan 23 21:18:19.434 PST: PfR RSVP:Resolve on flow: src: 10.1.0.12 dst: 10.1.25.19
 proto: 17 sport: 1 dport: 1
Jan 23 21:18:19.434 PST: PfR RSVP:Filtering flow: src: 10.1.0.12 dst: 10.1.25.19
 proto: 17 sport: 1 dport: 1
```

Displaying PfR Performance and Statistics Information

Enter the commands in this task to view more detailed performance or statistical information about PfR traffic classes or exits. The commands can be entered in any order within each section.

SUMMARY STEPS

1. **enable**
2. **show pfr master traffic-class** [**policy** *policy-seq-number* | *rc-protocol* | **state** {**hold**| **in** | **out** | **uncontrolled**}] [**detail**]
3. **show pfr master traffic-class performance** [**application** *application-name* [*prefix*] | **history** [**active** | **passive**] | **inside** | **learn** [**delay** | **inside** | **list** *list-name* | **rsvp** | **throughput**] | **policy** *policy-seq-number* | *rc-protocol* | **state** {**hold** | **in** | **out** | **uncontrolled**} | **static**] [**detail**]
4. **show pfr master exits**
5. **show pfr master active-probes** [**assignment** | **running**] [**forced** *policy-sequence-number* | **longest-match**]
6. **show pfr master border** [*ip-address*] [**detail** | **report** | **statistics** | **topology**]

7. **show pfr master statistics** [**active-probe** | **border** | **cc** | **exit** | **netflow** | **prefix** | **process** | **system** | **timers**]

DETAILED STEPS

Step 1 **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Router> enable
```

Step 2 **show pfr master traffic-class** [**policy** *policy-seq-number* | *rc-protocol* | **state** {**hold**| **in** | **out** | **uncontrolled**}] [**detail**]

This command is used to display information about traffic classes that are monitored and controlled by a PfR controller. In this example, the **state in** keywords are used to filter the output to show only traffic classes that are in an in-policy state.

Example:

```
Router# show pfr master traffic-class state in

OER Prefix Statistics:
 Pas - Passive, Act - Active, S - Short term, L - Long term, Dly - Delay (ms),
 P - Percentage below threshold, Jit - Jitter (ms),
 MOS - Mean Opinion Score
 Los - Packet Loss (packets-per-million), Un - Unreachable (flows-per-million),
 E - Egress, I - Ingress, Bw - Bandwidth (kbps), N - Not applicable
 U - unknown, * - uncontrolled, + - control more specific, @ - active probe all
 # - Prefix monitor mode is Special, & - Blackholed Prefix
 % - Force Next-Hop, ^ - Prefix is denied

DstPrefix           Appl_ID Dscp Prot     SrcPort     DstPort SrcPrefix
          Flags             State     Time            CurrBR  CurrI/F Protocol
        PasSDly  PasLDly   PasSUn   PasLUn  PasSLos  PasLLos      EBw      IBw
        ActSDly  ActLDly   ActSUn   ActLUn  ActSJit  ActPMOS  ActSLos  ActLLos
-------------------------------------------------------------------------------
10.1.0.0/24               N    N    N           N           N N
                          INPOLICY        0          10.1.1.1 Et0/0           BGP
              14       14        0        0        0        0       78        9
               N        N        N        N        N        N

10.2.0.0/24               N    N    N           N           N N
                          INPOLICY        0          10.1.1.2 Et0/0           BGP
              14       14        0        0        0        0       75        9
               N        N        N        N        N        N

10.3.0.0/24               N    N    N           N           N N
                          INPOLICY        0          10.1.1.3 Et0/0           BGP
              14       14        0        0        0        0       77        9
               N        N        N        N        N        N

10.4.0.0/24               N    N    N           N           N N
                          INPOLICY        0          10.1.1.4 Et0/0           BGP
              14       14        0        0        0        0       77        9
               N        N        N        N        N        N

10.1.8.0/24               N    N    N           N           N N
                          INPOLICY        0          10.1.1.3 Et0/0           BGP
              14       14    62500    73359        0        0        5        1
```

```
               N        N        N        N        N        N

10.1.1.0/24               N    N    N          N           N N
                          INPOLICY        0          10.1.1.2 Et0/0           BGP
              14       14     9635     9386     1605     1547       34        4
               N        N        N        N        N        N
```

Step 3 **show pfr master traffic-class performance** [**application** *application-name* [*prefix*] | **history** [**active** | **passive**] | **inside** | **learn** [**delay** | **inside** | **list** *list-name* | **rsvp** | **throughput**] | **policy** *policy-seq-number* | *rc-protocol* | **state** {**hold** | **in** | **out** | **uncontrolled**} | **static**] [**detail**]

This command displays performance information about traffic classes that are monitored and controlled by a PfR controller.

Note Only the syntax applicable to this example is shown.

Example:

The following output shows traffic-class performance history on current exits during the last 60 minutes.

```
Router# show pfr master traffic-class performance history

Prefix: 10.70.0.0/16
efix performance history records
 Current index 1, S_avg interval(min) 5, L_avg interval(min) 60

Age       Border          Interface       OOP/RteChg Reasons
Pas: DSum  Samples  DAvg  PktLoss  Unreach   Ebytes   Ibytes     Pkts    Flows
Act: Dsum Attempts  DAvg    Comps  Unreach   Jitter LoMOSCnt   MOSCnt
00:00:33  10.1.1.4        Et0/0
Pas: 6466      517    12        2       58  3400299   336921    10499     2117
Act:    0        0     0        0        0        N        N        N
00:01:35  10.1.1.4        Et0/0
Pas:15661     1334    11        4      157  4908315   884578    20927     3765
Act:    0        0     0        0        0        N        N        N
00:02:37  10.1.1.4        Et0/0
Pas:13756     1164    11        9      126  6181747   756877    21232     4079
Act:    0        0     0        0        0        N        N        N
00:03:43  10.1.1.1        Et0/0
Pas:14350     1217    11        6      153  6839987   794944    22919     4434
Act:    0        0     0        0        0        N        N        N
00:04:39  10.1.1.3        Et0/0
Pas:13431     1129    11       10      122  6603568   730905    21491     4160
Act:    0        0     0        0        0        N        N        N
00:05:42  10.1.1.2        Et0/0
Pas:14200     1186    11        9      125  4566305   765525    18718     3461
Act:    0        0     0        0        0        N        N        N
00:06:39  10.1.1.3        Et0/0
Pas:14108     1207    11        5      150  3171450   795278    16671     2903
Act:    0        0     0        0        0        N        N        N
00:07:39  10.1.1.4        Et0/0
Pas:11554      983    11       15      133  8386375   642790    23238     4793
Act:    0        0     0        0        0        N        N        N
```

Step 4 **show pfr master exits**

Use this command to display information about the exits used for PfR traffic classes, including the IP address, nickname of the PfR managed external interface, the exit policy, interface of the border router, and exit performance data. The example below shows RSVP pool information.

Example:

```
Router# show pfr master exits
```

```
PfR Master Controller Exits:

General Info:
=============
  E - External
  I - Internal
  N/A - Not Applicable
                                                                                                  Up/
   ID Name         Border          Interface   ifIdx IP Address      Mask Policy      Type Down
  --- ------------ --------------- ----------- ----- --------------- ---- ----------- ---- ----
    6 external1    10.1.0.23       Fa1/0           9 10.185.252.23     27 Util          E  UP
    5 external2    10.1.0.23       Fa1/1          10 172.16.43.23      27 Util          E  UP
    4              10.1.0.24       Tu24           33 10.20.20.24       24 Util          E  UP

Global Exit Policy:
===================
    Range Egress:      In Policy - No difference between exits - Policy 10%
   Range Ingress:      In Policy - No difference between entrances - Policy 0%
     Util Egress:      In Policy
    Util Ingress:      In Policy
            Cost:      In Policy

Exits Performance:
==================
                  Egress                                          Ingress
    ----------------------------------------------- -----------------------------------
 ID Capacity  MaxUtil    Usage   %      RSVP POOL    OOP Capacity  MaxUtil    Usage   %  OOP
 --- -------- -------- -------- --- -------------- ----- -------- -------- -------- --- -----
  6   100000    90000       66   0           9000    N/A   100000   100000       40   0  N/A
  5   100000    90000       34   0           8452    N/A   100000   100000       26   0  N/A
  4   100000    90000      128   0           5669    N/A   100000   100000      104   0  N/A

TC and BW Distribution:
=======================
                     # of TCs                  BW (kbps)            Probe   Active
     Name/ID   Current Controlled InPolicy    Controlled       Total  Failed  Unreach
                                                                     (count) (fpm)
        ----   ----------------------------   ----------------------  ------  --------
           6       0          0        0           0           66       0          0
           5     548        548      548           0           34       0          0
           4    3202       3202     3202           0          128       0          0

Exit Related TC Stats:
======================
                              Priority
                            highest       nth
                           ------------------
  Number of TCs with range:        0         0
   Number of TCs with util:        0         0
   Number of TCs with cost:        0         0

       Total number of TCs:     3800
```

Step 5 **show pfr master active-probes** [**assignment** | **running**] [**forced** *policy-sequence-number* | **longest-match**]

The following example shows the status of all created or in-progress probes.

Example:

```
Router# show pfr master active-probes running

PfR Master Controller running probes:

Border         Interface   Type     Target         TPort Codec    Freq Forced Pkts DSCP
```

```
                                                                        (Pol
                                                                         Seq)
-------------- ----------- -------- -------------- ----- -------- ---- ------ ---- ----
10.100.100.200 Ethernet1/0 tcp-conn 10.100.200.100 65535 g711alaw 10   20     100  ef
10.2.2.3       Ethernet1/0 tcp-conn 10.1.5.1       23    N        56   10     1    defa
10.1.1.1       Ethernet1/0 tcp-conn 10.1.5.1       23    N        30   N      1    defa
10.1.1.2       Ethernet1/0 tcp-conn 10.1.2.1       23    N        56   N      1    defa
10.2.2.3       Ethernet1/0 tcp-conn 10.1.2.1       23    N        56   N      1    defa
10.1.1.1       Ethernet1/0 tcp-conn 10.1.2.1       23    N        56   N      1    defa
```

Step 6 **show pfr master border** [*ip-address*] [**detail** | **report** | **statistics** | **topology**]

Entered on the controller, this command displays statistics about all the border routers.

Example:

```
Router# show pfr master border statistics

PFR Master Controller Border
 MC Version: 2.3
 Keepalive : 5 second
 Keepalive : DISABLED

                                                 Last
Border           Status Up/Down UpTime   AuthFail  Receive  Version
---------------- ------ ------- -------- --------  -------- -------
10.200.200.200   ACTIVE UP      03:12:12        0  00:00:04 2.2
10.1.1.2         ACTIVE UP      03:10:53        0  00:00:10 2.2
10.1.1.1         ACTIVE UP      03:12:12        0  00:01:00 2.2

Border Connection Statistics
============================

                          Bytes         Bytes    Msg    Msg    Sec Buf
Border                     Sent         Recvd   Sent  Recvd Bytes Used
---------------- -------------- ------------- ------ ------ ----------
10.200.200.200           345899        373749      5     10          0
10.1.1.2                 345899        373749      5     10          0
10.1.1.1                 345899        373749      5     10          0

                 Socket Invalid   Context
Border           Closed Message Not Found
---------------- ------ ------- ---------
10.200.200.200        5      10       100
10.1.1.2              5      10       100
10.1.1.1              5      10       100
```

Step 7 **show pfr master statistics** [**active-probe** | **border** | **cc** | **exit** | **netflow** | **prefix** | **process** | **system** | **timers**]

This command displays statistics from the controller. Use the keywords to filter the display information. In the example below, the **system** keyword displays PfR system statistics.

Example:

```
Router# show pfr master statistics system

   Active Timers: 14
    Total Traffic Classes = 65, Prefixes = 65, Appls =0
   TC state:
    DEFAULT = 0, HOLDDOWN = 11, INPOLICY = 54, OOP = 0, CHOOSE = 0,
    Inside = 1, Probe all = 0, Non-op = 0, Denied = 0
    Controlled 60, Uncontrolled 5, Alloced 65, Freed 0, No memory 0
   Errors:
    Invalid state = 0, Ctrl timeout = 0, Ctrl rej = 0, No ctx = 7616,
```

```
 Martians = 0
 Total Policies = 0
 Total Active Probe Targets = 325
 Total Active Probes Running = 0
Cumulative Route Changes:
 Total  : 3246
 Delay  : 0
 Loss   : 0
 Jitter : 0
 MOS    : 0
 Range  : 0
 Cost   : 0
 Util   : 0
Cumulative Out-of-Policy Events:
 Total  : 0
 Delay  : 0
 Loss   : 0
 Jitter : 0
 MOS    : 0
 Range  : 0
 Cost   : 0
 Util   :
```

Configuration Examples for PfR RSVP Control

Example Defining Traffic Classes Using RSVP Flows

The following example, configured on the primary controller, defines a learn list that will contain traffic classes that are automatically learned based on RSVP flows and filtered by a prefix list. In this example, the goal is to optimize all video traffic using the policy named POLICY_RSVP_VIDEO. The RSVP_VIDEO traffic class is defined as any prefix that matches 10.100.0.0/16 or 10.200.0.0/16 and is learned from RSVP flows.

This example configures prefix learning based on RSVP traffic flows.

```
ip prefix-list RSVP_VIDEO permit seq 10 10.100.0.0/16
ip prefix-list RSVP_VIDEO permit seq 20 10.200.0.0/16
pfr master
 policy-rules POLICY_RSVP_VIDEO
 rsvp signaling-retries 1
 rsvp post-dial-delay 100
 learn
 list seq 10 refname LEARN_RSVP_VIDEO
 traffic-class prefix-list RSVP_VIDEO
 rsvp
 exit
 exit
pfr-map POLICY_RSVP_VIDEO 10
 match learn list LEARN_RSVP_VIDEO
 set mode route control
 set resolve equivalent-path-round-robin
 end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Command List, All Releases
Cisco PfR commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	Cisco IOS Performance Routing Command Reference
Basic PfR configuration	"Configuring Basic Performance Routing" module
NetFlow and NetFlow data export	*Configuring NetFlow and NetFlow Data Export*
PfR home page with links to PfR-related content on our DocWiki collaborative environment	PfR:Home

RFCs

RFC	Title
RFC 3954	*Cisco Systems NetFlow Services Export Version 9*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

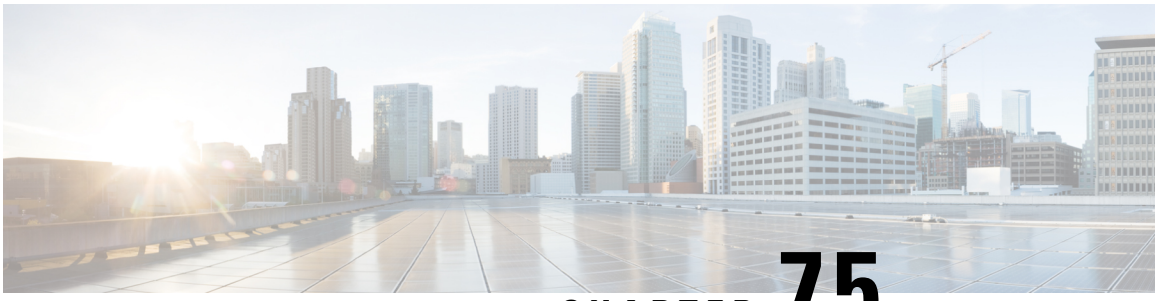
Feature Information for PfR RSVP Control

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 84: Feature Information for PfR RSVP Control

Feature Name	Releases	Feature Information
PfR RSVP Control	Cisco IOS XE Release 3.4S	The PfR RSVP Control feature provides support for optimizing RSVP flows using application-aware PfR techniques. The following commands were introduced or modified by this feature: **debug pfr border rsvp**, **debug pfr master rsvp**, **rsvp (PfR)**, **rsvp post-dial-delay**, **rsvp signaling-retries**, **resolve (PfR)**, **set resolve (PfR)**, **show pfr border rsvp**, **show pfr border routes**, **show pfr master active-probes**, **show pfr master border**, **show pfr master exits**, **show pfr master policy**, **show pfr master statistics**, **show pfr master traffic-class**, and **show pfr master traffic-class performance**.



CHAPTER 75

RSVP over UDP

The Resource Reservation Protocol (RSVP) over UDP feature provides the capability for routers to enable neighbor routers to process and send RSVP control packets over UDP. With the implementation of the RSVP over UDP feature, the RSVP protocol stack is enhanced to support processing of RSVP control messages over UDP and raw IP.

Prerequisites for RSVP Over UDP

- You must enable RSVP before you enable the RSVP over UDP feature.
- The RSVP stack running on the client host must support sending and receiving the RSVP control messages with the first hop routers they are connected to.

Information About RSVP over UDP

RSVP over UDP

The RSVP over UDP feature addresses the following scenarios:

- A router intends to communicate to the first hop router over UDP but not raw IP.
- A firewall that is located in between two routers drops raw IP packets due to security concerns, but allows UDP packets.

How to Configure RSVP over UDP

Enabling RSVP

This task starts RSVP and sets the bandwidth and single-flow limits. By default, RSVP is disabled so that it is backward compatible with systems that do not implement RSVP. To enable RSVP for IP on an interface, perform the following task.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **ip rsvp bandwidth** [*interface-bandwidth* [**percent** *percent-bandwidth* | [*single-flow-bandwidth*] [**sub-pool** *bandwidth*]]]
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Device(config)# interface fastethernet 0/1`	Configures the specified interface and enters interface configuration mode.
Step 4	**ip rsvp bandwidth** [*interface-bandwidth* [**percent** *percent-bandwidth*	[*single-flow-bandwidth*] [**sub-pool** *bandwidth*]]] **Example:** `Device(config-if)# ip rsvp bandwidth 23 54`	Enables RSVP for IP on an interface.
Step 5	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring RSVP over UDP

To enable RSVP over UDP, perform the following task:

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip rsvp udp neighbor** *neighbor-IP-address* **router** [**vrf** *vrf-name*]
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**ip rsvp udp neighbor** *neighbor-IP-address* **router** [**vrf** *vrf-name*] **Example:** `Device(config)# ip rsvp udp neighbor 10.1.1.1 router vrf vrf-1`	Configures the RSVP over UDP feature for the neighbor router.
Step 4	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Configuration examples for RSVP over UDP

Example: Enabling RSVP

The following example shows how to enable RSVP for IP on an interface:

```
enable
 configure terminal
 interface  fastethernet 0/1
  ip rsvp bandwidth 23 54
```

```
 end
```

Example: Configuring RSVP over UDP

The following example shows how to configure the RSVP over UDP feature on a neighbor router:

```
enable
 configure terminal
 ip rsvp udp neighbor 10.1.1.1 router vrf vrf-1
 end
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Master Command List, All Releases
RSVP commands	*Quality of Service Solutions Command Reference*
Overview on RSVP	*Signaling Overview*

Standards and RFCs

Standard/RFC	Title
RFC 2205	*RSVP—Version 1 Function Specification*
RFC 2209	*RSVP—Version 1 Message Processing Rules*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for RSVP over UDP

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

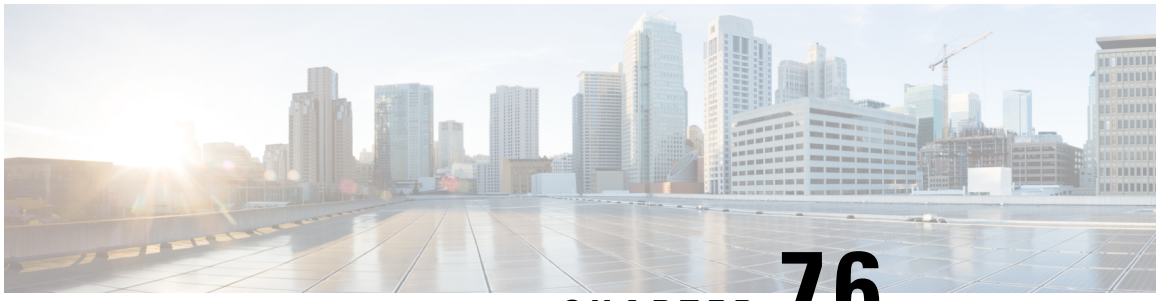
Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 85: Feature Information for RSVP over UDP

Feature Name	Releases	Feature Information
RSVP over UDP	15.2(4)M 15.4(1)S XE 3.11.0 S	The RSVP over UDP feature allows a router to enable a neighbor router to process and send RSVP control packets over UDP. The following commands were introduced or modified: **ip rsvp udp neighbor**.

PART IV

QoS Latency and Jitter



CHAPTER 76

Link Efficiency Mechanisms Overview

Cisco IOS software offers a number of link-layer efficiency mechanisms or features (listed below) designed to reduce latency and jitter for network traffic. These mechanisms work with queuing and fragmentation to improve the efficiency and predictability of the application service levels.

This chapter gives a brief introduction to these link-layer efficiency mechanisms described in the following sections:

Multilink PPP

At the top level, Multilink PPP (also known as MLP or simply Multilink) provides packet interleaving, packet fragmentation, and packet resequencing across multiple logical data links. The packet interleaving, packet fragmentation, and packet resequencing are used to accommodate the fast transmission times required for sending real-time packets (for example, voice packets) across the network links. Multilink is especially useful over slow network links (that is, a network link with a link speed less than or equal to 768 kbps).

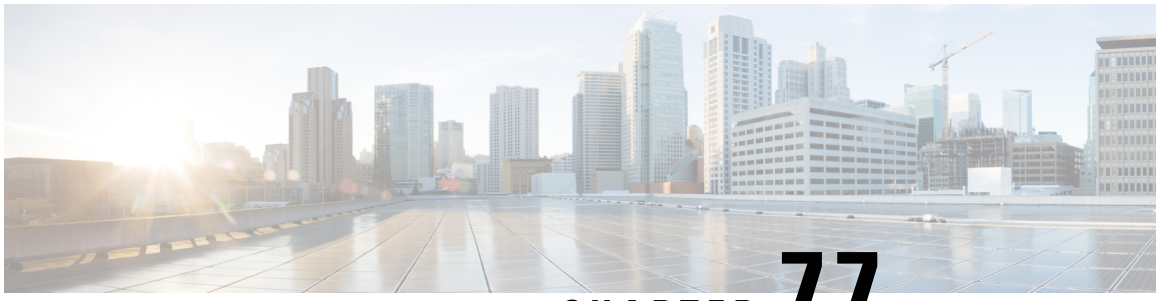
For more information about the functionality of Multilink when providing quality of service (QoS) on your network, see the "Reducing Latency and Jitter for Real-Time Traffic Using Multilink PPP" module.

Header Compression

Header compression is a mechanism that compresses the IP header in a packet before the packet is transmitted. Header compression reduces network overhead and speeds up the transmission of Real-Time Transport Protocol (RTP) and Transmission Control Protocol (TCP) packets. Header compression also reduces the amount of bandwidth consumed when the RTP or TCP packets are transmitted.

Cisco provides two basic types of header compression: RTP header compression (used for RTP packets) and TCP header compression (used for TCP packets).

For more information about header compression, see the "Header Compression" module.



CHAPTER 77

Reducing Latency and Jitter for Real-Time Traffic Using Multilink PPP

This module contains information about reducing latency and jitter for real-time traffic on your network. One Cisco mechanism for reducing latency and jitter for real-time traffic is Multilink PPP (MLP), also known as Multilink. This module contains conceptual information about Multilink and describes how Multilink PPP can be used with network peers to reduce latency and jitter for real-time traffic on your network.

Information About Multilink

Queueing Mechanisms for Multilink

You can use the following queueing mechanisms with Multilink:

- Low latency queueing (LLQ)
- Weighted fair queueing (WFQ)
- Class-based weighted fair queueing (CBWFQ)

Multilink Functionality

At the top level, Multilink provides packet interleaving, packet fragmentation, and packet resequencing across multiple logical data links. The packet interleaving, packet fragmentation, and packet resequencing is used to accommodate the fast transmission times required for sending real-time packets (for example, voice packets) across the network links. Multilink is especially useful over slow network links (that is, a network link with a link speed less than or equal to 768 kbps).

Multilink Interleaving

Multilink interleaving is based upon two other integral Multilink activities:

- The ability to fragment packets (or datagrams)

- The ability to multiplex at least two independent data streams

The term interleaving comes from the latter activity, that is, the interleaving of two (or more) independent data streams which are processed independently by the network peer.

Multilink interleaving is a mechanism that allows short, real-time (that is, time-sensitive) packets to be transmitted to a network peer within a certain amount of time (the "delay budget"). To accomplish this task, Multilink interleaving interrupts the transmission of large non-time-sensitive (sometimes referred to as "bulk") datagrams or packets in favor of transmitting the time-sensitive packet. Once the real-time packet is sent, the system resumes sending the bulk packet.

An example may help to illustrate the concept of delay budget. The network starts transmitting a large datagram to a network peer. This large datagram takes 500 milliseconds (ms) to transmit. Three milliseconds later (while the large datagram is still being transmitted), a voice packet arrives in the transmit queue. By the time the large datagram is completely transmitted (497 ms later) the voice packet (which is highly time-sensitive) is subject to unacceptable delay (that is, its delay budget is exceeded).

Multilink interleaving is particularly useful for applications where too much latency (that is, delay) is detrimental to the function of the application, such as Voice over IP (VoIP). However, it is also beneficial for other forms of "interactive" data, such as Telnet packets where the Telnet packets echo the keystrokes entered by the user at a keyboard.

Multilink Fragmentation

With Multilink fragmentation, the large datagram is fragmented ("chopped") into a number of small packet fragments, Multilink headers are added to the packet fragments, and the packet fragments are transmitted individually to a network peer.

When interleaving is enabled, the packet fragments are small enough so that the time it takes to transmit them does not exceed the time budgeted for transmitting the real-time (time-sensitive) data packet. The real-time data packets are interleaved between the fragments of the large datagram.

Each time Multilink prepares to send another data packet fragment or frame to the receiving network peer, Multilink first checks to see if a real-time (time-sensitive) packet has arrived in the transmit queue. If so, the high-priority packet is sent first before sending the next fragment from the large datagram.

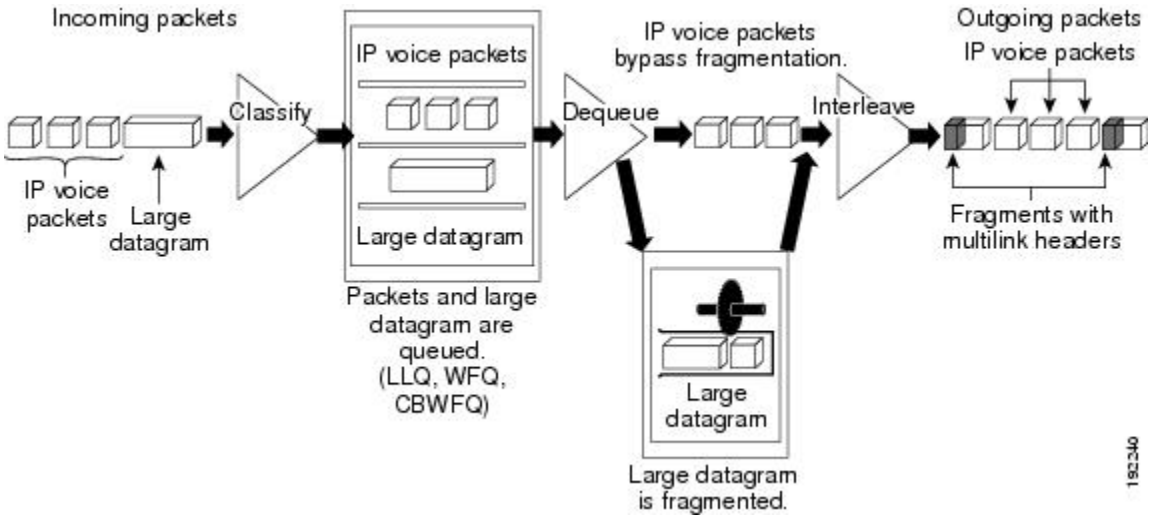
The time delay before the priority packets arrive at the receiving network link is subject to the usual serialization delays at the network link level. That is, any other data already being transmitted has to be finished before the priority packet can be sent. By segmenting long datagrams into small fragments, and checking for newly arrived priority frames between fragments, the priority frame is delayed only by the time it takes to transmit a previously queued fragment rather than a complete large datagram.

Thus, the maximum size of the fragments dictates the responsiveness for insertion of priority packets into the stream. The fragment size can be tuned by adjusting the fragment delay with the **ppp multilink fragment delay** command.

To ensure correct order of transmission and reassembly (which occurs later), multilink headers are added to the large datagram fragments after the packets are dequeued and ready to be sent.

The figure below is a simplified illustration of how Multilink fragments and interleaves packets.

Figure 108: Multilink Fragmentation and Interleaving



In the figure above, both IP voice packets and a large datagram arrive at the interface from a single network link. Your network may have multiple links. The IP voice packet and large datagram are queued according to their classification. The large datagram is fragmented (the IP voice packets are not). The IP voice packets are interleaved between the fragments of the large datagram, to which multilink headers are added.

Packets Dequeued and Transmitted

When the large datagram is dequeued, and space becomes available on a member link, Multilink takes a fragment from the original large datagram and transmits the fragments over that link. If an IP voice packet (or other real-time packet) arrives at the transmit queue before Multilink has completely sent the datagram fragment, the next time a link is available to send more packets, Multilink will dequeue and send the high-priority packet. The high-priority packet will be sent instead of another fragment from the large datagram.

Multilink Resequencing

A multilink bundle is a virtual Point-to-Point Protocol (PPP) connection or session over a network link. A multilink bundle at the transmitting end of the network sends the fragments to a multilink bundle on the receiving end of the network link.

The multilink bundle at the receiving end of the network accepts the fragments from the transmitting multilink bundle.

As fragments are received, the multilink bundle reassembles (resequences) the original large datagram from the fragments using the sequence number in the multilink header attached to the fragment by the sender. The reassembled large datagrams are then forwarded in normal fashion.

Multilink Bundles and Their Network Links

As mentioned earlier, a multilink bundle is a virtual PPP connection over a network link. The transmitting multilink bundle transmits the packet over a network link to a receiving multilink bundle, where the multilink bundle reassembles the fragments using the sequence number in the multilink header of the fragment.

The individual member links in a multilink bundle are standard serial PPP connections. Most forms of PPP connections may be used as member links in a bundle, including PPP over ATM, PPP over Frame Relay, and PPP over dial interfaces. However, there may be certain limitations and issues associated with using PPP

sessions over certain media types, particularly those for "tunneling" protocols such as PPP over ATM, PPP over Frame Relay, and PPP over Ethernet.

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	Cisco IOS Quality of Service Solutions Command Reference
LLQ, WFQ, CBWFQ, and other queueing mechanisms	"Applying QoS Features Using the MQC" module
Multilink PPP over serial interface links	"Using Multilink PPP over Serial Interface Links" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

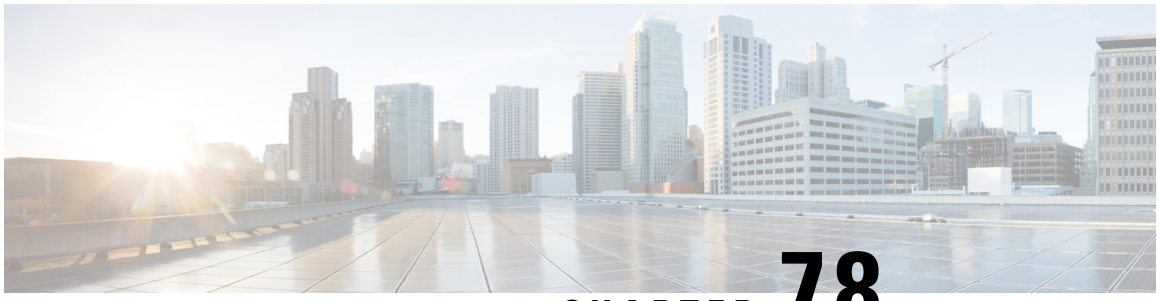
MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS XE Software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1990	*The PPP Multilink Protocol (MP)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html



CHAPTER 78

Using Multilink PPP over Serial Interface Links

This module tells you how to use Multilink PPP over serial interface links.

Note As of Cisco IOS XE Software Release 2.1, Multilink PPP over serial interface links is the only Multilink PPP type supported. Multiclass MLPPP is not supported.

Prerequisites for Using Multilink PPP over Serial Interface Links

Be familiar with the concepts in the "Reducing Latency and Jitter for Real-Time Traffic Using Multilink PPP" module.

Enable a queueing mechanism such as low latency queueing (LLQ), weighted fair queueing (WFQ), class-based WFQ (CBWFQ) and Weighted Random Early Detection (WRED), as applicable, before configuring multilink.

Restrictions for Using Multilink PPP over Serial Interface Links

If a multilink bundle has one link or packet order is not important for interleaved packets, use Link Fragmentation and Interleaving (LFI) without multiclass. Use LFI with multiclass if a multilink bundle has multiple links.

Only Voice over IP (VoIP) is supported.

As of Cisco IOS XE Release 2.1, Multilink PPP over serial interface links is the only Multilink PPP type supported. Multiclass MLPPP is not supported.

Information About Using Multilink PPP over Serial Interface Links

MQC and Multilink PPP over Serial Interface Links

Before using Multilink PPP over serial interface links, a traffic policy (policy map) must be created. Policy maps are created using the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC).

How to Configure Multilink PPP over Serial Interface Links

Configuring Multilink PPP over Serial Interface Links on a Multilink Group Interface

Before you begin

Before proceeding with this task, you must create a policy map. The policy map contains the configuration parameters used to apply the specific quality of service feature to the network traffic. To create a policy map, use the MQC.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface multilink** *multilink-bundle-number*
4. **ip address** *ip-address mask* [**secondary**]
5. **service-policy output** *policy-map-name*
6. **service-policy input** *policy-map-name*
7. **ppp multilink fragment delay** *milliseconds* [*microseconds*]
8. **ppp multilink interleave**
9. **ppp multilink multiclass**
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface multilink** *multilink-bundle-number* **Example:** `Router(config)# interface multilink 1`	Creates a multilink bundle and enters interface configuration mode. • Enter the multilink bundle number.
Step 4	**ip address** *ip-address mask* [**secondary**] **Example:** `Router(config-if)# ip address 10.10.100.1 255.255.255.0`	Sets a primary IP address for an interface. This command can also set the optional secondary IP address for an interface. • Enter the primary IP address (and, optionally, the secondary IP address).
Step 5	**service-policy output** *policy-map-name* **Example:** `Router(config-if)# service-policy output policy1`	Attaches the previously created QoS traffic policy (policy map). The policy map evaluates and applies QoS features for traffic *leaving* the interface. • Enter the policy map name.
Step 6	**service-policy input** *policy-map-name* **Example:** `Router(config-if)# service-policy input policy1`	Attaches the previously created QoS traffic policy (policy map). The policy map evaluates and applies QoS features for traffic *entering*the interface. • Enter the policy map name.
Step 7	**ppp multilink fragment delay** *milliseconds* [*microseconds*] **Example:** `Router(config-if)# ppp multilink fragment delay 20`	Specifies a maximum size in units of time for packet fragments on a Multilink PPP (MLP) bundle. • Enter the maximum amount of time, in milliseconds.
Step 8	**ppp multilink interleave** **Example:** `Router(config-if)# ppp multilink interleave`	Enables interleaving of packets among the fragments of larger packets on a multilink bundle.
Step 9	**ppp multilink multiclass** **Example:** `Router(config-if)# ppp multilink multiclass`	(Optional) Enables Multiclass Multilink PPP (MCMP) on an interface. **Note** Use this command only if there are multiple links in the multilink bundle.
Step 10	**end** **Example:**	(Optional) Exits interface configuration mode.

	Command or Action	Purpose
	`Router(config-if)# end`	

Associating the Serial Interface with the Multilink Group

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface serial** *slot* / *port* : *timeslot*
4. **no fair-queue**
5. **encapsulation ppp**
6. **ppp multilink**
7. **ppp multilink group** *group-number*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface serial** *slot* / *port* : *timeslot* **Example:** `Router# interface serial 4/1:23` **Example:**	Specifies a serial interface created on a channelized E1 or channelized T1 controller (for ISDN PRI, channel-associated signaling, or robbed-bit signaling), and enters interface configuration mode. • Enter the slot number and port number where the channelized E1 or T1 controller is located.
Step 4	**no fair-queue** **Example:** `Router(config-if)# no fair-queue`	Disables WFQ (or DWFQ for VIP-enabled routers).
Step 5	**encapsulation ppp** **Example:** `Router(config-if)# encapsulation ppp`	Sets the serial interface encapsulation method used by the interface.

	Command or Action	Purpose
Step 6	**ppp multilink** **Example:** `Router(config-if)# ppp multilink`	Enables Multilink on an interface.
Step 7	**ppp multilink group** *group-number* **Example:** `Router(config-if)# ppp multilink group 1`	Restricts a physical link to joining only a designated multilink group interface. • Enter the multilink group number.
Step 8	**end** **Example:** `Router(config-if)# end`	(Optional) Exits interface configuration mode.

Verifying the Multilink PPP over Serial Interface Link Configuration

SUMMARY STEPS

1. **enable**
2. **show interfaces** [*type number*] [*first*] [*last*] [**accounting**]
3. **show ppp multilink** [**active** | **inactive** | **interface** *bundle-interface* | [**username** *name*] [**endpoint** *endpoint*]]
4. **show policy-map interface** *interface-name* [**vc** [*vpi/*] *vci*] [**dlci** *dlci*] [**input** | **output**]
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show interfaces** [*type number*] [*first*] [*last*] [**accounting**] **Example:** `Router# show interfaces`	(Optional) Displays statistics for all interfaces configured on the router or access server.
Step 3	**show ppp multilink** [**active**	**inactive**	**interface** *bundle-interface*	[**username** *name*] [**endpoint** *endpoint*]] **Example:** `Router# show ppp multilink`	(Optional) Displays bundle information for mutlilink bundles.

	Command or Action	Purpose
Step 4	**show policy-map interface** *interface-name* [**vc** [*vpi/*] *vci*] [**dlci** *dlci*] [**input** | **output**] **Example:** `Router# show policy-map interface serial0/0/0`	(Optional) Displays the packet statistics of all classes that are configured for all service policies either on the specified interface or subinterface or on a specific permanent virtual circuit (PVC) on the interface.
Step 5	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Configuration Examples for Using Multilink PPP over Serial Interface Links

Configuring Multilink PPP over Serial Interface Links on a Multilink Group Interface Example

The following is an example of configuring Multilink PPP over serial interface links on a multilink group interface:

```
Router> enable

Router# configure terminal

Router(config)# interface multilink 1

Router(config-if)# ip address 10.10.100.1 255.255.255.0


Router(config-if)# service-policy output policy1

Router(config-if)# service-policy input policy1

Router(config-if)# ppp multilink fragment delay 20

Router(config-if)# ppp multilink interleave

Router(config-if)# ppp multilink multiclass

Router(config-if)# end
```

Associating the Serial Interface with the Multilink Group Example

The following is an example of associating the serial interface serial4/1 with the multilink group:

```
Router> enable

Router# configure terminal

Router(config)# interface serial 4/1:23

Router(config-if)# no fair-queue

Router(config-if)# encapsulation ppp

Router(config-if)# ppp multilink

Router(config-if)# ppp multilink group 1

Router(config-if)# end
```

Example Verifying the Multilink PPP over Serial Interface Link Configuration

You can verify the Multilink PPP over serial interface links configuration by using one or more of the following **show** commands:

- **show interfaces**
- **show ppp multilink**
- **show policy-map interface**

The following section provides sample output of the **showpppmultilink** command only. For sample output of the other commands, see the *Cisco IOS Quality of Service Solutions Command Reference*.

show ppp multilink Command Output Example

The following is an example of the **showpppmultilink** command output. In this example, one multilink bundle called bundle-1 is on the system. This bundle has two member links: one active link and one inactive link.

```
Router# show ppp multilink
Multilink2, bundle name is bundle-1
  Endpoint discriminator is bundle-1
  Bundle up for 00:00:09, 1/255 load
  Receive buffer limit 12000 bytes, frag timeout 1500 ms
    0/0 fragments/bytes in reassembly list
    0 lost fragments, 0 reordered
    0/0 discarded fragments/bytes, 0 lost received
    0x0 received sequence, 0x3 sent sequence
  Member links:1 active, 1 inactive (max not set, min not set)
```

```
Se3/2, since 00:00:10, 240 weight, 232 frag size
Se3/3 (inactive)
```

Additional References

The following sections provide references related to using Multilink PPP over ATM links.

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
LLQ, WFQ, CBWFQ, PQ, CQ, FIFO, and other queueing mechanisms	"Configuring Weighted Fair Queueing" module
MQC	"Applying QoS Features Using the MQC" module
Multilink PPP configurations	"Configuring Media-Independent PPP and Multilink PPP" module
Virtual template interfaces	"Configuring Virtual Template Interfaces" module
Multilink PPP overview module	"Reducing Latency and Jitter for Real-Time Traffic Using Multilink PPP" module
Multilink PPP over Frame Relay	"Using Multilink PPP over Frame Relay" module
Multilink PPP over dialer interface links	"Using Multilink PPP over Dialer Interface Links" module
Multilink PPP over serial interface links	"Using Multilink PPP over Serial Interface Links" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1990	*The PPP Multilink Protocol (MP)*
RFC 2686	*Multiclass Extension to Multilink PPP (MCML)*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Using Multilink PPP over Serial Interface Links

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

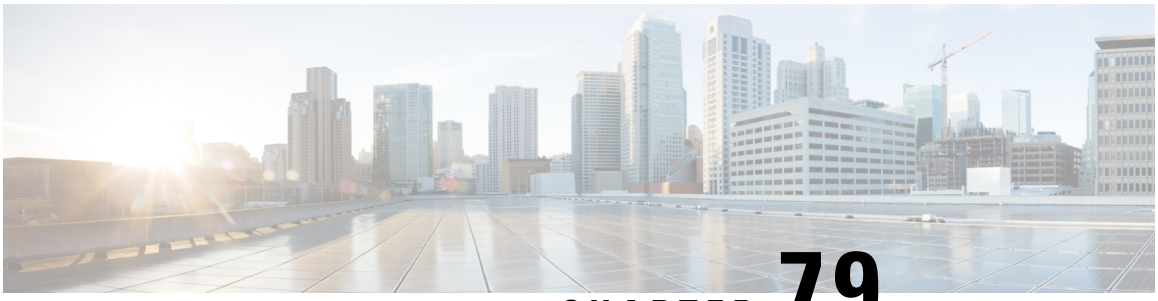
Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 86: Feature Information for Using Multilink PPP over Serial Interface Links

Feature Name	Software Releases	Feature Configuration Information
MLPPP QoS (CBWFQ, LLQ, WRED)	Cisco IOS XE Release 2.1	The MLPPP QoS feature implements Multilink PPP (MLPPP) using a distributed hierarchical queueing framework (HQF). The MLPPP QoS feature incorporates class-based weighted fair queueing (CBWFQ), low latency queueing (LLQ), and weighted random early detection (WRED) functionality.

PART V

QoS Congestion - Avoidance/Management



CHAPTER 79

Congestion Avoidance Overview

Congestion avoidance techniques monitor network traffic loads in an effort to anticipate and avoid congestion at common network bottlenecks. Congestion avoidance is achieved through packet dropping. Among the more commonly used congestion avoidance mechanisms is Random Early Detection (RED), which is optimum for high-speed transit networks. Cisco IOS XE Software includes an implementation of RED, called Weighted RED (WRED), that combines the capabilities of the RED algorithm with the IP Precedence feature. WRED, when configured, controls when the router drops packets.

Weighted Random Early Detection

WRED helps avoid the globalization problems that can occur. Global synchronization occurs as waves of congestion crest only to be followed by troughs during which the transmission link is not fully utilized. Global synchronization of TCP hosts, for example, can occur because packets are dropped all at once. Global synchronization manifests when multiple TCP hosts reduce their transmission rates in response to packet dropping and then increase their transmission rates once again when the congestion is reduced.

About Random Early Detection

The RED mechanism was proposed by Sally Floyd and Van Jacobson in the early 1990s to address network congestion in a responsive rather than reactive manner. Underlying the RED mechanism is the premise that most traffic runs on data transport implementations that are sensitive to loss and will temporarily slow down when some of their traffic is dropped. TCP, which responds appropriately--even robustly--to traffic drop by slowing down its traffic transmission, effectively allows the traffic-drop behavior of RED to work as a congestion-avoidance signalling mechanism.

TCP constitutes the most heavily used network transport. Given the ubiquitous presence of TCP, RED offers a widespread, effective congestion-avoidance mechanism.

In considering the usefulness of RED when robust transports such as TCP are pervasive, it is important to consider also the seriously negative implications of employing RED when a significant percentage of the traffic is not robust in response to packet loss. Neither Novell NetWare nor AppleTalk is appropriately robust in response to packet loss, therefore you should not use RED for them.

How It Works

The DiffServ Compliant WRED feature enables WRED to use the DSCP value when it calculates the drop probability for a packet. The DSCP value is the first six bits of the IP type of service (ToS) byte.

This feature adds two new commands, **random-detect dscp** and **dscp**. It also adds two new arguments, *dscp-based* and *prec-based* **,** to two existing WRED-related commands--the **random-detect**(interface) command and the **random-detect-group** command.

The *dscp-based* argument enables WRED to use the DSCP value of a packet when it calculates the drop probability for the packet. The *prec-based* argument enables WRED to use the IP Precedence value of a packet when it calculates the drop probability for the packet.

These arguments are optional (you need not use any of them to use the commands) but they are also mutually exclusive. That is, if you use the *dscp-based* argument, you cannot use the *prec-based* argument with the same command.

After enabling WRED to use the DSCP value, you can then use the new **random-detect dscp** command to change the minimum and maximum packet thresholds for that DSCP value.

Three scenarios for using these arguments are provided.

Packet Drop Probability

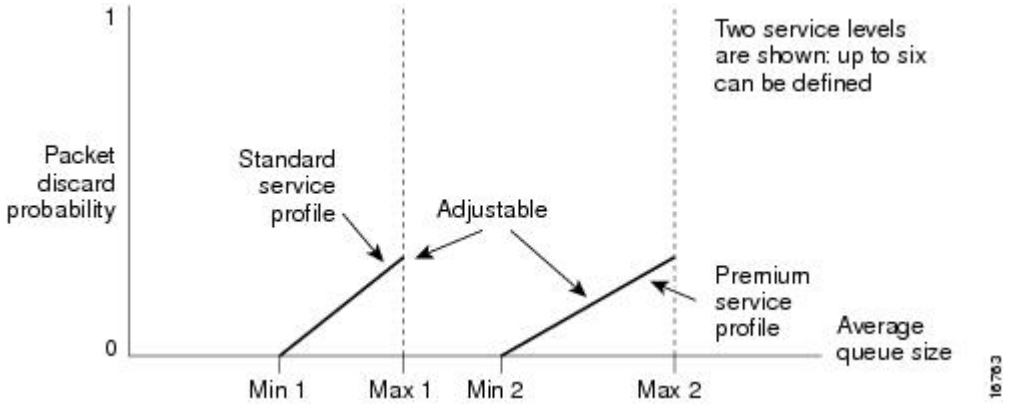
The packet drop probability is based on the minimum threshold, maximum threshold, and mark probability denominator.

When the average queue depth is above the minimum threshold, RED starts dropping packets. The rate of packet drop increases linearly as the average queue size increases until the average queue size reaches the maximum threshold.

The mark probability denominator is the fraction of packets dropped when the average queue depth is at the maximum threshold. For example, if the denominator is 512, one out of every 512 packets is dropped when the average queue is at the maximum threshold.

When the average queue size is above the maximum threshold, all packets are dropped. The figure below summarizes the packet drop probability.

Figure 109: RED Packet Drop Probability



The minimum threshold value should be set high enough to maximize the link utilization. If the minimum threshold is too low, packets may be dropped unnecessarily, and the transmission link will not be fully used.

The difference between the maximum threshold and the minimum threshold should be large enough to avoid global synchronization of TCP hosts (global synchronization of TCP hosts can occur as multiple TCP hosts reduce their transmission rates). If the difference between the maximum and minimum thresholds is too small, many packets may be dropped at once, resulting in global synchronization.

How TCP Handles Traffic Loss

Note Both this section and How the Router Interacts with TCP, on page 991 contain detailed information that you need not read in order to use WRED or to have a general sense of the capabilities of RED. If you want to understand why problems of global synchronization occur in response to congestion and how RED addresses them, read these sections.

When the recipient of TCP traffic--called the receiver--receives a data segment, it checks the four octet (32-bit) sequence number of that segment against the number the receiver expected, which would indicate that the data segment was received in order. If the numbers match, the receiver delivers all of the data that it holds to the target application, then it updates the sequence number to reflect the next number in order, and finally it either immediately sends an acknowledgment (ACK) packet to the sender or it schedules an ACK to be sent to the sender after a short delay. The ACK notifies the sender that the receiver received all data segments up to but not including the one marked with the new sequence number.

Receivers usually try to send an ACK in response to alternating data segments they receive; they send the ACK because for many applications, if the receiver waits out a small delay, it can efficiently include its reply acknowledgment on a normal response to the sender. However, when the receiver receives a data segment out of order, it immediately responds with an ACK to direct the sender to resend the lost data segment.

When the sender receives an ACK, it makes this determination: It determines if any data is outstanding. If no data is outstanding, the sender determines that the ACK is a keepalive, meant to keep the line active, and it does nothing. If data is outstanding, the sender determines whether the ACK indicates that the receiver has received some or none of the data. If the ACK indicates receipt of some data sent, the sender determines if new credit has been granted to allow it to send more data. When the ACK indicates receipt of none of the data sent and there is outstanding data, the sender interprets the ACK to be a repeatedly sent ACK. This condition indicates that some data was received out of order, forcing the receiver to remit the first ACK, and that a second data segment was received out of order, forcing the receiver to remit the second ACK. In most cases, the receiver would receive two segments out of order because one of the data segments had been dropped.

When a TCP sender detects a dropped data segment, it resends the segment. Then it adjusts its transmission rate to half of what is was before the drop was detected. This is the TCP back-off or slow-down behavior. Although this behavior is appropriately responsive to congestion, problems can arise when multiple TCP sessions are carried on concurrently with the same router and all TCP senders slow down transmission of packets at the same time.

How the Router Interacts with TCP

Note The sections How TCP Handles Traffic Loss, on page 991 contain detailed information that you need not read in order to use WRED or to have a general sense of the capabilities of RED. If you want to understand why problems of global synchronization occur in response to congestion and how RED addresses them, read these sections.

To see how the router interacts with TCP, we will look at an example. In this example, on average, the router receives traffic from one particular TCP stream every other, every 10th, and every 100th or 200th message in the interface in MAE-EAST or FIX-WEST. A router can handle multiple concurrent TCP sessions. Because network flows are additive, there is a high probability that when traffic exceeds the Transmit Queue Limit (TQL) at all, it will vastly exceed the limit. However, there is also a high probability that the excessive traffic depth is temporary and that traffic will not stay excessively deep except at points where traffic flows merge or at edge routers.

If the router drops all traffic that exceeds the TQL, many TCP sessions will simultaneously go into slow start. Consequently, traffic temporarily slows down to the extreme and then all flows slow-start again; this activity creates a condition of global synchronization.

However, if the router drops no traffic, as is the case when queueing features such as fair queueing or priority queueing (PQ) are used, then the data is likely to be stored in main memory, drastically degrading router performance.

By directing one TCP session at a time to slow down, RED solves the problems described, allowing for full utilization of the bandwidth rather than utilization manifesting as crests and troughs of traffic.

About WRED

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.

For interfaces configured to use the Resource Reservation Protocol (RSVP) feature, WRED chooses packets from other flows to drop rather than the RSVP flows. Also, IP Precedence governs which packets are dropped--traffic that is at a lower precedence has a higher drop rate and therefore is more likely to be throttled back.

WRED differs from other congestion avoidance techniques such as queueing strategies because it attempts to anticipate and avoid congestion rather than control congestion once it occurs.

Why Use WRED

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 where 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 is also RSVP-aware, and it can provide the controlled-load QoS service of integrated service.

How It Works

By randomly dropping packets prior to periods of high congestion, WRED tells the packet source to decrease its transmission rate. If the packet source is using TCP, it will decrease its transmission rate until all the packets reach their destination, which indicates that the congestion is cleared.

WRED generally drops packets selectively based on IP precedence. Packets with a higher IP precedence are less likely to be dropped than packets with a lower precedence. Thus, the higher the priority of a packet, the higher the probability that the packet will be delivered.

WRED selectively drops packets when the output interface begins to show signs of congestion. By dropping some packets early rather than waiting until the queue is full, WRED avoids dropping large numbers of packets at once and minimizes the chances of global synchronization. Thus, WRED allows the transmission line to be used fully at all times.

In addition, WRED statistically drops more packets from large users than small. Therefore, traffic sources that generate the most traffic are more likely to be slowed down than traffic sources that generate little traffic.

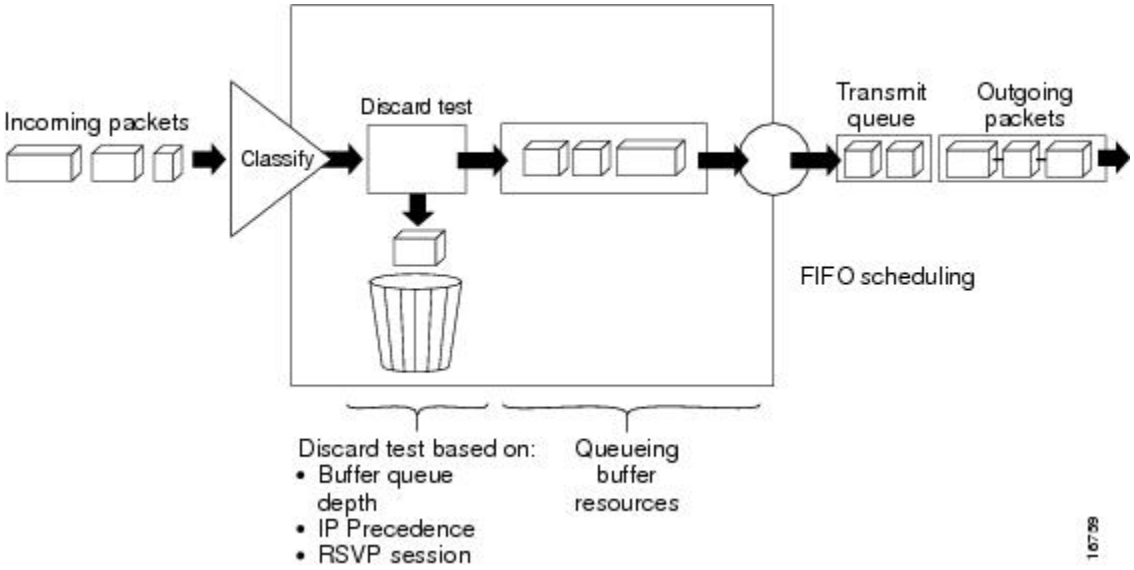
WRED helps to avoid the globalization problems. Global synchronization manifests when multiple TCP hosts reduce their transmission rates in response to packet dropping and then increase their transmission rates once again when the congestion is reduced.

WRED is only useful when the bulk of the traffic is TCP/IP traffic. With TCP, dropped packets indicate congestion, so the packet source will reduce 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.

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.

The figure below illustrates how WRED works.

Figure 110: Weighted Random Early Detection



Average Queue Size

The average queue size is based on the previous average and the current size of the queue. The formula is:

```
average = (old_average * (1-1/2^n)) + (current_queue_size * 1/2^n)
```

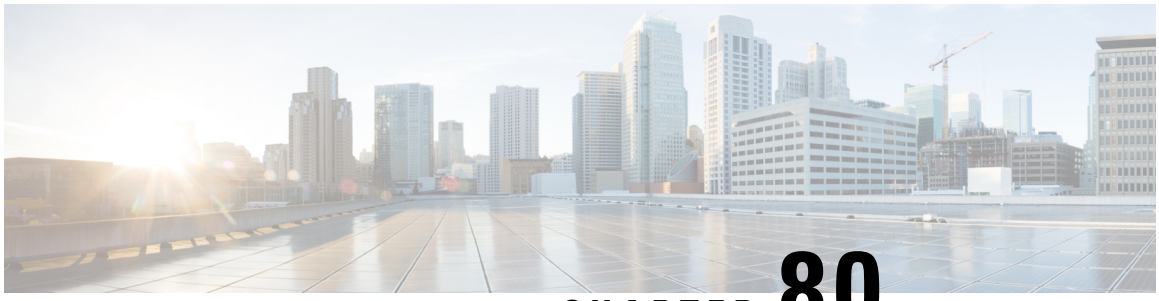
where *n* is the exponential weight factor, a user-configurable value.

For high values of *n*, the previous average queue size 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 drastic swings in size. The WRED process will be 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 will accommodate temporary bursts in traffic.

Note If the value of *n* gets too high, WRED will not react to congestion. Packets will be sent or dropped as if WRED were not in effect.

For low values of *n*, 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 *n* gets too low, WRED will overreact to temporary traffic bursts and drop traffic unnecessarily.



CHAPTER 80

IPv6 QoS: MQC WRED-Based Drop

WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of CBWFQ.

Information About IPv6 QoS: MQC WRED-Based Drop

Implementation Strategy for QoS for IPv6

IPv6 packets are forwarded by paths that are different from those for IPv4. QoS features supported for IPv6 environments include packet classification, queueing, traffic shaping, weighted random early detection (WRED), class-based packet marking, and policing of IPv6 packets. These features are available at both the process switching and Cisco Express Forwarding switching paths of IPv6.

All of the QoS features available for IPv6 environments are managed from the modular QoS command-line interface (MQC). The MQC allows you to define traffic classes, create and configure traffic policies (policy maps), and then attach those traffic policies to interfaces.

To implement QoS in networks running IPv6, follow the same steps that you would follow to implement QoS in networks running only IPv4. At a very high level, the basic steps for implementing QoS are as follows:

- Know which applications in your network need QoS.
- Understand the characteristics of the applications so that you can make decisions about which QoS features would be appropriate.
- Know your network topology so that you know how link layer header sizes are affected by changes and forwarding.
- Create classes based on the criteria you establish for your network. In particular, if the same network is also carrying IPv4 traffic along with IPv6, decide if you want to treat both of them the same way or treat them separately and specify match criteria accordingly. If you want to treat them the same, use match statements such as **match precedence**, **match dscp**. If you want to treat them separately, add match criteria such as **match protocol ip** and **match protocol ipv6** in a match-all class map.
- Create a policy to mark each class.
- Work from the edge toward the core in applying QoS features.

- Build the policy to treat the traffic.
- Apply the policy.

Congestion Avoidance for IPv6 Traffic

WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of class-based weighted fair queueing (CBWFQ). WRED supports class-based and flow-based queueing (using DSCP or precedence values).

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Command List, All Releases*
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Modular Quality of Service Command-Line Interface	"Applying QoS Features Using the MQC" module
Configuring RADIUS-based policing	*Intelligent Services Gateway Configuration Guide*
CISCO ASR 1000 Series software configuration	*Cisco ASR 1000 Series Aggregation Services Routers Software Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

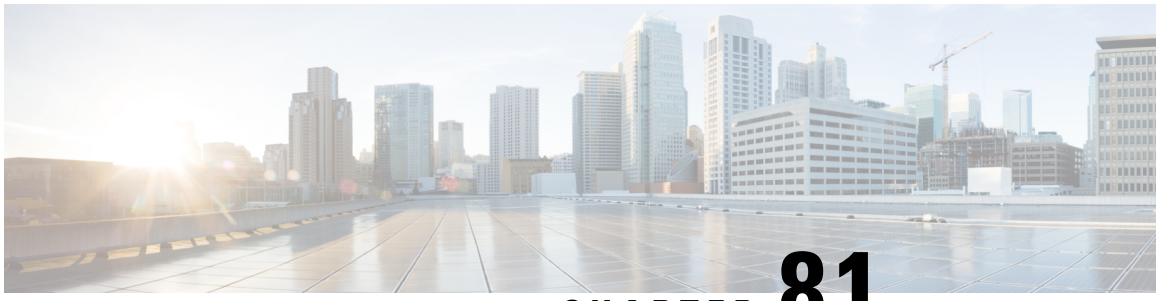
Feature Information for IPv6 QoS: MQC WRED-Based Drop

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 87: Feature Information for IPv6 QoS: MQC WRED-Based Drop

Feature Name	Releases	Feature Information
IPv6 QoS: MQC WRED-Based Drop	Cisco IOS XE Release 2.1	WRED implements the RED-based drop policy on the packets that are likely to overflow the limits of CBWFQ.



CHAPTER 81

Configuring Weighted Random Early Detection

This module describes the tasks for configuring Weighted Random Early Detection (WRED) on a router.

About Weighted Random Early Detection

Random Early Detection (RED) is a congestion avoidance mechanism that 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. WRED drops packets selectively based on IP precedence. Edge routers assign IP precedences to packets as they enter the network. (WRED is useful on any output interface where you expect to have congestion. However, WRED is usually used in the core routers of a network, rather than at the edge.) WRED uses these precedences to determine how it treats different types of traffic.

When a packet arrives, the following events occur:

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

Note WRED is useful with adaptive traffic such as TCP/IP. With TCP, dropped packets indicate congestion, so the packet source will reduce 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. WRED treats non-IP traffic as precedence 0, the lowest precedence. Therefore, non-IP traffic is more likely to be dropped than IP traffic.

When you enable WRED with the **random-detect** interface configuration command, the parameters are set to their default values. The weight factor is 9. For all precedences, the mark probability denominator is 10, and maximum threshold is based on the output buffering capacity and the transmission speed for the interface.

The default minimum threshold depends on the precedence. The minimum threshold for IP Precedence 0 corresponds to half of the maximum threshold. The values for the remaining precedences fall between half the maximum threshold and the maximum threshold at evenly spaced intervals.

Note The default WRED parameter values are based on the best available data. We recommend that you do not change the parameters from their default values unless you have determined that your applications will benefit from the changed values.

How to Configure WRED

Enabling WRED

Command	Purpose
`Router(config-if)# random-detect`	Enables WRED.

Changing WRED Parameters

Command	Purpose
`Router(config-if)# random-detect exponential-weighting-constant` *exponent*	Configures the weight factor used in calculating the average queue length.
`Router(config-if)# random-detect precedence` *precedence min-threshold max-threshold mark-prob-denominator*	Configures parameters for packets with a specific IP Precedence. The minimum threshold for IP Precedence 0 corresponds to half the maximum threshold for the interface. Repeat this command for each precedence. To configure RED, rather than WRED, use the same parameters for each precedence.

Monitoring WRED

Command	Purpose
`Router# show queue` *interface-type interface-number*	Displays the header information of the packets inside a queue.

Command	Purpose
`Router# show queueing interface interface-number [vc [[vpi/] vci]]`	Displays the WRED statistics of a specific virtual circuit (VC) on an interface.
`Router# show queueing random-detect`	Displays the queueing configuration for WRED.
`Router# show interfaces [type slot | port-adapter | port]`	Displays WRED configuration on an interface.

WRED Configuration Examples

Example WRED Configuration

The following example enables WRED with default parameter values:

```
interface Serial5/0
 description to qos1-75a
 ip address 200.200.14.250 255.255.255.252
 random-detect
```

Use the **show interfaces** command output to verify the configuration. Notice that the "Queueing strategy" report lists "random early detection (RED)."

```
Router# show interfaces serial 5/0
Serial5/0 is up, line protocol is up
  Hardware is M4T
  Description: to qos1-75a
  Internet address is 200.200.14.250/30
  MTU 1500 bytes, BW 128 Kbit, DLY 20000 usec,
     reliability 255/255, txload 1/255, rxload 237/255
  Encapsulation HDLC, crc 16, loopback not set
  Keepalive not set
  Last input 00:00:15, output 00:00:00, output hang never
  Last clearing of "show interface" counters 00:05:08
  Input queue: 0/75/0 (size/max/drops); Total output drops: 1036
  Queueing strategy: random early detection(RED)
  5 minutes input rate 0 bits/sec, 2 packets/sec
  5 minutes output rate 119000 bits/sec, 126 packets/sec
     594 packets input, 37115 bytes, 0 no buffer
     Received 5 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
     37525 packets output, 4428684 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 output buffer failures, 0 output buffers swapped out
     0 carrier transitions     DCD=up  DSR=up  DTR=up  RTS=up  CTS=up
```

Use the **show queue** command output to view the current contents of the interface queue. Notice that there is only a single queue into which packets from all IP precedences are placed after dropping has taken place. The output has been truncated to show only three of the five packets.

```
Router# show queue serial 5/0
```

```
Output queue for Serial5/0 is 5/0
Packet 1, linktype: ip, length: 118, flags: 0x288
  source: 190.1.3.4, destination: 190.1.2.2, id: 0x0001, ttl: 254,
  TOS: 128 prot: 17, source port 11111, destination port 22222
    data: 0x2B67 0x56CE 0x005E 0xE89A 0xCBA9 0x8765 0x4321
          0x0FED 0xCBA9 0x8765 0x4321 0x0FED 0xCBA9 0x8765
Packet 2, linktype: ip, length: 118, flags: 0x288
  source: 190.1.3.5, destination: 190.1.2.2, id: 0x0001, ttl: 254,
  TOS: 160 prot: 17, source port 11111, destination port 22222
    data: 0x2B67 0x56CE 0x005E 0xE89A 0xCBA9 0x8765 0x4321
          0x0FED 0xCBA9 0x8765 0x4321 0x0FED 0xCBA9 0x8765
Packet 3, linktype: ip, length: 118, flags: 0x280
  source: 190.1.3.6, destination: 190.1.2.2, id: 0x0001, ttl: 254,
  TOS: 192 prot: 17, source port 11111, destination port 22222
    data: 0x2B67 0x56CE 0x005E 0xE89A 0xCBA9 0x8765 0x4321
          0x0FED 0xCBA9 0x8765 0x4321 0x0FED 0xCBA9 0x8765
```

Use the **show queueing** command output to view the current settings for each of the precedences. Also notice that the default minimum thresholds are spaced evenly between half and the entire maximum threshold. Thresholds are specified in terms of packet count.

```
Router# show queueing
Current random-detect configuration:
     Serial5/0
       Queueing strategy:random early detection (WRED)
       Exp-weight-constant:9 (1/512)
       Mean queue depth:28

       Class   Random       Tail    Minimum    Maximum     Mark
                 drop       drop  threshold  threshold  probability
         0        330          0         20         40     1/10
         1        267          0         22         40     1/10
         2        217          0         24         40     1/10
         3        156          0         26         40     1/10
         4         61          0         28         40     1/10
         5          6          0         31         40     1/10
         6          0          0         33         40     1/10
         7          0          0         35         40     1/10
        rsvp        0          0         37         40     1/10
```

Example Parameter-Setting WRED

The following example enables WRED on the interface and specifies parameters for the different IP precedences:

```
interface Hssi0/0/0
 description 45Mbps to R1
 ip address 10.200.14.250 255.255.255.252
 random-detect
 random-detect precedence 0 32 256 100
 random-detect precedence 1 64 256 100
 random-detect precedence 2 96 256 100
 random-detect precedence 3 120 256 100
 random-detect precedence 4 140 256 100
 random-detect precedence 5 170 256 100
 random-detect precedence 6 290 256 100
 random-detect precedence 7 210 256 100
 random-detect precedence rsvp 230 256 100
```

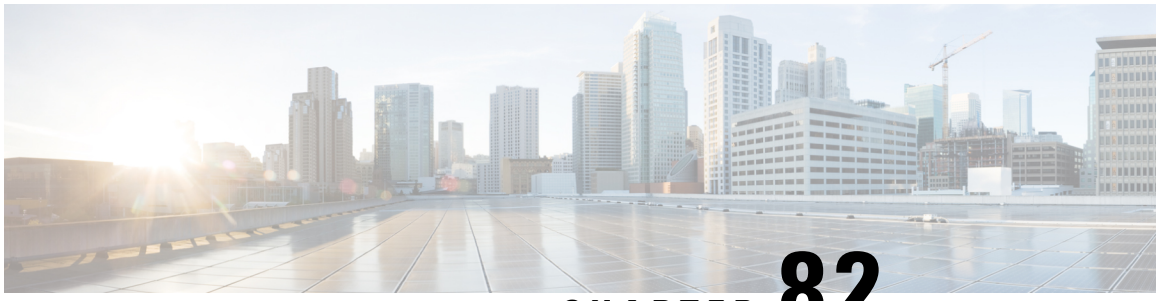
Feature Information for Configuring Weighted Random Early Detection

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 88: Feature Information for Configuring Weighted Random Early Detection

Feature Name	Releases	Feature Information
Class-Based Weighted Fair Queueing (CBWFQ) and Weighted Random Early Detection (WRED)	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers. **Note** For information about CBWFQ, see the "Configuring Weighted Fair Queueing" module.
Random Early Detection (RED)	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers.
Weighted Random Early Detection	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers.
Weighted RED (WRED)	Cisco IOS XE Release 2.1	This feature was introduced on Cisco ASR 1000 Series Routers.



CHAPTER 82

Byte-Based Weighted Random Early Detection

This module explains how to enable byte-based Weighted Random Early Detection (WRED), and set byte-based queue limits and WRED thresholds.

Restrictions for Byte-Based Weighted Random Early Detection

- WRED is only useful when the bulk of the traffic is TCP/IP traffic. With TCP, dropped packets indicate congestion, so the packet source will reduce 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.
- You cannot configure byte-based WRED on a class in which the queue-limit is configured in milliseconds or packets.

Information About Byte-Based Weighted Random Early Detection

Changes in functionality of WRED

This feature extends the functionality of WRED. In previous releases, you specified the WRED actions based on the number of packets. With the byte-based WRED, you can specify WRED actions based on the number of bytes.

Changes in Queue Limit and WRED Thresholds

In Cisco IOS XE Release 2.4, the Cisco ASR 1000 Series Aggregation Services Routers support the addition of bytes as a unit of configuration for both queue limits and WRED thresholds. Therefore, as of this release, packet-based and byte-based limits are configurable, with some restrictions.

How to Configure Byte-Based Weighted Random Early Detection

Configuring Byte-Based WRED

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** *class-map-name*
4. **match ip precedence** ip-precedence-value
5. exit
6. **policy-map** *policy-name*
7. **class** *class-name*
8. **random-detect**
9. **random-detect precedence** *precedence min-threshold* **bytes** *max-threshold* **bytes** *mark-prob-denominator*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**class-map** *class-map-name* **Example:** `Router(config)# class-map c1`	Specifies the user-defined name of the traffic class.
Step 4	**match ip precedence** ip-precedence-value **Example:** `Router(config-cmap)# match ip precedence 1`	Specifies up to eight IP Precedence values used as match criteria.

	Command or Action	Purpose
Step 5	exit **Example:** `Router(config-cmap)# exit`	Exits from class-map configuration mode.
Step 6	**policy-map** *policy-name* **Example:** `Router(config)# policy-map p1`	Specifies the name of the traffic policy to configure.
Step 7	**class** *class-name* **Example:** `Router(config-pmap)# class c1`	Specifies the name of a predefined traffic class, which was configured with the **class-map** command, used to classify traffic to the traffic policy.
Step 8	**random-detect** **Example:** `Router(config-pmap-c)# random-detect`	Enables WRED.
Step 9	**random-detect precedence** *precedence min-threshold* **bytes** *max-threshold* **bytes** *mark-prob-denominator* **Example:** **Example:** `Router(config-pmap-c)# random-detect precedence 1 2000 bytes 3000 bytes 200`	Configures the parameters for bytes with a specific IP precedence.

Configuring the Queue Depth and WRED Thresholds

Before you begin

Be sure that your configuration satisfies the following conditions when configuring the queue depth and WRED thresholds:

- When configuring byte-based mode, the queue limit must be configured prior to the WRED threshold and before the service policy is applied.
- When setting the queue depth and WRED thresholds in an enhanced QoS policies aggregation configuration, the limits are supported only for the default class at a subinterface policy map and for any classes at the main interface policy map.

Note Consider the following restrictions when you configure the queue depth and WRED thresholds:

- Do not configure the queue limit unit before you configure a queueing feature for a traffic class.
- If you do not configure a queue limit, then the default mode is packets.
- When you configure WRED thresholds, the following restrictions apply:
  - The WRED threshold must use the same unit as the queue limit. For example, if the queue limit is in packets, then the WRED thresholds also must be in packets.
  - If you do not configure a queue limit in bytes, then the default mode is packets and you must also configure the WRED threshold in packets.
  - The queue limit size must be greater than the WRED threshold.
- The unit modes for either the queue limit or WRED thresholds cannot be changed dynamically after a service policy is applied.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name*
5. *qos-queueing-feature*
6. **queue-limit** *queue-limit-size* [**bytes** | **packets**]
7. **random-detect** [**dscp-based** | **prec-based**]
8. Do one of the following:
   - **random-detect dscp** dscp-value {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*]
   -
   -
   - **random-detect precedence** precedence {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} max-probability-denominator

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Router# configure terminal`	
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map main-interface`	Specifies the name of the traffic policy that you want to configure or modify and enters policy-map configuration mode.
Step 4	**class** *class-name* **Example:** `Router(config-pmap)# class AF1`	Specifies the name of the traffic class and enters policy-map class configuration mode.
Step 5	*qos-queueing-feature* **Example:** `Router(config-pmap-c)# bandwidth remaining ratio 90`	Enters a QoS configuration command. Some of the queueing features that are currently supported are **bandwidth**, **priority**, and **shape**. **Note** Multiple QoS queueing commands can be entered at this step. However, due to dependencies between the queue limit and WRED thresholds, you should configure WRED after you configure the queue limit.
Step 6	**queue-limit** *queue-limit-size* [**bytes** | **packets**] **Example:** `Router(config-pmap-c)# queue-limit 547500 bytes`	Specifies the maximum number (from 1 to 8192000) of bytes or packets that the queue can hold for this class.
Step 7	**random-detect** [**dscp-based** | **prec-based**] **Example:** `Router(config-pmap-c)# random-detect dscp-based`	Enables WRED in either DSCP-based mode or precedence-based mode.
Step 8	Do one of the following: • **random-detect dscp** dscp-value {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*] • • • **random-detect precedence** precedence {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} max-probability-denominator **Example:** `Router(config-pmap-c)# random-detect precedence 8 750000 bytes 750000 bytes`	Configures WRED parameters for a particular DSCP value or IP precedence. **Note** Use the *min-threshold max-threshold* arguments without the **bytes** keyword to configure packet-based thresholds, when the **queue-limit** unit is also packets (the default). Alternatively, use these arguments with the **bytes** keyword when the **queue-limit** unit is configured in bytes.

Examples

Correct Configuration

Invalid Configuration

Correct Configuration

Invalid Configuration

The following examples show both correct and invalid configurations to demonstrate some of the restrictions.

The following example shows the correct usage of setting the queue limit in bytes mode after the **bandwidth remaining ratio** queueing feature has been configured for a traffic class:

```
class AF1
 bandwidth remaining ratio 90
  queue-limit 750000 bytes
```

The following example shows an invalid configuration for the queue limit in bytes mode before the **bandwidth remaining ratio** queueing feature has been configured for a traffic class:

```
class AF1
 queue-limit 750000 bytes
  bandwidth remaining ratio 90
```

The following example shows the correct usage of setting the queue limit in bytes mode after the **bandwidth remaining ratio** queueing feature has been configured for a traffic class, followed by the setting of the thresholds for WRED in compatible byte mode:

```
class AF1
 bandwidth remaining ratio 90
  queue-limit 750000 bytes
   random-detect dscp-based
    random-detect dscp 8 750000 bytes 750000 bytes
```

This example shows an invalid configuration of the WRED threshold in bytes without any queue limit configuration, which therefore defaults to a packet-based queue depth. Therefore, the WRED threshold must also be in packets:

```
class AF1
 bandwidth remaining ratio 90
 random-detect dscp-based
 random-detect dscp 8 750000 bytes 750000 bytes
```

Changing the Queue Depth and WRED Threshold Unit Modes

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **interface** *type number*
4. **no service-policy output** *policy-map-name*
5. **exit**
6. **policy-map** *policy-map-name*
7. **class** *class-name*
8. **queue-limit** *queue-limit-size* [**bytes** | **packets**]
9. Do one of the following:
   - **no random-detect dscp** *dscp-value* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*]
   -
   -
   - **no random-detect precedence** *precedence* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} *max-probability-denominator*
10. Do one of the following:
    - **random-detect dscp** *dscp-value* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*]
    -
    -
    - **random-detect precedence** *precedence* {*min-threshold max-threshold*| *min-threshold* **bytes** *max-threshold* **bytes**} *max-probability-denominator*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* **Example:** `Router(config)# policy-map main-interface`	Specifies the interface where you want to remove a service policy, and enters interface configuration mode.
Step 4	**no service-policy output** *policy-map-name* **Example:** `Router(config-if)# no service-policy output main-interface-policy`	Removes a service policy applied to the specified interface.

	Command or Action	Purpose
Step 5	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode and returns you to global configuration mode.
Step 6	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map main-interface-policy`	Specifies the name of the Traffic policy that you want to modify and enters policy-map configuration mode.
Step 7	**class** *class-name* **Example:** `Router(config-pmap)# class AF1`	Specifies the name of the traffic class and enters policy-map class configuration mode.
Step 8	**queue-limit** *queue-limit-size* [**bytes** | **packets**] **Example:** `Router(config-pmap-c)# queue-limit 5000 packets`	Specifies the maximum number (from 1 to 8192000) of bytes or packets that the queue can hold for this class.
Step 9	Do one of the following: • **no random-detect dscp** *dscp-value* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*] • • • **no random-detect precedence** *precedence* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} *max-probability-denominator* **Example:** `Router(config-pmap-c)# no random-detect dscp 8 750000 bytes 750000 bytes`	Removes the previously configured WRED parameters for a particular DSCP value or IP precedence.
Step 10	Do one of the following: • **random-detect dscp** *dscp-value* {*min-threshold max-threshold* | *min-threshold* **bytes** *max-threshold* **bytes**} [*max-probability-denominator*] • • • **random-detect precedence** *precedence* {*min-threshold max-threshold*| *min-threshold* **bytes** *max-threshold* **bytes**} *max-probability-denominator* **Example:**	Configures WRED parameters for a particular DSCP value or IP precedence. **Note** Use the *min-threshold max-threshold* arguments without the **bytes** keyword to configure packet-based thresholds, when the queue-limit unit is also packets (the default). Alternatively, use these arguments with the **bytes** keyword when the queue-limit unit is configured in bytes.

	Command or Action	Purpose
	`Router(config-pmap-c)# random-detect dscp 8 4000 4000`	

Examples

The following example shows how to change the queue depth and WRED thresholds to packet-based values once a service policy has been applied to an interface:

```
interface GigabitEthernet1/2/0
no service-policy output main-interface-policy
end
policy-map main-interface-policy
class AF1
 queue-limit 5000 packets
 no random-detect dscp 8 750000 bytes 750000 bytes
 random-detect dscp 8 4000 4000
```

Verifying the Configuration for Byte-Based WRED

SUMMARY STEPS

1. **show policy-map**
2. The **show policy-map interface** command shows output for an interface that is configured for byte-based WRED.

DETAILED STEPS

Step 1 **show policy-map**

The **show policy-map** command shows the output for a service policy called pol1 that is configured for byte-based WRED.

Example:

```
Router# show policy-map
Policy Map pol1
  Class class c1
Bandwidth 10 (%)
exponential weight 9
        class   min-threshold(bytes)   max-threshold(bytes)  mark-probability
        ----------------------------------------------------------------------

        0       -                     -                     1/10
        1       20000                 30000                 1/10
        2       -                     -                     1/10
        3       -                     -                     1/10
        4       -                     -                     1/10
        5       -                     -                     1/10
        6       -                     -                     1/10
        7       -                     -                     1/10
        rsvp    -                     -                     1/10
```

Step 2 The **show policy-map interface** command shows output for an interface that is configured for byte-based WRED.

Example:

```
Router# show policy-map interface
serial3/1
Service-policy output: pol
Class-map: silver (match-all)
366 packets, 87840 bytes
30 second offered rate 15000 bps, drop rate 300 bps
Match: ip precedence 1
Queueing
Output Queue: Conversation 266
Bandwidth 10 (%)
(pkts matched/bytes matched) 363/87120
depth/total drops/no-buffer drops) 147/38/0
exponential weight: 9
mean queue depth: 25920
class     Transmitted       Random drop      Tail drop     Minimum Maximum Mark
          pkts/bytes        pkts/bytes       pkts/bytes    thresh  thresh  prob
                                                           (bytes)  (bytes)
0             0/0               0/0              0/0       20000    40000  1/10
1           328/78720         38/9120            0/0       22000    40000  1/10
2             0/0               0/0              0/0       24000    40000  1/10
3             0/0               0/0              0/0       26000    40000  1/10
4             0/0               0/0              0/0       28000    40000  1/10
```

Configuration Examples for Byte-Based Weighted Random Early Detection

Example Configuring Byte-Based WRED

The following example shows a service policy called wred-policy that sets up byte-based WRED for a class called prec2 and for the default class. The policy is then applied to Fast Ethernet interface 0/0/1.

```
policy wred-policy
 class prec2
   bandwidth 1000
   random-detect
   random-detect precedence 2 100 bytes 200 bytes 10
class class-default
   random-detect
   random-detect precedence 4 150 bytes 300 bytes 15
   random-detect precedence 6 200 bytes 400 bytes 5
interface fastethernet0/0/1
  service-policy output wred-policy
```

The following example shows the byte-based WRED results for the service policy attached to Ethernet interface 0/0/1.

```
Router# show policy-map interface
 Ethernet0/0/1
Service-policy output: wred-policy (1177)
```

```
Class-map: prec2 (match-all) (1178/10)
0 packets, 0 bytes
5 minute offered rate 0 bps, drop rate 0 bps
Match: ip precedence 2  (1179)
Queueing
queue limit 62500 bytes
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts queued/bytes queued) 0/0
bandwidth 1000 (kbps)
Exp-weight-constant: 9 (1/512)
Mean queue depth: 0 bytes
class     Transmitted       Random drop      Tail drop Minimum        Maximum     Mark
          pkts/bytes        pkts/bytes       pkts/bytes thresh        thresh     prob
                                                        bytes        bytes
0             0/0               0/0              0/0     15625        31250     1/10
1             0/0               0/0              0/0     17578        31250     1/10
2             0/0               0/0              0/0       100          200     1/10
3             0/0               0/0              0/0     21484        31250     1/10
4             0/0               0/0              0/0     23437        31250     1/10
5             0/0               0/0              0/0     25390        31250     1/10
6             0/0               0/0              0/0     27343        31250     1/10
7             0/0               0/0              0/0     29296        31250     1/10
Class-map: class-default (match-any) (1182/0)
0 packets, 0 bytes
5 minute offered rate 0 bps, drop rate 0 bps
Match: any  (1183)
0 packets, 0 bytes
5 minute rate 0 bps
queue limit 562500 bytes
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts queued/bytes queued) 0/0
Exp-weight-constant: 9 (1/512)
Mean queue depth: 0 bytes
class     Transmitted       Random drop      Tail drop Minimum        Maximum     Mark
          pkts/bytes        pkts/bytes       pkts/bytes thresh        thresh     prob
                                                        bytes        bytes
0             0/0               0/0              0/0    140625       281250     1/10
1             0/0               0/0              0/0    158203       281250     1/10
2             0/0               0/0              0/0    175781       281250     1/10
3             0/0               0/0              0/0    193359       281250     1/10
4             0/0               0/0              0/0       150          300     1/15
5             0/0               0/0              0/0    228515       281250     1/10
6             0/0               0/0              0/0       200          400     1/5
7             0/0               0/0              0/0    263671       281250     1/10
```

Additional References

Related Documents

Related Topic	Document Title
QoS Commands	*Cisco IOS Quality of Service Solutions Command Reference*
Modular QoS CLI	Modular Quality of Service Command-Line Interface module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

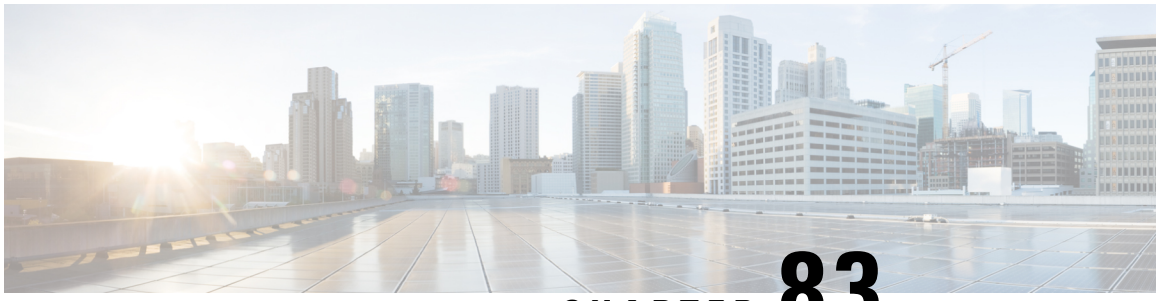
Feature Information for Byte-Based Weighted Random Early Detection

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 89: Feature Information for Byte-Based Weighted Random Early Detection

Feature Name	Releases	Feature Information
Byte-Based Weighted Random Early Detection	Cisco IOS XE Release 2.4	The Byte-Based Weighted Random Early Detection feature extends the functionality of WRED. In previous releases, you specified the WRED actions based on the number of packets. With the byte-based WRED, you can specify WRED actions based on the number of bytes. This feature was introduced on Cisco ASR 1000 Series Routers. The following commands were introduced or modified: **random-detect**, **random-detect precedence**, **show policy-map, show policy-map interface**.



CHAPTER 83

QoS Time-Based Thresholds for WRED and Queue Limit

The QoS Time-Based Thresholds for WRED and Queue Limit feature allows you to specify the Weighted Random Early Detection (WRED) minimum and maximum thresholds or the queue limit threshold in milliseconds (ms). Previously, these thresholds could only be specified in packets or bytes. Now, all three units of measure are available. Once the threshold limits are configured in a policy map, the policy map can be used on multiple interfaces, including those with different amounts of bandwidth.

Prerequisites for QoS Time-Based Thresholds for WRED and Queue Limit

Before configuring this feature, a traffic class must be configured and a policy map must exist. To create the traffic class (specifying the appropriate match criteria) and the policy map, use the modular quality of service (QoS) command-line interface (MQC).

Restrictions for QoS Time-Based Thresholds for WRED and Queue Limit

This feature allows you to specify either the WRED thresholds or the queue limit threshold in packets (the default unit of measure), bytes, or milliseconds (ms). However, these units cannot be mixed. That is, the unit of measure in the *same* class, in the *same* policy map, cannot be mixed. For example, if you specify the minimum threshold for a particular class in milliseconds, the maximum threshold for that class must also be in milliseconds.

Information About QoS Time-Based Thresholds for WRED and Queue Limit

Benefits of QoS Time-Based Thresholds for WRED and Queue Limit

Queue Limit Thresholds Specified in Additional Units of Measure

Previously, the WRED thresholds and the queue limit thresholds could only be specified in packets or bytes. With this feature, the thresholds can be specified either in packets, bytes or milliseconds. These additional units of measure provide more flexibility and allow you to fine-tune your configuration.

Policy Maps Can be Reused as Needed on Multiple Interfaces

The WRED and queue limit thresholds are specified and configured in policy maps. Once the threshold limits are configured in a policy map, the policy map can be used on multiple interfaces, including those with different amounts of bandwidth. This is especially useful when the bandwidth for a class on given interface is being specified as a percentage of the total bandwidth available.

Setting Thresholds by Using WRED

WRED is a congestion avoidance mechanism. WRED combines the capabilities of the Random Early Detection (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.

WRED differs from other congestion avoidance techniques such as queueing strategies because it attempts to anticipate and avoid congestion rather than control congestion once it occurs.

WRED is enabled by using the **random-detect**command. Then the minimum threshold, maximum threshold, and mark probability denominator can be set to determine the treatment that packets receive by using the appropriate command. For example, the **random-detect precedence** command can be used to determine the thresholds for a specific IP precedence.

Setting Thresholds by Using the queue-limit Command

The **queue-limit**command allows you to specify or modify the maximum number of packets the queue can hold (that is, the threshold) for a class policy configured in a policy map. Packets belonging to a class are subject to the guaranteed bandwidth allocation and the queue limits that characterize the traffic class. With the **queue-limit** command, the threshold is the aggregate threshold for the entire class.

After a queue has reached its configured queue limit, enqueuing of additional packets to the traffic class causes tail drop or WRED (if configured) to take effect, depending on how the policy map is configured. (Tail drop is a means of avoiding congestion that treats all traffic equally and does not differentiate between classes of service.)

Queues fill during periods of congestion. When the output queue is full and tail drop is in effect, packets are dropped until the congestion is eliminated and the queue is no longer full).

Tail drop is used for distributed class-based weighted fair queueing (DCBWFQ) traffic classes unless you explicitly configure a service policy to use WRED to drop packets as a means of avoiding congestion. Note that if you use WRED instead of tail drop for one or more traffic classes making up a service policy, you must ensure that WRED is not configured for the interface to which you attach that service policy.

random-detect Commands with the Milliseconds Keyword

This feature allows you to specify the WRED minimum and maximum thresholds in milliseconds (ms). You can specify the threshold in milliseconds by using the **ms** keyword available with the **random-detect** commands listed in the table below.

Table 90: random-detect Commands with the Milliseconds (ms) Keyword

Command	Description
random-detect clp	Configures the WRED parameters for a particular cell loss priority (CLP) value, or a particular CLP value for a class policy in a policy map.
random-detect cos	Configures the WRED parameters for a particular class of service (CoS) value, or a particular CoS value for a class policy in a policy map.
random-detect discard-class	Configures the WRED parameters for a particular discard-class, or a particular discard-class for a class policy in a policy map.
random-detect dscp	Configures the WRED parameters for a particular differentiated services code point (DSCP) value, or a particular DSCP value for a class policy in a policy map.
random-detect precedence	Configures WRED parameters for a particular IP precedence, or a particular IP precedence for a class policy in a policy map.

Mixing Threshold Units of Measure

With this feature, the thresholds can be specified in packets (the default unit of measure), bytes, or milliseconds (ms). For instance, with WRED, you can specify the minimum threshold and the maximum threshold in packets, bytes, or milliseconds. However, the units cannot be mixed. For example, if you specify the minimum threshold in milliseconds, the maximum threshold must also be specified in milliseconds.

How to Configure QoS Time-Based Thresholds for WRED and Queue Limit

Enabling WRED and Using WRED to Specify Thresholds

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **policy-map** *policy-name*
4. **class** {*class-name***class-default**}
5. To continue with the configuration, you must either specify a bandwidth or enable traffic shaping. Choose one or the other.
6. **bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*}
7. **shape** [**average** | **peak**] *mean-rate* [*burst-size*] [*excess-burst-size*]
8. **random-detect**
9. **random-detect precedence** {*precedence* | **rsvp**} *min-threshold* {**bytes**| **ms**| **packets**} *max-threshold*{**bytes** | **ms**| **packets**} [*mark-probability-denominator*]
10. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-name* **Example:** `Router(config)# policy-map policy1`	Specifies the name of the policy map to be created. Enters policy-map configuration mode. • Enter policy map name.
Step 4	**class** {*class-name***class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the class so that you can configure or modify its policy. Enters policy-map class configuration mode. • Enter the class name or specify the default class (class-default).
Step 5	To continue with the configuration, you must either specify a bandwidth or enable traffic shaping. Choose one or the other.	
Step 6	**bandwidth** {*bandwidth-kbps*	**remaining percent** *percentage*	**percent** *percentage*} **Example:** `Router(config-pmap-c)# bandwidth percent 40`	(Optional) Specifies or modifies the bandwidth allocated for a class belonging to a policy map. • Enter the bandwidth to be set or modified.
Step 7	**shape** [**average**	**peak**] *mean-rate* [*burst-size*] [*excess-burst-size*] **Example:**	(Optional) Enables either average or peak rate traffic shaping. • Specify either average or peak traffic shaping.

	Command or Action	Purpose
	`Router(config-pmap-c)# shape average 51200`	
Step 8	**random-detect** **Example:** `Router(config-pmap-c)# random-detect`	Enables WRED or distributed WRED (DWRED).
Step 9	**random-detect precedence** {*precedence* | **rsvp**} *min-threshold* {**bytes**| **ms**| **packets**} *max-threshold*{**bytes** | **ms**| **packets**} [*mark-probability-denominator*] **Example:** `Router(config-pmap-c)# random-detect precedence 2 512 ms 1020 ms`	Configures WRED and DWRED parameters for a particular IP precedence. • Specify the IP precedence or RSVP value, and thresholds, as needed. **Note** In this example, the WRED parameters were specified for traffic with a specific IP precedence value. Other values can be specified with other **random-detect**commands.
Step 10	**exit** **Example:** `Router(config-pmap-c)# exit`	(Optional) Exits policy-map class configuration mode.

Using the queue-limit Command to Specify the Thresholds

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-name*
4. **class** {*class-name* **class-default**}
5. To continue with the configuration, you must either specify a bandwidth or enable traffic shaping. Choose one or the other.
6. **bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*}
7. **shape** [**average** | **peak**] *mean-rate* [[*burst-size*] [*excess-burst-size*]]
8. **queue-limit** *number-of-packets* [**bytes** | **ms** | **packets**]
9. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router> enable`	
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**policy-map** *policy-name*
	Example:
	`Router(config)#`
	`policy-map`
	`policy1`
		Specifies the name of the policy map to be created. Enters policy-map configuration mode.
		• Enter policy map name.

Step 4	**class** {*class-name* **class-default**} **Example:** `Router(config-pmap)# class class1`	Specifies the class so that you can configure or modify its policy. Enters policy-map class configuration mode. • Enter the class name or specify the default class (class-default).
Step 5	To continue with the configuration, you must either specify a bandwidth or enable traffic shaping. Choose one or the other.	
Step 6	**bandwidth** {*bandwidth-kbps* | **remaining percent** *percentage* | **percent** *percentage*} **Example:** `Router(config-pmap-c)# bandwidth percent 40`	(Optional) Specifies or modifies the bandwidth allocated for a class belonging to a policy map. • Enter the bandwidth to be set or modified.
Step 7	**shape** [**average** | **peak**] *mean-rate* [[*burst-size*] [*excess-burst-size*]] **Example:** `Router(config-pmap-c)# shape average 51200`	(Optional) Enables either average or peak rate traffic shaping. • Specifies either average or peak traffic shaping.
Step 8	**queue-limit** *number-of-packets* [**bytes** | **ms** | **packets**] **Example:** `Router(config-pmap-c)# queue-limit 200 ms`	(Optional) Specifies or modifies the maximum number of packets the queue can hold (that is, the queue limit) for a class configured in a policy map. • Enter the queue limit. The unit of measure can be bytes, milliseconds, or packets.
Step 9	**exit** **Example:** `Router(config-pmap-c)# exit`	(Optional) Exits policy-map class configuration mode.

Attaching the Policy Map to an Interface in a QoS Time-Based Threshold for WRED Configuration

Note Depending on the needs of your network, you may need to attach the policy map to a subinterface, an ATM PVC, a Frame Relay DLCI, or other type of interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number*
4. **pvc** [*name*] *vpi* / *vci* [**ilmi** | **qsaal** | **smds**]
5. **service-policy** {**input**| **output**} *policy-map-name*
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Step 3	**interface** *type number*
	Example:
	`Router(config)#`
	`interface serial4/0`
		Configures an interface (or subinterface) type and enters interface configuration mode.
		• Enter the interface type number.

Step 4	**pvc** [*name*] *vpi* / *vci* [**ilmi**	**qsaal**	**smds**] **Example:** `Router(config-if)# pvc cisco 0/16 ilmi`	(Optional) Creates or assigns a name to an ATM PVC and specifies the encapsulation type on an ATM PVC. Enters ATM VC configuration mode. **Note** This step is required only if you are attaching the policy map to an ATM PVC. If you are not attaching the policy map to an ATM PVC, skip this step and proceed with Step 5 .
Step 5	**service-policy** {**input**| **output**} *policy-map-name* **Example:**	Specifies the name of the policy map to be attached to the input *or* output direction of the interface.

	Command or Action	Purpose

	`Router(config-if)#`
	`service-policy output policy1`
	Example:
		Note Policy maps can be configured on ingress or egress routers. They can also be attached in the input or output direction of an interface. The direction (input or output) and the router (ingress or egress) to which the policy map should be attached varies according your network configuration. When using the **service-policy** command to attach the policy map to an interface, be sure to choose the router and the interface direction that are appropriate for your network configuration.
		• Enter the policy map name.

Step 6	**exit** **Example:** `Router(config-if)# exit`	(Optional) Exits interface configuration mode.

Verifying the QoS Time-Based Thresholds for WRED and Queue Limit Configuration

SUMMARY STEPS

1. **enable**
2. **show policy-map** [*policy-map*]
3. and/or
4. **show policy-map interface** *interface-name*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show policy-map** [*policy-map*] **Example:** `Router# show policy-map policy1`	Displays all information about a class map, including the match criterion. • Enter class map name.
Step 3	and/or	

	Command or Action	Purpose

Step 4	**show policy-map interface** *interface-name*
	Example:
	`Router#`
	`show policy-map interface serial4/0`
		Displays the packet statistics of all classes that are configured for all service policies either on the specified interface or subinterface or on a specific PVC on the interface.
		• Enter the interface name.

Step 5	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Troubleshooting Tips

The commands in the "Verifying the Configuration" section allow you to verify that you achieved the intended configuration and that the feature is functioning correctly. If, after using the **show** commands listed above, you find that the configuration is not correct or the feature is not functioning as expected, perform these operations:

If the configuration is not the one you intended, complete the following steps:

1. Use the **show running-config** command and analyze the output of the command.
2. If the policy map does not appear in the output of the **show running-config** command, enable the **logging console** command.
3. Attach the policy map to the interface again.

If the packets are not being matched correctly (for example, the packet counters are not incrementing correctly), complete the following procedures:

1. Run the **show policy-map**command and analyze the output of the command.
2. Run the **show running-config** command and analyze the output of the command.
3. Use the **show policy-map interface** command and analyze the output of the command. Check the the following findings:
   a. If a policy map applies queueing, and the packets are matching the correct class, but you see unexpected results, compare the number of the packets in the queue with the number of the packets matched.

If the interface is congested, and only a small number of the packets are being matched, check the tuning of the transmission (tx) ring, and evaluate whether the queueing is happening on the tx ring. To do this, use the **show controllers** command, and look at the value of the tx count in the output of the command..

Configuration Examples for QoS Time-Based Thresholds for WRED and Queue Limit

Example Using WRED to Set Thresholds

In the following example, WRED has been configured in the policy map called "policy1". In this WRED configuration, the bandwidth has been specified as a percentage (80%), and the minimum and maximum thresholds for IP precedence 2 are set to 512 milliseconds and 1020 milliseconds, respectively.

```
Router> enable
Router# configure terminal
Router(config)#

policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth percent 80
Router(config-pmap-c)# random-detect
Router(config-pmap-c)# random-detect precedence 2 512 ms 1020 ms
Router(config-pmap-c)# exit

Router(config-pmap)# exit

Router(config)# interface s4/0
Router(config-if)#

service-policy output policy1
Router(config-if)# end
```

Example Using the queue-limit Command to Set Thresholds

In the following example, a policy map called "policy2" has been configured. The policy2 policy map contains a class called "class1." The bandwidth for this class has been specified as a percentage (80%) and the **queue-limit** command has been used to set the threshold to 200 milliseconds.

```
Router> enable
Router# configure terminal
Router(config)#

policy-map policy2
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth percent 80
Router(config-pmap-c)# queue-limit 200 ms
Router(config-pmap-c)# exit

Router(config-pmap)# exit

Router(config)# interface s4/0
Router(config-if)#

service-policy output policy1
Router(config-if)# end
```

Example Verifying the Configuration

To verify that this feature is configured correctly, use either the **show policy-map**command or the **show policy-map interface** command.

This section contains two sets of sample output from the **show policy-map interface** command and the **show policy-map** command--one set showing the output when WRED is used to configure the feature, one set showing the output when the **queue-limit**command is used to configure the feature.

Example WRED Threshold Configuration Sample Output

The following is sample output of the **show policy-map** command when WRED has been used to specify the thresholds. The words "time-based wred" indicates that the thresholds have been specified in milliseconds (ms).

```
Router# show policy-map
  Policy Map policy1
    Class class1
      bandwidth 80 (%)
       time-based wred, exponential weight 9
      class    min-threshold    max-threshold    mark-probability
      ----------------------------------------------------------
      0       -                -                1/10
      1       -                -                1/10
      2       512              1024             1/10
      3       -                -                1/10
      4       -                -                1/10
      5       -                -                1/10
      6       -                -                1/10
      7       -                -                1/10
```

The following is sample output of the **show policy-map interface** command when WRED has been used to specify the thresholds.

```
Router# show policy-map interface Ethernet2/0
 Ethernet2/0
Service-policy output: policy1 (1100)
    Class-map: class1 (match-all) (1101/1)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol ftp (1102)
      Queueing
      queue limit 16 ms/ 16000 bytes
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts queued/bytes queued) 0/0
      bandwidth 80.00% (%) (8000 kbps)
        Exp-weight-constant: 9 (1/512)
        Mean queue depth: 0 ms/ 0 bytes
        class    Transmitted  Random drop   Tail drop    Minimum       Maximum       Mark
                 pkts/bytes   pkts/bytes    pkts/bytes   thresh        thresh        prob
                                                         ms/bytes      ms/bytes
        0        0/0          0/0           0/0           4/4000        8/8000        1/10
        1        0/0          0/0           0/0           4/4500        8/8000        1/10
        2        0/0          0/0           0/0         512/512000   1024/1024000    1/10
        3        0/0          0/0           0/0           5/5500        8/8000        1/10
        4        0/0          0/0           0/0           6/6000        8/8000        1/10
        5        0/0          0/0           0/0           6/6500        8/8000        1/10
        6        0/0          0/0           0/0           7/7000        8/8000        1/10
        7        0/0          0/0           0/0           7/7500        8/8000        1/10
```

```
    Class-map: class-default (match-any) (1105/0)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: any  (1106)
        0 packets, 0 bytes
        5 minute rate 0 bps

      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts queued/bytes queued) 0/0
```

Formula for Converting the Threshold from Milliseconds to Bytes

When converting the threshold from milliseconds to bytes, the following formula is used:

milliseconds * (bandwidth configured for the class) / 8 = total number of bytes

For this example, the following numbers would be used in the formula:

512 ms * 8000 kbps / 8 = 512000 bytes

Note Class1 has a bandwidth of 8000 kbps.

Example queue-limit command Threshold Configuration Sample Output

The following is sample output of the **show policy-map** command when the **queue-limit** command has been used to specify the thresholds in milliseconds.

```
Router# show policy-map
  Policy Map policy1
    Class class1
      bandwidth 80 (%)
      queue-limit 200 ms
```

The following is sample output from the **show policy-map interface** command when the **queue-limit** command has been used to specify the thresholds.

```
Router# show policy-map interface
 Ethernet2/0
  Service-policy output: policy1 (1070)
    Class-map: class1 (match-all) (1071/1)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match: protocol ftp (1072)
      Queueing
      queue limit 200 ms/ 200000 bytes
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts queued/bytes queued) 0/0
      bandwidth 80.00% (%) (8000 kbps)
    Class-map: class-default (match-any) (1075/0)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
Match: any  (1076)
        0 packets, 0 bytes
        5 minute rate 0 bps

      queue limit 64 packets
```

```
(queue depth/total drops/no-buffer drops) 0/0/0
(pkts queued/bytes queued) 0/0
```

Formula for Converting the Threshold from Milliseconds to Bytes

When converting the threshold from milliseconds to bytes, the following formula is used:

milliseconds * (bandwidth configured for the class)/ 8 = total number of bytes

For this example, the following numbers would be used in the formula:

200 ms * 8000 kbps / 8 = 200000 bytes

Note Class1 has a bandwidth of 8000 kbps.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Quality of service (QoS) commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
Congestion avoidance mechanisms, including tail drop, RED and WRED	*Cisco IOS Quality of Service Solutions Configuration Guide*
Congestion management mechanisms, including CBWFQ, and DCBWFQ	*Cisco IOS Quality of Service Solutions Configuration Guide*
Byte-Based WRED	Byte-Based Weight Random Early Detection module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS XE software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
No new or modified RFCs are supported, and support for existing RFCs has not been modified.	--

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

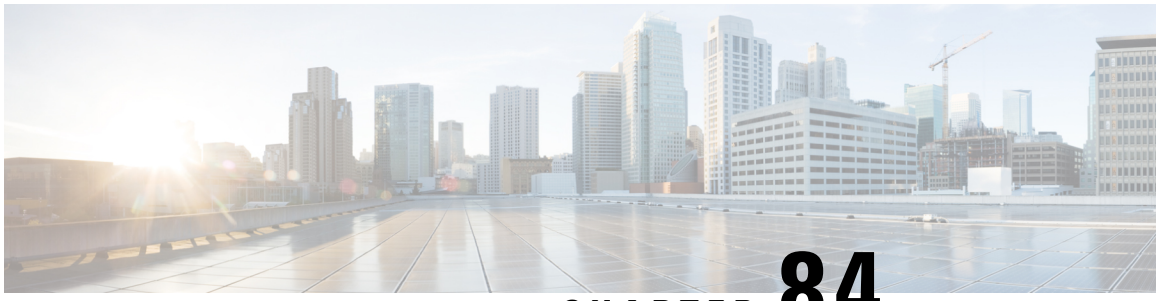
Feature Information for QoS Time-Based Thresholds for WRED and Queue Limit

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 91: Feature Information for QoS Time-Based Thresholds for WRED and Queue Limit

Feature Name	Releases	Feature Information
QoS Time-Based Thresholds for WRED and Queue Limit	Cisco IOS XE Release 3.2S	The QoS Time-Based Thresholds for WRED and Queue Limit feature allows you to specify the Weighted Random Early Detection (WRED) minimum and maximum thresholds or the queue limit threshold in milliseconds (ms). The following commands are introduced or modified: **queue-limit, random-detect precedence, show policy-map, show policy-map interface.**



CHAPTER 84

WRED Explicit Congestion Notification

Prerequisites for WRED-Explicit Congestion Notification

ECN must be configured through the Modular Quality of Service Command-Line Interface (MQC). For more information about the MQC, see the "Applying QoS Features Using the MQC" module.

Information About WRED-Explicit Congestion Notification

WRED-Explicit Congestion Notification Feature Overview

Currently, the congestion control and avoidance algorithms for Transmission Control Protocol (TCP) are based on the idea that packet loss is an appropriate indication of congestion on networks transmitting data using the best-effort service model. When a network uses the best-effort service model, the network delivers data if it can, without any assurance of reliability, delay bounds, or throughput. However, these algorithms and the best-effort service model are not suited to applications that are sensitive to delay or packet loss (for instance, interactive traffic including Telnet, web-browsing, and transfer of audio and video data). Weighted Random Early Detection (WRED), and by extension, Explicit Congestion Notification (ECN), helps to solve this problem.

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

How WRED Works

WRED makes early detection of congestion possible and provides a means for handling multiple classes of traffic. WRED can selectively discard lower priority traffic when the router begins to experience congestion

and provide differentiated performance characteristics for different classes of service. It also protects against global synchronization. Global synchronization occurs as waves of congestion crest, only to be followed by periods of time during which the transmission link is not used to capacity. For these reasons, WRED is useful on any output interface or router where congestion is expected to occur.

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

For more information about WRED, refer to the "Congestion Avoidance Overview" module.

ECN Extends WRED Functionality

WRED drops packets, based on the average queue length exceeding a specific threshold value, to indicate congestion. ECN is an extension to WRED in that ECN marks packets instead of dropping them when the average queue length exceeds a specific threshold value. When configured with the WRED -- Explicit Congestion Notification feature, routers and end hosts would use this marking as a signal that the network is congested and slow down sending packets.

As stated in RFC 3168, *The Addition of Explicit Congestion Notification (ECN) to IP,*implementing ECN requires an ECN-specific field that has two bits--the ECN-capable Transport (ECT) bit and the CE (Congestion Experienced) bit--in the IP header. The ECT bit and the CE bit can be used to make four ECN field combinations of 00 to 11. The first number is the ECT bit and the second number is the CE bit. The table below lists each of the ECT and CE bit combination settings in the ECN field and what the combinations indicate.

Table 92: ECN Bit Setting

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

The ECN field combination 00 indicates that a packet is not using ECN.

The ECN field combinations 01 and 10--called ECT(1) and ECT(0), respectively--are set by the data sender to indicate that the endpoints of the transport protocol are ECN-capable. Routers treat these two field combinations identically. Data senders can use either one or both of these two combinations. For more information about these two field combinations, and the implications of using one over the other, refer to RFC 3168, *The Addition of Explicit Congestion Notification (ECN) to IP.*

The ECN field combination 11 indicates congestion to the endpoints. Packets arriving a full queue of a router will be dropped.

How Packets Are Treated When ECN Is Enabled

If the number of packets in the queue is below the minimum threshold, packets are transmitted. This happens whether or not ECN is enabled, and this treatment is identical to the treatment a packet receives when WRED only is being used on the network.

If the number of packets in the queue is between the minimum threshold and the maximum threshold, one of the following three scenarios can occur:

- If the ECN field on the packet indicates that the endpoints are ECN-capable (that is, the ECT bit is set to 1 and the CE bit is set to 0, or the ECT bit is set to 0 and the CE bit is set to 1)--and the WRED algorithm determines that the packet should have been dropped based on the drop probability--the ECT and CE bits for the packet are changed to 1, and the packet is transmitted. This happens because ECN is enabled and the packet gets marked instead of dropped.
- If the ECN field on the packet indicates that neither endpoint is ECN-capable (that is, the ECT bit is set to 0 and the CE bit is set to 0), the packet may be dropped based on the WRED drop probability. This is the identical treatment that a packet receives when WRED is enabled without ECN configured on the router.
- If the ECN field on the packet indicates that the network is experiencing congestion (that is, both the ECT bit and the CE bit are set to 1), the packet is transmitted. No further marking is required.

If the number of packets in the queue is above the maximum threshold, packets are dropped based on the drop probability. This is the identical treatment a packet receives when WRED is enabled without ECN configured on the router.

Benefits of WRED Explicit Congestion Notification

Improved Method for Congestion Avoidance

This feature provides an improved method for congestion avoidance by allowing the network to mark packets for transmission later, rather than dropping them from the queue. Marking the packets for transmission later accommodates applications that are sensitive to delay or packet loss and provides improved throughput and application performance.

Enhanced Queue Management

Currently, dropped packets indicate that a queue is full and that the network is experiencing congestion. When a network experiences congestion, this feature allows networks to mark the IP header of a packet with a CE bit. This marking, in turn, triggers the appropriate congestion avoidance mechanism and allows the network to better manage the data queues. With this feature, ECN-capable routers and end hosts can respond to congestion before a queue overflows and packets are dropped, providing enhanced queue management.

How to Configure WRED-Explicit Congestion Notification

Configuring Explicit Congestion Notification

To configure ECN, complete the following steps.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name*| **class-default**}
5. **bandwidth** {*bandwidth-kbps* | **percent** *percent*
6. **random-detect**
7. **random-detect ecn**
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Router(config)# policy-map policy1`	Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy. Enters QoS policy-map configuration mode. • Enter the name of the policy map.
Step 4	**class** {*class-name*| **class-default**} **Example:** `Router(config-pmap)# class class-default`	Specifies the name of the class whose policy you want to create or change or specifies the default class (commonly known as the class-default class) before you configure its policy. Enters policy-map-class configuration mode. • Enter the name of the class or enter the **class-default**keyword.

Step 5	**bandwidth** {*bandwidth-kbps*	**percent** *percent*
		Example:
		`Router(config-pmap-c)#`
		`bandwidth percent 35`
			Specifies or modifies the bandwidth (either in kbps or a percentage) allocated for a class belonging to a policy map.
			• Enter the bandwidth in kilobytes per second or enter the bandwidth percentage.

Step 6	**random-detect** **Example:** `Router(config-pmap-c)#`	Enables WRED or distributed WRED (dWRED).

	Command or Action	Purpose
	`random-detect`	
Step 7	**random-detect ecn** **Example:** `Router(config-pmap-c)# random-detect ecn`	Enables ECN.
Step 8	**end** **Example:** `Router(config-pmap-c)# end`	(Optional) Exits policy-map class configuration mode.

Verifying the Explicit Congestion Notification Configuration

To verify the ECN configuration, complete the following steps.

SUMMARY STEPS

1. **enable**
2. **show policy-map**
3. **show policy-map interface**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show policy-map** **Example:** `Router# show policy-map`	If ECN is enabled, displays ECN marking information for a specified policy map.
Step 3	**show policy-map interface** **Example:** `Router# show policy-map interface`	If ECN is enabled, displays ECN marking information for a specified interface.

	Command or Action	Purpose

Step 4	**end**
	Example:
	`Router#`
	`end`
		(Optional) Exits privileged EXEC mode.

Configuration Examples for WRED-Explicit Congestion Notification

Example Enabling ECN

The following example enables ECN in the policy map called pol1:

```
Router(config)# policy-map pol1
 Router(config-pmap)# class class-default
 Router(config-pmap-c)# bandwidth per 70
 Router(config-pmap-c)# random-detect
 Router(config-pmap-c)# random-detect ecn
```

Example Verifying the ECN Configuration

The following is sample output from the **show policy-map** command. The words "explicit congestion notification" (along with the ECN marking information) included in the output indicate that ECN has been enabled.

```
Router# show policy-map
   Policy Map pol1
     Class class-default
       Weighted Fair Queueing
             Bandwidth 70 (%)
             exponential weight 9
             explicit congestion notification
             class    min-threshold    max-threshold    mark-probability
             ------------------------------------------------------------
             ------------------------------------------------------------

             0        -                -                1/10
             1        -                -                1/10
             2        -                -                1/10
             3        -                -                1/10
             4        -                -                1/10
             5        -                -                1/10
             6        -                -                1/10
             7        -                -                1/10
             rsvp     -                -                1/10
```

The following is sample output from the **show policy-map interface**command. The words "explicit congestion notification" included in the output indicate that ECN has been enabled.

```
Router# show policy-map interface
 Serial4/1
 Serial4/1
  Service-policy output:policy_ecn
        Class-map:prec1 (match-all)
          1000 packets, 125000 bytes
          30 second offered rate 14000 bps, drop rate 5000 bps
          Match:ip precedence 1
          Weighted Fair Queueing
            Output Queue:Conversation 42
            Bandwidth 20 (%)
            Bandwidth 100 (kbps)
             (pkts matched/bytes matched) 989/123625
         (depth/total drops/no-buffer drops) 0/455/0
              exponential weight:9
              explicit congestion notification
              mean queue depth:0
     class   Transmitted  Random drop  Tail drop   Minimum     Maximum     Mark
             pkts/bytes   pkts/bytes    pkts/bytes threshold   threshold   probability
       0       0/0          0/0          0/0          20          40        1/10
       1     545/68125      0/0          0/0          22          40        1/10
       2       0/0          0/0          0/0          24          40        1/10
       3       0/0          0/0          0/0          26          40        1/10
       4       0/0          0/0          0/0          28          40        1/10
       5       0/0          0/0          0/0          30          40        1/10
       6       0/0          0/0          0/0          32          40        1/10
       7       0/0          0/0          0/0          34          40        1/10
     rsvp      0/0          0/0          0/0          36          40        1/10
     class   ECN Mark
             pkts/bytes
       0     0/0
       1    43/5375
       2     0/0
       3     0/0
       4     0/0
       5     0/0
       6     0/0
       7     0/0
     rsvp    0/0
```

Additional References

Related Documents

Related Topic	Document Title
QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples	*Cisco IOS Quality of Service Solutions Command Reference*
MQC	"Applying QoS Features Using the MQC" module
Congestion avoidance concepts	"Congestion Avoidance Overview" module

Standards

Standard	Title
No new or modified standards are supported, and support for existing standards has not been modified.	--

MIBs

MIB	MIBs Link
No new or modified MIBs are supported, and support for existing MIBs has not been modified.	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2309	*Internet Performance Recommendation*
RFC 2884	*Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks*
RFC 3168	*The Addition of Explicit Congestion Notification (ECN) to IP*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

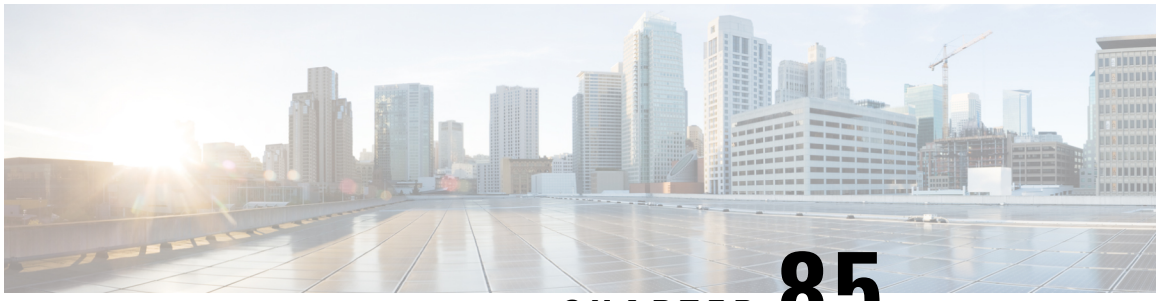
Feature Information for WRED Explicit Congestion Notification

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 93: Feature Information for WRED Explicit Congestion Notification

Feature Name	Software Releases	Feature Configuration Information
WRED Explicit Congestion Notification	Cisco IOS XE Release 2.1	Currently, the congestion control and avoidance algorithms for Transmission Control Protocol (TCP) are based on the idea that packet loss is an appropriate indication of congestion on networks transmitting data using the best-effort service model. When a network uses the best-effort service model, the network delivers data if it can, without any assurance of reliability, delay bounds, or throughput. However, these algorithms and the best-effort service model are not suited to applications that are sensitive to delay or packet loss (for instance, interactive traffic including Telnet, web-browsing, and transfer of audio and video data). Weighted Random Early Detection (WRED), and by extension, Explicit Congestion Notification (ECN), helps to solve this problem. The following commands were introduced or modified: **random-detect ecn**, **show policy-map**, **show policy-map interface**.



CHAPTER 85

Shaping on Dialer Interfaces

The Shaping on Dialer Interfaces feature provides support for Point-to-Point Protocol over Ethernet (PPPoE) and Point-to-Point Protocol over Asynchronous Transfer Mode (PPPoA) configurations on dialer interfaces. The feature provides support for Modular QoS CLI (MQC)-based queuing and shaping that supports per-customer quality of service (QoS). Parent policies are attached to an Ethernet in the First Mile (EFM) interface, and child policies are attached to individual dialer interfaces. Class of service (CoS) values are set by applying a policy to the dialer interface. The feature also enables the collection of queuing statistics on the dialer interface and the polling of traffic counters for dialer interfaces.

Restrictions for Shaping on Dialer Interfaces

- The output queueing policy must have a parent class-default shaper, and any other queueing actions must be configured in a child policy.

Information About Shaping on Dialer Interfaces

QoS on PPP Session on Dialer Interfaces

The Shaping on Dialer Interfaces feature consolidates the output queueing and classification on the egress interface (where all the queueing features are run). The police and set features (such as CoS marking) also work in the output path.

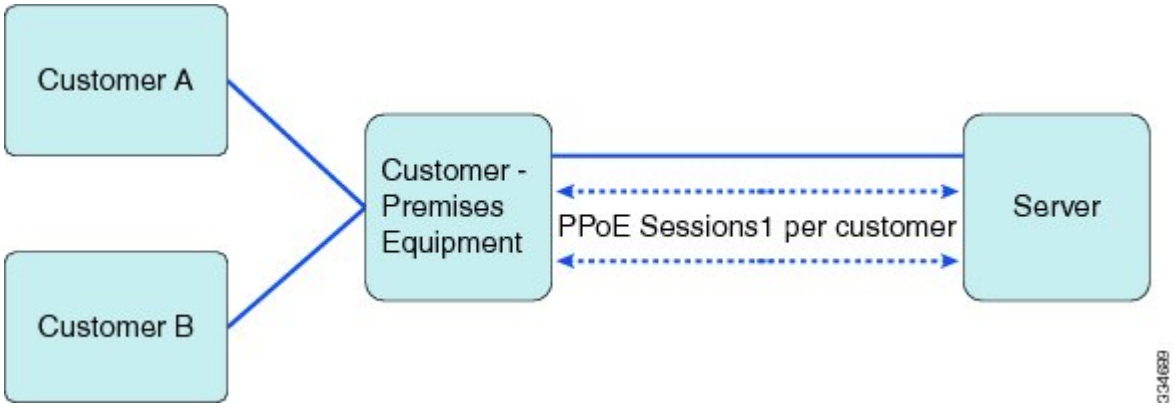
MQC-based QoS queuing and shaping features can be used to attach flat class-default shaped policies to the EFM and attach HQoS parent-shaped policies to the dialer interface.

Policies are applied to the dialer interface using the **service-policy** command. In addition the related show and debug commands display policy and queueing statistics associated with the dialer target.

QoS Dialer Interface Topology

The following figure shows the supported topology for the Shaping on Dialer Interfaces feature:

Figure 111: Shaping on Dialer Interfaces Topology



The Customer Premises Equipment (CPE) is shared between several customers. Each customer connects to the CPE through a VLAN on a Gigabit Ethernet port. The CPE connects to the service over a DSL using an EFM interface (this looks like an Ethernet connection but uses DSL) over which all the incoming VLANs will be forwarded. The traffic for each VLAN (customer) is transmitted in a separate PPP session. Each session is set up using a dialer interface.

How to Configure Shaping on Dialer Interfaces

Configuring an Output Queueing Policy for Dialer Interfaces

Before you begin

Because the dialer target is added to the dynamic target API, the output queueing policy must have a parent class-default shaper with any other queueing actions configured in a child policy.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** *class-name*
5. **priority percent** *percentage*
6. **exit**
7. **class** *class-name*
8. **bandwidth percent** *percentage*
9. **exit**
10. **class** {*class-name* | **class-default**}
11. **fair-queue**

12. **exit**
13. **exit**
14. **policy-map** *policy-map-name*
15. **class class-default**
16. **shape average** *target-bit-rate*
17. **service-policy** *policy-map-name*
18. **exit**
19. **exit**
20. **interface** *type number*
21. **service-policy output** *policy-name*
22. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map child`	Specifies the name of the policy map created earlier and enters policy-map configuration mode. • Enter the policy map name.
Step 4	**class** *class-name* **Example:** `Device(config-pmap)# class voice`	Specifies the name of the class whose policy you want to create and enters policy-map class configuration mode. This class is associated with the class map created earlier. • Enter the name of the class or enter the **class-default** keyword.
Step 5	**priority percent** *percentage* **Example:** `Device(config-pmap-c)# priority percent 30`	Specifies that the amount of guaranteed bandwidth will be specified by the percent of available bandwidth.
Step 6	**exit** **Example:** `Device(config-pmap-c)# exit`	Returns to policy-map configuration mode.

	Command or Action	Purpose
Step 7	**class** *class-name* **Example:** `Device(config-pmap)# class video`	Specifies the name of the class whose policy you want to create and enters policy-map class configuration mode. This class is associated with the class map created earlier. • Enter the name of the class or enter the **class-default** keyword.
Step 8	**bandwidth percent** *percentage* **Example:** `Device(config-pmap-c)# bandwidth percent 50`	Specifies that the amount of guaranteed bandwidth will be specified by the percent of total bandwidth.
Step 9	**exit** **Example:** `Device(config-pmap-c)# exit`	Returns to policy-map configuration mode.
Step 10	**class** {*class-name* | **class-default**} **Example:** `Device(config-pmap)# class class-default`	Specifies the name of the class whose policy you want to create and enters policy-map class configuration mode. This class is associated with the class map created earlier. • Enter the name of the class or enter the **class-default** keyword.
Step 11	**fair-queue** **Example:** `Device(config-pmap-c)# fair-queue`	Enables flow-based fair queueing in this class.
Step 12	**exit** **Example:** `Device(config-pmap-c) exit`	Returns to policy-map configuration mode.
Step 13	**exit** **Example:** `Device(config-pmap) exit`	Returns to global configuration mode.
Step 14	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map parent`	Specifies the name of a policy map and enters policy-map configuration mode. • Enter the policy map name.
Step 15	**class class-default** **Example:** `Device(config-pmap)# class class-default`	Creates the class-default class.

	Command or Action	Purpose
Step 16	**shape average** *target-bit-rate* **Example:** `Device(config-pmap-c)# shape average 1000000`	Specifies average rate traffic shaping as bits-per-second on an interface.
Step 17	**service-policy** *policy-map-name* **Example:** `Device(config-pmap-c)# service policy child`	Configures a service policy policy map.
Step 18	**exit** **Example:** `Device(config-pmap-c) exit`	Returns to policy-map configuration mode.
Step 19	**exit** **Example:** `Device(config-pmap) exit`	Returns to global configuration mode.
Step 20	**interface** *type number* **Example:** `Device(config)# interface Dialer 0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 21	**service-policy output** *policy-name* **Example:** `Device(config-if)# service-policy output parent`	Attaches a policy map to an output interface that will be used as the service policy for the interface.
Step 22	**exit** **Example:** `Device(config-if) exit`	Returns to global configuration mode.

Configuring QoS for PPPoEoA for Dialer Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [**name-tag**]
4. **no ip address**
5. **no atm ilmi-keepalive**
6. **exit**
7. **interface** *type number* [**name-tag**]
8. **pvc** *vpi*/*vci*

9. **vbr-nrt** *output-pcr output-scr*
10. **pppoe-client dial-pool-number** *number*
11. **exit**
12. **exit**
13. **interface** *type number* [**name-tag**]
14. **mtu** *ip-address*
15. **ip address** *ip-address mask*
16. **encapsulation** *encapsulation-type*
17. **dialer pool** *number*
18. **dialer-group** *number*
19. **service-policy output** *name*
20. **exit**
21. **dialer-list** *dialer-group* **protocol** *protocol-name* **permit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface ATM 0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 4	**no ip address** **Example:** `Device(config-if)# no ip address`	Disables IP processing on the interface.
Step 5	**no atm ilmi-keepalive** **Example:** `Device(config-if)# no atm ilmi-keepalive`	Disables Interim Local Management Interface (ILMI) keepalives on the interface.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.

	Command or Action	Purpose
Step 7	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface ATM 0.1 point-to-point`	Configures an interface type and enters interface configuration mode. • Enter the interface type, number, and name.
Step 8	**pvc** *vpi/vci* **Example:** `Device(config-if)# pvc 4/46`	Creates an ATM permanent virtual circuit (PVC), and enters ATM virtual circuit configuration mode. • Enter the ATM network virtual path identifier (VPI) and ATM network virtual channel identifier (VCI) for this PVC.
Step 9	**vbr-nrt** *output-pcr output-scr* **Example:** `Device(config-if-atm-vc)# vbr-nrt 738 738`	Configures the variable bit rate-nonreal time (VBR-NRT) quality of service (QoS) and specifies the output peak cell rate (PCR), and output sustainable cell rate (SCR) for an ATM permanent virtual circuit (PVC).
Step 10	**pppoe-client dial-pool-number** *number* **Example:** `Device(config-if-atm-vc)# pppoe-client dial-pool-number 1`	Configures a PPP over Ethernet (PPPoE) client and specifies the dial-on-demand routing (DDR) functionality.
Step 11	**exit** **Example:** `Device(config-if-atm-vc)# exit`	Exits ATM virtual circuit configuration mode.
Step 12	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 13	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface Dialer 0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 14	**mtu** *ip-address* **Example:** `Device(config-if)# mtu 1200`	Adjusts the maximum packet size or maximum transmission unit (MTU) size.
Step 15	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.16.0.0 255.0.0.0`	Sets the primary IP address for the interface. • Enter the IP address and the IP address mask.

	Command or Action	Purpose
Step 16	**encapsulation** *encapsulation-type* **Example:** `Device(config-if)# encapsulation ppp`	Sets the encapsulation method used by the interface.
Step 17	**dialer pool** *number* **Example:** `Device(config-if)# dialer pool 1`	Specifies the dialing pool that the dialer interface uses to connect to a specific destination subnetwork.
Step 18	**dialer-group** *number* **Example:** `Device(config-if)# dialer-group 1`	Controls access by configuring the interface to belong to a specific dialing group.
Step 19	**service-policy output** *name* **Example:** `Device(config-if)# service-policy output dialer-output-sp`	Attaches a policy map to an output interface that will be used as the service policy for the interface.
Step 20	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 21	**dialer-list** *dialer-group* **protocol** *protocol-name* **permit** **Example:** `Device(config)# dialer-list 1 protocol ip permit`	Defines a dial-on-demand routing (DDR) dialer list for dialing by protocol or by a combination of a protocol and a previously defined access list.

Configuring QoS for PPPoE for Dialer Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [**name-tag**]
4. **ppp enable group** *group-name*
5. **pppoe-client dial-pool-number** *number*
6. **exit**
7. **interface** *type number* [**name-tag**]
8. **mtu** *ip-address*
9. **ip address** *ip-address mask*
10. **encapsulation** *encapsulation-type*

11. **dialer pool** *number*
12. **dialer-group** *number*
13. **service-policy output** *name*
14. **exit**
15. **dialer-list** *dialer-group* **protocol** *protocol-name* **permit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface Ethernet 0/0`	Configures an interface type and enters interface configuration mode. • Enter the interface type, number, and name.
Step 4	**ppp enable group** *group-name* **Example:** `Device(config-if)# ppp enable group global`	Enables PPPoE sessions on an Ethernet interface or subinterface.
Step 5	**pppoe-client dial-pool-number** *number* **Example:** `Device(config-if)# pppoe-client dial-pool-number 1`	Configures a PPPoE client and to specify the dial-on-demand routing (DDR) functionality.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 7	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface Dialer 0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 8	**mtu** *ip-address* **Example:**	Adjusts the maximum packet size or maximum transmission unit (MTU) size.

	Command or Action	Purpose
	`Device(config-if)# mtu 1200`	
Step 9	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.16.0.0 255.0.0.0`	Sets the primary IP address for the interface. • Enter the IP address and the IP address mask.
Step 10	**encapsulation** *encapsulation-type* **Example:** `Device(config-if)# encapsulation ppp`	Sets the encapsulation method used by the interface.
Step 11	**dialer pool** *number* **Example:** `Device(config-if)# dialer pool 1`	Specifies the dialing pool that the dialer interface uses to connect to a specific destination subnetwork.
Step 12	**dialer-group** *number* **Example:** `Device(config-if)# dialer-group 1`	Controls access by configuring the interface to belong to a specific dialing group.
Step 13	**service-policy output** *name* **Example:** `Device(config-if)# service-policy output dialer-output-sp`	Attaches a policy map to an output interface that will be used as the service policy for the interface.
Step 14	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 15	**dialer-list** *dialer-group* **protocol** *protocol-name* **permit** **Example:** `Device(config)# dialer-list 1 protocol ip permit`	Defines a dial-on-demand routing (DDR) dialer list for dialing by protocol or by a combination of a protocol and a previously defined access list.

Configuring QoS for PPPoA for Dialer Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [**name-tag**]

4. **pvc** *vpi*/*vci*
5. **vbr-nrt** *output-pcr output-scr output-maxburstsize*
6. **dialer pool-member** *number*
7. **protocol** *protocol*
8. **exit**
9. **exit**
10. **interface** *type number* [**name-tag**]
11. **mtu** *ip-address*
12. **ip address** *ip-address mask*
13. **encapsulation** *encapsulation-type*
14. **dialer pool** *number*
15. **dialer-group** *number*
16. **service-policy output** *name*
17. **exit**
18. **dialer-list** *dialer-group* **protocol** *protocol-name* **permit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface ATM 0.1 point-to-point`	Configures an interface type and enters interface configuration mode. • Enter the interface type, number, and name.
Step 4	**pvc** *vpi*/*vci* **Example:** `Device(config-if)# pvc 4/46`	Creates an ATM permanent virtual circuit (PVC), and enters ATM virtual circuit configuration mode. • Enter the ATM network virtual path identifier (VPI) and ATM network virtual channel identifier (VCI) for this PVC.
Step 5	**vbr-nrt** *output-pcr output-scr output-maxburstsize* **Example:** `Device(config-if-atm-vc)# vbr-nrt 738 738 32`	Configures the variable bit rate-nonreal time (VBR-NRT) quality of service (QoS) and specifies the output peak cell rate (PCR), output sustainable cell rate (SCR), and output maximum burst cell size for an ATM permanent virtual circuit (PVC).

	Command or Action	Purpose
Step 6	**dialer pool-member** *number* **Example:** `Device(config-if-atm-vc)# dialer pool-member 1`	Configures a physical interface to be a member of a dialer profiles dialing pool.
Step 7	**protocol** *protocol* **Example:** `Device(config-if-atm-vc)# protocol ppp dialer`	Configures a static map for an ATM permanent virtual circuit (PVC), switched virtual circuit (SVC), or virtual circuit (VC) class.
Step 8	**exit** **Example:** `Device(config-if-atm-vc)# exit`	Exits ATM virtual circuit configuration mode.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 10	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface Dialer 0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 11	**mtu** *ip-address* **Example:** `Device(config-if)# mtu 1200`	Adjusts the maximum packet size or maximum transmission unit (MTU) size.
Step 12	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 172.16.0.0 255.0.0.0`	Sets the primary IP address for the interface. • Enter the IP address and the IP address mask.
Step 13	**encapsulation** *encapsulation-type* **Example:** `Device(config-if)# encapsulation ppp`	Sets the encapsulation method used by the interface.
Step 14	**dialer pool** *number* **Example:** `Device(config-if)# dialer pool 1`	Specifies the dialing pool that the dialer interface uses to connect to a specific destination subnetwork.
Step 15	**dialer-group** *number* **Example:**	Controls access by configuring the interface to belong to a specific dialing group.

	Command or Action	Purpose
	`Device(config-if)# dialer-group 1`	
Step 16	**service-policy output** *name* **Example:** `Device(config-if)# service-policy output dialer-output-sp`	Attaches a policy map to an output interface that will be used as the service policy for the interface.
Step 17	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 18	**dialer-list** *dialer-group* **protocol** *protocol-name* **permit** **Example:** `Device(config)# dialer-list 1 protocol ip permit`	Defines a dial-on-demand routing (DDR) dialer list for dialing by protocol or by a combination of a protocol and a previously defined access list, .

Configuring QoS for Multiple Sessions on Dialer Interfaces

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [**name-tag**]
4. **ppp enable group** *group-name*
5. **pppoe-client dial-pool-number** *number*
6. **pppoe-client dial-pool-number** *number*
7. **pppoe-client dial-pool-number** *number*
8. **exit**
9. **interface** *type number* [**name-tag**]
10. **dialer pool** *number*
11. **service-policy output** *name*
12. **exit**
13. **interface** *type number* [**name-tag**]
14. **dialer pool** *number*
15. **service-policy output** *name*
16. **exit**
17. **interface** *type number* [**name-tag**]
18. **dialer pool** *number*
19. **service-policy output** *name*
20. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface Ethernet 0/0`	Configures an interface type and enters interface configuration mode. • Enter the interface type, number, and name.
Step 4	**ppp enable group** *group-name* **Example:** `Device(config-if)# ppp enable group global`	Enables PPPoE sessions on an Ethernet interface or subinterface.
Step 5	**pppoe-client dial-pool-number** *number* **Example:** `Device(config-if)# pppoe-client dial-pool-number 1`	Configures a PPPoE client and to specify the dial-on-demand routing (DDR) functionality.
Step 6	**pppoe-client dial-pool-number** *number* **Example:** `Device(config-if)# pppoe-client dial-pool-number 2`	Configures a PPPoE client and to specify the dial-on-demand routing (DDR) functionality.
Step 7	**pppoe-client dial-pool-number** *number* **Example:** `Device(config-if)# pppoe-client dial-pool-number 3`	Configures a PPPoE client and to specify the dial-on-demand routing (DDR) functionality.
Step 8	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 9	**interface** *type number* [**name-tag**] **Example:**	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.

	Command or Action	Purpose
	`Device(config)# interface Dialer 0`	
Step 10	**dialer pool** *number* **Example:** `Device(config-if)# dialer pool 1`	Specifies the dialing pool that the dialer interface uses to connect to a specific destination subnetwork.
Step 11	**service-policy output** *name* **Example:** `Device(config-if)# service-policy output dialer-output-sp`	Attaches a policy map to an output interface that will be used as the service policy for the interface.
Step 12	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 13	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface Dialer 1`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 14	**dialer pool** *number* **Example:** `Device(config-if)# dialer pool 2`	Specifies the dialing pool that the dialer interface uses to connect to a specific destination subnetwork.
Step 15	**service-policy output** *name* **Example:** `Device(config-if)# service-policy output dialer-output-sp`	Attaches a policy map to an output interface that will be used as the service policy for the interface.
Step 16	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 17	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface Dialer 2`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 18	**dialer pool** *number* **Example:**	Specifies the dialing pool that the dialer interface uses to connect to a specific destination subnetwork.

	Command or Action	Purpose
	`Device(config-if)# dialer pool 3`	
Step 19	**service-policy output** *name* **Example:** `Device(config-if)# service-policy output dialer-output-sp`	Attaches a policy map to an output interface that will be used as the service policy for the interface.
Step 20	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.

Applying CoS Values to a Dialer Interface

Class of Service (CoS) values are set by applying a policy to the dialer interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class class-default**
5. **set cos** *cos-value*
6. **exit**
7. **exit**
8. **interface** *type number* [**name-tag**]
9. **service-policy output** *name*
10. **exit**
11. **interface** *type number* [**name-tag**]
12. **encapsulation** *encapsulation-type*
13. **pppoe-client dial-pool-number** *number*
14. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:**	Enters global configuration mode.

	Command or Action	Purpose
	`Device# configure terminal`	
Step 3	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map output_cos`	Specifies the name of the policy map created earlier and enters policy-map configuration mode. • Enter the policy map name.
Step 4	**class class-default** **Example:** `Device(config-pmap)# class class-default`	Creates the default class for traffic classification and enters policy-map class configuration mode.
Step 5	**set cos** *cos-value* **Example:** `Device(config-pmap-c)# set cos 1`	Specifies an IEEE 802.1Q CoS value from 0 to 7.
Step 6	**exit** **Example:** `Device(config-pmap-c)# exit`	Returns to policy-map configuration mode.
Step 7	**exit** **Example:** `Device(config-pmap)# exit`	Returns to global configuration mode.
Step 8	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface Dialer 1`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 9	**service-policy output** *name* **Example:** `Device(config-if)# service-policy output output-cos`	Attaches a policy map to an output interface that will be used as the service policy for the interface.
Step 10	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode.
Step 11	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface Ethernet 0.10`	Configures an interface type and enters sub-interface configuration mode. • Enter the interface type and number.

	Command or Action	Purpose
Step 12	**encapsulation** *encapsulation-type* **Example:** `Device(config-subif)# encapsulation dot1q 10`	Sets the encapsulation method used by the interface.
Step 13	**pppoe-client dial-pool-number** *number* **Example:** `Device(config-subif)# pppoe-client dial-pool-number 1`	Configures a PPPoE client and to specify the dial-on-demand routing (DDR) functionality.
Step 14	**exit** **Example:** `Device(config-subif)# exit`	Returns to global configuration mode.

Configuration Examples for Shaping on Dialer Interfaces

Example: Configuring Output Queuing Policy for a Dialer Interface

The following example shows how to configure parent and child policy maps and how to attach the parent map to the dialer interface:

```
Device(config)# policy-map childExample
 Device(config-pmap)# class voice
  Device(config-pmap-c)# priority percent 30
  Device(config-pmap-c)# exit

 Device(config-pmap)# class video
  Device(config-pmap-c)# bandwidth percent 50
  Device(config-pmap-c)# exit

 Device(config-pmap)# class class-default
  Device(config-pmap-c)# fair-queue
  Device(config-pmap-c)# exit

Device(config)# policy-map parent
 Device(config-pmap)# class class-default
  Device(config-pmap-c)# shape average 1000000
  Device(config-pmap-c)# service-policy child
  Device(config-pmap-c)# exit

Device(config)# interface dialer 0
 Device(config-if)# service-policy output parent
```

Example: Configuring QoS for PPPoEoA for a Dialer Interface

```
Device(config)# interface ATM 0
 Device(config-if)# no ip address
 Device(config-if)# no atm ilmi-keepalive
 Device(config-if)# exit

Device(config)# interface ATM 0.1 point-to-point
 Device(config-if)# ip address 192.168.0.0 255.255.255.224
 Device(config-if)# pvc 4/46
  Device(config-if-atm-vc)# vbr-nrt 738 738
  Device(config-if-atm-vc)# pppoe-client dial-pool-number 1
  Device(config-if-atm-vc)# exit
 Device(config-if)# exit

Device(config)# interface Dialer 0
 Device(config-if)# mtu 1200
 Device(config-if)# ip address 172.16.0.0 255.0.0.0
 Device(config-if)# encapsulation ppp
 Device(config-if)# dialer pool 1
 Device(config-if)# dialer-group 1
 Device(config-if)# service-policy output dialer-output-sp
!
Device(config)# dialer-list 1 protocol ip permit
```

Example: Configuring QoS for a PPPoE on a Dialer Interface

```
Device(config)# interface ethernet 0/0
 Device(config-if)# pppoe enable group global
 Device(config-if)# pppoe-client dial-pool-number 1
 Device(config-if)# exit

Device(config)# interface Dialer 0
 Device(config-if)# mtu 1200
 Device(config-if)# ip address 172.16.0.0 255.0.0.0
 Device(config-if)# encapsulation ppp
 Device(config-if)# dialer pool 1
 Device(config-if)# dialer-group 1
 Device(config-if)# service-policy output dialer-output-sp
 Device(config-if)# exit

Device(config)# dialer-list 1 protocol ip permit
```

Example: Configuring QoS for PPPoA on a Dialer Interface

```
Device(config)# interface ATM 0.1 point-to-point
 Device(config-if)# ip address 192.168.0.0 255.255.255.224
 Device(config-if)# pvc 4/46
  Device(config-if-atm-vc)# vbr-nrt 738 738
  Device(config-if-atm-vc)# dialer pool-member 1
  Device(config-if-atm-vc)# protocol ppp dialer
  Device(config-if-atm-vc)# exit
 Device(config-if)# exit
```

```
Device(config)# interface Dialer 0
 Device(config-if)# mtu 1200
 Device(config-if)# ip address 172.16.0.0 255.0.0.0
 Device(config-if)# encapsulation ppp
 Device(config-if)# dialer pool 1
 Device(config-if)# dialer-group 1
 Device(config-if)# service-policy output dialer-output-sp
 Device(config-if)# exit

Device(config)# dialer-list 1 protocol ip permit
```

Example: Configuring QoS for Multiple Sessions on a Dialer Interface

```
Device(config)# interface ethernet 0/0
 Device(config-if)# pppoe enable group global
 Device(config-if)# pppoe-client dial-pool-number 1
 Device(config-if)# pppoe-client dial-pool-number 2
 Device(config-if)# pppoe-client dial-pool-number 3
 Device(config-if)# exit

Device(config)# interface Dialer 0
 Device(config-if)# dialer pool 1
 Device(config-if)# service-policy output dialer-output-sp
 Device(config-if)# exit

Device(config)# interface Dialer 1
 Device(config-if)# dialer pool 2
 Device(config-if)# service-policy output dialer-output-sp
 Device(config-if)# exit

Device(config)# interface Dialer 2
 Device(config-if)# dialer pool 3
 Device(config-if)# service-policy output dialer-output-sp
 Device(config-if)# exit
```

Example: Applying CoS Values to a Dialer Interface

```
Device> enable
Device# configure terminal
Device(config)# policy-map output_cos
Device(config-pmap)# class class-default
Device(config-pmap-c)# set cos 1
Device(config-pmap-c)# exit
Device(config-pmap)# exit
Device(config)# interface Dialer 1
Device(config-if)# service-policy output output-cos
Device(config-if)# exit
Device(config)# interface Ethernet 0.10
Device(config-subif)# encapsulation dot1q 10
Device(config-subif)# pppoe-client dial-pool-number 1
Device(config-subif)# exit
```

Additional References for Shaping on Dialer Interfaces

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*
MQC	*QoS: Modular QoS: Command-Line Interface Configuration Guide*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

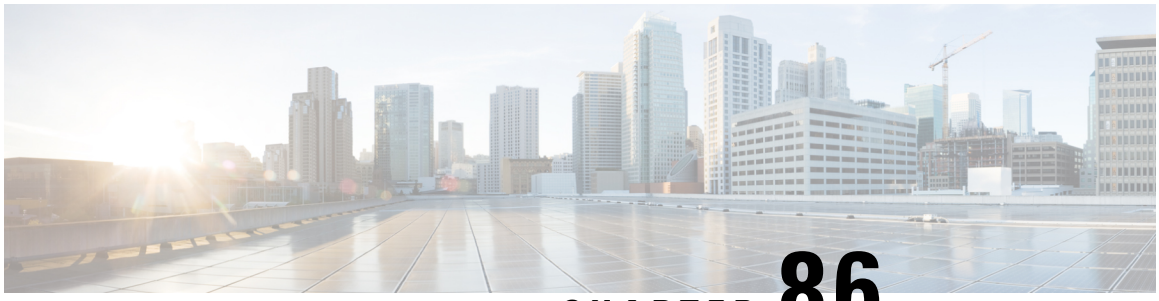
Feature Information for Shaping on Dialer Interfaces

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 94: Feature Information for Shaping on Dialer Interfaces

Feature Name	Releases	Feature Information
Shaping on Dialer Interfaces	15.3(1)T Cisco IOS XE Release 3.13S	The Shaping on Dialer Interfaces feature provides support for PPPoE/A configurations on dialer interfaces.



CHAPTER 86

DiffServ Compliant WRED

DiffServ Compliant WRED extends the functionality of Weighted Random Early Detection to enable support for DiffServ and Assured Forwarding (AF) per hop behavior (PHB). This feature enables customers to implement AF PHB by coloring packets according to Differentiated Services Code Point (DSCP) values and then assigning preferential drop probabilities to those packets.

Note This feature can be used with IP packets only. It is not intended for use with Multiprotocol Label Switching (MPLS)-encapsulated packets.

Information About DiffServ Compliant WRED

Differentiated Services for WRED

Differentiated Services is a multiple service model that can satisfy differing Quality of Service (QoS) requirements. With Differentiated Services, 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. The DiffServ Compliant WRED feature enables WRED to use either the 6-bit differentiated services code point (DSCP) or the IP Precedence setting in IP packets when it calculates the drop probability for a packet. The DSCP value is the first six bits of the IP type of service (ToS) byte.

Usage Guidelines for DiffServ Compliant WRED

To configure the DiffServ Compliant WRED feature, first specify the policy map, add the class, and configure the bandwidth or shape for the class. If you want WRED to use the DSCP value when it calculates the drop probability, use the *dscp-based* argument with the **random-detect** command to specify the DSCP value and then use the **random-detect dscp** command to modify the default minimum and maximum thresholds for the DSCP value. If you want WRED to use the IP Precedence value when it calculates the drop probability, use

the *precedence-based* argument with the **random-detect** command to specify the IP Precedence value. This configuration can then be applied wherever policy maps are attached (for example, at the interface level, the per-VC level, or the shaper level).

Remember the following points when using the commands included with this feature:

- If you use the *dscp-based* argument, WRED will use the DSCP value to calculate the drop probability.
- If you use the *precedence-based* argument, WRED will use the IP Precedence value to calculate the drop probability.
- The *dscp-based* and *precedence-based* arguments are mutually exclusive.
- If you do not specify either argument, WRED will use the IP Precedence value to calculate the drop probability (the default method).

How to Configure DiffServ Compliant WRED

Configuring DiffServ Compliant WRED

This example configures DiffServ Compliant WRED to use the DSCP value to calculate the drop probability for a packet.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type number* [**name-tag**]
4. **class-map** *class-map-name*
5. **match** *match-criterion*
6. **policy-map** *policy-map-name*
7. **class** {*class-name* | **class-default**}
8. **bandwidth** {*kbps* | **remaining** *percentage* | **percent** *percentage*}
9. **random-detect** [**dscp-based** | **precedence-based**]
10. **random-detect dscp** *dscp-value min-threshold max-threshold* [*mark-probability-denominator*]
11. **exit**
12. **exit**
13. **interface** *type number* [**name-tag**]
14. **service-policy output** *policy-map-name*
15. **end**
16. **show policy-map interface** *type number*
17. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 4	**class-map** *class-map-name* **Example:** `Device(config-if)# class-map diffservclass`	Specifies the name of the class map to be created and enters QoS class-map configuration mode.
Step 5	**match** *match-criterion* **Example:** `Device(config-cmap)# match any`	Configures the match criteria for a class map.
Step 6	**policy-map** *policy-map-name* **Example:** `Device(config-cmap)# policy-map diffservpm`	Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy, and enters QoS policy-map configuration mode.
Step 7	**class** {*class-name* | **class-default**} **Example:** `Device(config-pmap)# class diffservclass`	Specifies the name of the class whose policy you want to create or change or specifies the default class (commonly known as the class-default class) before you configure its policy. • Enters QoS policy-map class configuration mode.
Step 8	**bandwidth** {*kbps* | **remaining** *percentage* | **percent** *percentage*} **Example:** `Device(config-pmap-c)# bandwidth percent 30`	Specifies the bandwidth allocated for a class belonging to a policy map.
Step 9	**random-detect** [**dscp-based** | **precedence-based**] **Example:** `Device(config-pmap-c)# random-detect dscp-based`	Configures WRED for a class in a policy map.

	Command or Action	Purpose
Step 10	**random-detect dscp** *dscp-value min-threshold max-threshold* [*mark-probability-denominator*] **Example:** `Device(config-pmap-c)# random-detect dscp af11 10000 30000 25`	Changes the minimum and maximum packet thresholds for the differentiated services code point (DSCP) value.
Step 11	**exit** **Example:** `Device(config-pmap-c)# exit`	Exits QoS policy-map class configuration mode.
Step 12	**exit** **Example:** `Device(config-pmap)# exit`	Exits QoS policy-map configuration mode.
Step 13	**interface** *type number* [**name-tag**] **Example:** `Device(config)# interface GigabitEthernet 0/0/0`	Configures an interface type and enters interface configuration mode. • Enter the interface type and number.
Step 14	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output policy1`	Attaches a policy map to an output interface. • Enter the policy map name. **Note** Policy maps can be configured on ingress or egress routers. They can also be attached in the input or output direction of an interface. The direction (input or output) and the router (ingress or egress) to which the policy map should be attached varies according your network configuration. When using the **service-policy** command to attach the policy map to an interface, be sure to choose the router and the interface direction that are appropriate for your network configuration.
Step 15	**end** **Example:** `Device(config-if)# end`	Returns to privileged EXEC mode.
Step 16	**show policy-map interface** *type number* **Example:** `Device# show policy-map interface GigabitEthernet 0/0/0`	(Optional) Displays the traffic statistics of all classes that are configured for all service policies either on the specified interface or subinterface or on a specific PVC on the interface. • Enter the interface type and number.

	Command or Action	Purpose
Step 17	**exit** **Example:** `Device# exit`	(Optional) Exits privileged EXEC mode.

Configuration Examples for DiffServ Compliant WRED

Example: DiffServ compliant WRED

The following example enables WRED to use the DSCP value 8 for the class c1. The minimum threshold for the DSCP value 8 is 24 and the maximum threshold is 40. The last line attaches the traffic policy to the output interface or VC p1.

```
Device(config)# class-map c1
Device(config-cmap)# match ip precedence 1
Device(config-cmap)# policy-map p1
Device(config-pmap)# class c1
Device(config-pmap-c)# bandwidth 48
Device(config-pmap-c)# random-detect dscp-based
Device(config-pmap-c)# random-detect dscp 8 24 40 (bytes/ms)
Device(config-if)# service-policy output p1
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
QoS commands	*Cisco IOS Quality of Service Solutions Command Reference*
MQC	*QoS: Modular QoS: Command-Line Interface Configuration Guide*

Standards and RFCs

Standard/RFC	Title
RFC 2474	*Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*
RFC 2475	*An Architecture for Differentiated Services Framework*

Standard/RFC	Title
RFC 2597	*Assured Forwarding PHB*
RFC 2598	*An Expedited Forwarding PHB*

MIBs

MIB	MIBs Link
CISCO-CLASS-BASED-QOS-MIB CISCO-CLASS-BASED-QOS-CAPABILITY-MIB	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for DiffServ Compliant WRED

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 95: Feature Information for DiffServ Compliant WRED

Feature Name	Releases	Feature Information
DiffServ Compliant WRED	Cisco IOS XE Release 3.6S	DiffServ Compliant WRED extends the functionality of WRED to enable support for DiffServ and AF per-hop behavior. In Cisco IOS XE Release 3.6S, support was added for the Cisco ASR 903 Router. The following commands were introduced or modified: **random-detect**, **random-detect dscp**, **random-detect precedence**.