QoS: Congestion Management Configuration Guide, Cisco IOS Release 15M&T

First Published: March 04, 2013

Last Modified: March 04, 2013

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

CHAPTER 1

Congestion Management Overview

Congestion management features allow you to control congestion by determining the order in which packets are sent out an interface based on priorities assigned to those packets. Congestion management entails the creation of queues, assignment of packets to those queues based on the classification of the packet, and scheduling of the packets in a queue for transmission. The congestion management QoS feature offers four types of queueing protocols, each of which allows you to specify creation of a different number of queues, affording greater or lesser degrees of differentiation of traffic, and to specify the order in which that traffic is sent.

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

This module discusses these four types of queueing, which constitute the congestion management QoS features:

- FIFO (first-in, first-out). FIFO entails no concept of priority or classes of traffic. With FIFO, transmission of packets out the interface occurs in the order the packets arrive.
- Weighted fair queueing (WFQ). WFQ offers dynamic, fair queueing that divides bandwidth across queues of traffic based on weights. (WFQ ensures that all traffic is treated fairly, given its weight.) To understand how WFQ works, consider the queue for a series of File Transfer Protocol (FTP) packets as a queue for the collective and the queue for discrete interactive traffic packets as a queue for the individual. Given the weight of the queues, WFQ ensures that for all FTP packets sent as a collective an equal number of individual interactive traffic packets are sent.)

Given this handling, WFQ ensures satisfactory response time to critical applications, such as interactive, transaction-based applications, that are intolerant of performance degradation. For serial interfaces at E1 (2.048 Mbps) and below, flow-based WFQ is used by default. When no other queueing strategies are configured, all other interfaces use FIFO by default.

There are four types of WFQ:

- • Flow-based WFQ (WFQ)

  • Distributed WFQ (DWFQ)

  - Class-based WFQ (CBWFQ)
  - Distributed class-based WFQ (DCBWFQ)

- Custom queueing (CQ). With CQ, bandwidth is allocated proportionally for each different class of traffic. CQ allows you to specify the number of bytes or packets to be drawn from the queue, which is especially useful on slow interfaces.
- Priority queueing (PQ). With PQ, packets belonging to one priority class of traffic are sent before all lower priority traffic to ensure timely delivery of those packets.

Note You can assign only one queueing mechanism type to an interface.

Note A variety of queueing mechanisms can be configured using multilink, for example, Multichassis Multilink PPP (MMP). However, if only PPP is used on a tunneled interface--for example, virtual private dialup network (VPND), PPP over Ethernet (PPPoE), or PPP over Frame Relay (PPPoFR)--no queueing can be configured on the virtual interface.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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.

Why Use Congestion Management

Heterogeneous networks include many different protocols used by applications, giving rise to the need to prioritize traffic in order to satisfy time-critical applications while still addressing the needs of less time-dependent applications, such as file transfer. Different types of traffic sharing a data path through the network can interact with one another in ways that affect their application performance. If your network is designed to support different traffic types that share a single data path between routers, you should consider using congestion management techniques to ensure fairness of treatment across the various traffic types.

Here are some broad factors to consider in determining whether to configure congestion management QoS:

- Traffic prioritization is especially important for delay-sensitive, interactive transaction-based applications--for instance, desktop video conferencing--that require higher priority than do file transfer applications. However, use of WFQ ensures that all traffic is treated fairly, given its weight, and in a dynamic manner. For example, WFQ addresses the requirements of the interactive application without penalizing the FTP application.
- Prioritization is most effective on WAN links where the combination of bursty traffic and relatively lower data rates can cause temporary congestion.
- Depending on the average packet size, prioritization is most effective when applied to links at T1/E1 bandwidth speeds or lower.
- If users of applications running across your network notice poor response time, you should consider using congestion management features. Congestion management features are dynamic, tailoring themselves to the existing network conditions. However, consider that if a WAN link is constantly congested, traffic prioritization may *not* resolve the problem. Adding bandwidth might be the appropriate solution.
- If there is no congestion on the WAN link, there is no reason to implement traffic prioritization.

The following list summarizes aspects you should consider in determining whether you should establish and implement a queueing policy for your network:

- Determine if the WAN is congested--that is, whether users of certain applications perceive a performance degradation.
- Determine your goals and objectives based on the mix of traffic you need to manage and your network topology and design. In identifying what you want to achieve, consider whether your goal is among the following:
  - To establish fair distribution of bandwidth allocation across all of the types of traffic you identify.
  - To grant strict priority to traffic from special kinds of applications you service--for example, interactive multimedia applications--possibly at the expense of less-critical traffic you also support.
  - To customize bandwidth allocation so that network resources are shared among all of the applications you service, each having the specific bandwidth requirements you have identified.
  - To effectively configure queueing. You must analyze the types of traffic using the interface and determine how to distinguish them. See the "Classification Overview" module for a description of how packets are classified.

After you assess your needs, review the available congestion management queueing mechanisms described in this module and determine which approach best addresses your requirements and goals.

- Configure the interface for the kind of queueing strategy you have chosen, and observe the results.

Traffic patterns change over time, so you should repeat the analysis process described in the second bullet periodically, and adapt the queueing configuration accordingly.

See the following section Deciding Which Queueing Policy to Use for elaboration of the differences among the various queueing mechanisms.

Deciding Which Queueing Policy to Use

This section looks briefly at some of the differences between the types of queueing and includes a table that compares the main queueing strategies.

FIFO queueing performs no prioritization of data packets on user data traffic. It entails no concept of priority or classes of traffic. When FIFO is used, ill-behaved sources can consume available bandwidth, bursty sources can cause delays in time-sensitive or important traffic, and important traffic may be dropped because less important traffic fills the queue.

Consider these differences in deciding whether to use CQ or PQ:

- CQ guarantees some level of service to all traffic because you can allocate bandwidth to all classes of traffic. You can define the size of the queue by determining its configured packet-count capacity, thereby controlling bandwidth access.
- PQ guarantees strict priority in that it ensures that one type of traffic will be sent, possibly at the expense of all others. For PQ, a low priority queue can be detrimentally affected, and, in the worst case, never allowed to send its packets if a limited amount of bandwidth is available or if the transmission rate of critical traffic is high.

In deciding whether to use WFQ or one of the other two queueing types, consider these differences among WFQ and PQ and CQ:

- WFQ does not require configuration of access lists to determine the preferred traffic on a serial interface. Rather, the fair queue algorithm dynamically sorts traffic into messages that are part of a conversation.
- Low-volume, interactive traffic gets fair allocation of bandwidth with WFQ, as does high-volume traffic such as file transfers.
- Strict priority queueing can be accomplished with WFQ by using the IP RTP Priority, Frame Relay IP RTP Priority, low latency queueing (LLQ), distributed low latency queueing, low latency queueing for Frame Relay, or Frame Relay PVC Interface Priority Queueing features. Strict PQ allows delay-sensitive data such as voice to be dequeued and sent before packets in other queues are dequeued.

The table below compares the salient features of flow-based WFQ, CBWFQ and DCBWFQ, CQ, and PQ.

Table 1: Queueing Comparison

	Flow-Based WFQ	CBWFQ/DCBWFQ	CQ	PQ
Number of Queues	Configurable number of queues (256 user queues, by default)	One queue per class, up to 64 classes	16 user queues	4 queues

	Flow-Based WFQ	CBWFQ/DCBWFQ	CQ	PQ
Kind of Service	• Ensures fairness among all traffic flows based on weights • Strict priority queueing is available through use of the IP RTP Priority or Frame Relay IP RTP Priority features	• Provides class bandwidth guarantee for user-defined traffic classes • Provides flow-based WFQ support for nonuser-defined traffic classes • Strict priority queueing is available through use of the IP RTP Priority, Frame Relay IP RTP Priority, LLQ, Distributed LLQ, and LLQ for Frame Relay features	• Round-robin service	• High priority queues are serviced first • Absolute prioritization; ensures critical traffic of highest priority through use of the Frame Relay PVC Interface Priority Queueing feature
Configuration	No configuration required	Requires configuration	Requires configuration	Requires configuration

FIFO Queueing

In its simplest form, FIFO queueing--also known as first-come, first-served (FCFS) queueing--involves buffering and forwarding of packets in the order of arrival.

FIFO embodies no concept of priority or classes of traffic and consequently makes no decision about packet priority. There is only one queue, and all packets are treated equally. Packets are sent out an interface in the order in which they arrive.

When FIFO is used, ill-behaved sources can consume all the bandwidth, bursty sources can cause delays in time-sensitive or important traffic, and important traffic can be dropped because less important traffic fills the queue.

When no other queueing strategies are configured, all interfaces except serial interfaces at E1 (2.048 Mbps) and below use FIFO by default. (Serial interfaces at E1 and below use WFQ by default.)

FIFO, which is the fastest method of queueing, is effective for large links that have little delay and minimal congestion. If your link has very little congestion, FIFO queueing may be the only queueing you need to use.

Weighted Fair Queueing

This section discusses the four types of WFQ described in the following sections:

- Flow-Based Weighted Fair Queueing, on page 7
- Distributed Weighted Fair Queueing, on page 9
- Class-Based Weighted Fair Queueing, on page 11
- Distributed Class-Based Weighted Fair Queueing, on page 13

This section also discusses the six related features described in the following sections:

- IP RTP Priority, on page 15
- Frame Relay IP RTP Priority, on page 17
- Frame Relay PVC Interface Priority Queueing, on page 17
- Low Latency Queueing, on page 19
- Distributed Low Latency Queueing, on page 22
- Low Latency Queueing for Frame Relay, on page 25

The table below summarizes the differences among WFQ, DWFQ, CBWFQ, and DCBWFQ.

Table 2: WFQ, DWFQ, CBWFQ, and DCBWFQ Comparison

WFQ	DWFQ	CBWFQ	DCBWFQ
Flow-based WFQ: • Weighted, when packets are classified; for example, Resource Reservation Protocol (RSVP) • Fair queued (FQ), when packets are not classified (for example, best-effort traffic)	Flow-based DWFQ: • FQ, not weighted Class-based DWFQ: • Weighted • QoS-group-based • Type of Service (ToS)-based	Class-based WFQ: • Weighted • Bandwidth allocation can be specified for a specific class of traffic	Class-based distributed WFQ: • Weighted • Bandwidth allocation can be specified for a specific class of traffic
Runs on standard Cisco IOS platforms	Runs on Versatile Interface Processor (VIP) (faster performance)	Runs on standard Cisco IOS platforms	Runs on VIP (faster performance)

For DWFQ and DCBWFQ, all queueing is transacted by the VIP. On the VIP, all packets are sent directly out the interface. A Route Switch Processor (RSP) resides on the same platform as the VIP. The RSP handles all tasks associated with system maintenance and routing. The VIP and the RSP each handle some scheduling.

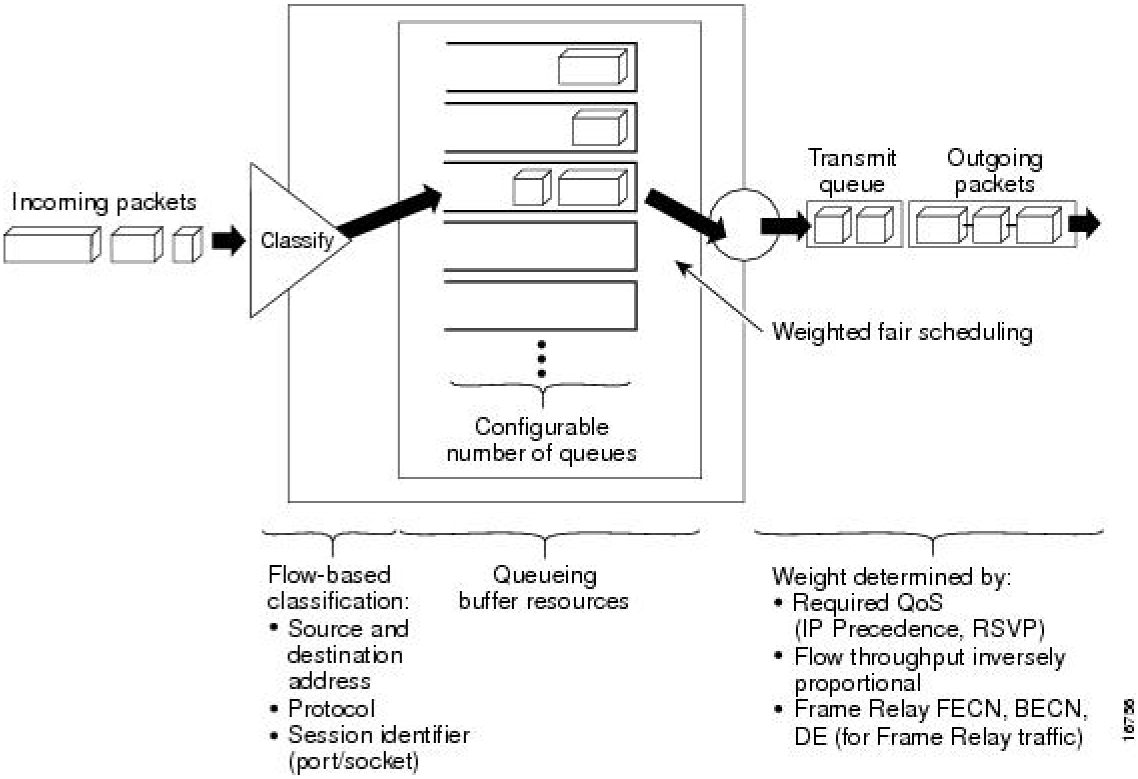
The dual-processor support accounts for the faster speed of DWFQ and DCBWFQ over WFQ running on standard Cisco IOS platforms.

For information on how to configure WFQ, DWFQ, CBWFQ, and DCBWFQ, see the "Configuring Weighted Fair Queueing" module. For information on how to configure per-VC WFQ and CBWFQ, see the "Configuring IP to ATM Class of Service" module.

Flow-Based Weighted Fair Queueing

WFQ is a dynamic scheduling method that provides fair bandwidth allocation to all network traffic. WFQ applies priority, or weights, to identified traffic to classify traffic into conversations and determine how much bandwidth each conversation is allowed relative to other conversations. WFQ is a flow-based algorithm that simultaneously schedules interactive traffic to the front of a queue to reduce response time and fairly shares the remaining bandwidth among high-bandwidth flows. In other words, WFQ allows you to give low-volume traffic, such as Telnet sessions, priority over high-volume traffic, such as FTP sessions. WFQ gives concurrent file transfers balanced use of link capacity; that is, when multiple file transfers occur, the transfers are given comparable bandwidth. The figure below shows how WFQ works.

Figure 1: Weighted Fair Queueing


Incoming packets
Classify
Transmit queue
Outgoing packets
Weighted fair scheduling
Configurable number of queues
Flow-based classification:
• Source and destination address
• Protocol
• Session identifier (port/socket)
Queueing buffer resources
Weight determined by:
• Required QoS (IP Precedence, RSVP)
• Flow throughput inversely proportional
• Frame Relay FECN, BECN, DE (for Frame Relay traffic)
16708

WFQ overcomes a serious limitation of FIFO queueing. When FIFO is in effect, traffic is sent in the order received without regard for bandwidth consumption or the associated delays. As a result, file transfers and other high-volume network applications often generate series of packets of associated data. These related packets are known as packet trains. Packet trains are groups of packets that tend to move together through the network. These packet trains can consume all available bandwidth, depriving other traffic of bandwidth.

WFQ provides traffic priority management that dynamically sorts traffic into messages that make up a conversation. WFQ breaks up the train of packets within a conversation to ensure that bandwidth is shared fairly between individual conversations and that low-volume traffic is transferred in a timely fashion.

WFQ classifies traffic into different flows based on packet header addressing, including such characteristics as source and destination network or MAC address, protocol, source and destination port and socket numbers of the session, Frame Relay data-link connection identifier (DLCI) value, and ToS value. There are two categories of flows: high-bandwidth sessions and low-bandwidth sessions. Low-bandwidth traffic has effective priority over high-bandwidth traffic, and high-bandwidth traffic shares the transmission service proportionally according to assigned weights. Low-bandwidth traffic streams, which comprise the majority of traffic, receive preferential service, allowing their entire offered loads to be sent in a timely fashion. High-volume traffic streams share the remaining capacity proportionally among themselves.

WFQ places packets of the various conversations in the fair queues before transmission. The order of removal from the fair queues is determined by the virtual time of the delivery of the last bit of each arriving packet.

New messages for high-bandwidth flows are discarded after the congestive-messages threshold has been met. However, low-bandwidth flows, which include control-message conversations, continue to enqueue data. As a result, the fair queue may occasionally contain more messages than are specified by the threshold number.

WFQ can manage duplex data streams, such as those between pairs of applications, and simplex data streams such as voice or video.

The WFQ algorithm also addresses the problem of round-trip delay variability. If multiple high-volume conversations are active, their transfer rates and interarrival periods are made much more predictable. WFQ greatly enhances algorithms such as Systems Network Architecture (SNA) Logical Link Control (LLC) and TCP congestion control and slow start features.

Flow-based WFQ is used as the default queueing mode on most serial interfaces configured to run at E1 speeds (2.048 Mbps) or below.

WFQ provides the solution for situations in which it is desirable to provide consistent response time to heavy and light network users alike without adding excessive bandwidth. WFQ automatically adapts to changing network traffic conditions.

Restrictions

WFQ is not supported with tunneling and encryption because these features modify the packet content information required by WFQ for classification.

Although WFQ automatically adapts to changing network traffic conditions, it does not offer the degree of precision control over bandwidth allocation that CQ and CBWFQ offer.

WFQ and IP Precedence

WFQ is IP precedence-aware. It can detect higher priority packets marked with precedence by the IP Forwarder and can schedule them faster, providing superior response time for this traffic. Thus, as the precedence increases, WFQ allocates more bandwidth to the conversation during periods of congestion.

WFQ assigns a weight to each flow, which determines the transmit order for queued packets. In this scheme, lower weights are served first. For standard Cisco IOS WFQ, the IP precedence serves as a divisor to this weighting factor.

Like CQ, WFQ sends a certain number of bytes from each queue. With WFQ, each queue corresponds to a different flow. For each cycle through all flows, WFQ effectively sends a number of bytes equal to the precedence of the flow plus one. This number is only used as a ratio to determine how many bytes per packets to send. However, for the purposes of understanding WFQ, using this number as the byte count is sufficient. For instance, traffic with an IP Precedence value of 7 gets a lower weight than traffic with an IP Precedence value of 3, thus, the priority in transmit order. The weights are inversely proportional to the IP Precedence value.

To determine the bandwidth allocation for each queue, divide the byte count for the flow by the total byte count for all flows. For example, if you have one flow at each precedence level, each flow will get precedence + 1 parts of the link:

1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 = 36

Thus, precedence 0 traffic will get 1/36 of the bandwidth, precedence 1 traffic will get 2/36, and precedence 7 traffic will get 8/36.

However, if you have 18 precedence 1 flows and one of each of the rest, the total is now:

1 + 2(18) + 3 + 4 + 5 + 6 + 7 + 8 = 70

Precedence 0 traffic will get 1/70, each of the precedence 1 flows will get 2/70, and so on.

As flows are added or ended, the actual allocated bandwidth will continuously change.

WFQ and RSVP

RSVP uses WFQ to allocate buffer space and schedule packets, and to guarantee bandwidth for reserved flows. WFQ works with RSVP to help provide differentiated and guaranteed QoS services.

RSVP is the Internet Engineering Task Force (IETF) Internet Standard (RFC 2205) protocol for allowing an application to dynamically reserve network bandwidth. RSVP enables applications to request a specific QoS for a data flow. The Cisco implementation allows RSVP to be initiated within the network using configured proxy RSVP.

RSVP is the only standard signalling protocol designed to guarantee network bandwidth from end to end for IP networks. Hosts and routers use RSVP to deliver QoS requests to the routers along the paths of the data stream and to maintain router and host state to provide the requested service, usually bandwidth and latency. RSVP uses a mean data rate, the largest amount of data the router will keep in queue, and minimum QoS to determine bandwidth reservation.

WFQ or Weighted Random Early Detection (WRED) acts as the preparer for RSVP, setting up the packet classification and scheduling required for the reserved flows. Using WFQ, RSVP can deliver an Integrated Services Guaranteed Service.

Distributed Weighted Fair Queueing

DWFQ is a special high-speed version of WFQ that runs on the VIP. It is supported on the following routers with a VIP2-40 or greater interface processor:

- Cisco 7000 series with RSP7000
- Cisco 7500 series

A VIP2-50 interface processor is recommended when the aggregate line rate of the port adapters on the VIP is greater than DS3. A VIP2-50 card is required for OC-3 rates.

To use DWFQ, distributed Cisco Express Forwarding (dCEF) switching must be enabled on the interface.

Note The VIP-distributed WFQ implementation differs from WFQ that runs on all other platforms.

There are two forms of distributed WFQ:

- Flow-based. In this form, packets are classified by flow. Packets with the same source IP address, destination IP address, source TCP or User Datagram Protocol (UDP) port, destination TCP or UDP port, protocol, and ToS field belong to the same flow. (All non-IP packets are treated as flow 0.)

Each flow corresponds to a separate output queue. When a packet is assigned to a flow, it is placed in the queue for that flow. During periods of congestion, DWFQ allocates an equal share of the bandwidth to each active queue.

Flow-based DWFQ is also called fair queueing because all flows are equally weighted and allocated equal bandwidth. In the current implementation of DWFQ, weights are not assigned to flows. With DWFQ, well-behaved hosts are protected from ill-behaved hosts.

- Class-based. In this form, packets are assigned to different queues based on their QoS group or the IP precedence in the ToS field.

QoS groups allow you to customize your QoS policy. A QoS group is an internal classification of packets used by the router to determine how packets are treated by certain QoS features, such as DWFQ and committed access rate (CAR). Use a CAR policy or the QoS Policy Propagation via Border Gateway Protocol (BGP) feature to assign packets to QoS groups.

If you want to classify packets based only on the two low-order IP Precedence bits, use ToS-based DWFQ. Specify a weight for each class. In periods of congestion, each group is allocated a percentage of the output bandwidth equal to the weight of the class. For example, if a class is assigned a weight of 50, packets from this class will be allocated at least 50 percent of the outgoing bandwidth during periods of congestion. When the interface is not congested, queues can use any available bandwidth.

The "Drop Policy" section describes the drop policy used by both forms.

Drop Policy

DWFQ keeps track of the number of packets in each queue and the total number of packets in all queues.

When the total number of packets is below the aggregate limit, queues can buffer more packets than the individual queue limit.

When the total number of packets reaches the aggregate limit, the interface starts enforcing the individual queue limits. Any new packets that arrive for a queue that has exceeded its individual queue limit are dropped. Packets that are already in the queue will not be dropped, even if the queue is over the individual limit.

In some cases, the total number of packets in all queues put together may exceed the aggregate limit.

Restrictions

Use DWFQ with IP traffic. All non-IP traffic is treated as a single flow and, therefore, placed in the same queue.

DWFQ has the following restrictions:

- Can be configured on interfaces, but not subinterfaces.

- Is not supported with the ATM encapsulations AAL5-MUX and AAL5-NLPID.
- Is not supported on Fast EtherChannel, tunnel interfaces, or other logical (virtual) interfaces such as Multilink PPP (MLP).
- Cannot be configured on the same interface as RSP-based PQ, CQ, or WFQ.

Class-Based Weighted Fair Queueing

CBWFQ extends the standard WFQ functionality to provide support for user-defined traffic classes. For CBWFQ, you define traffic classes based on match criteria including protocols, access control lists (ACLs), and input interfaces. Packets satisfying the match criteria for a class constitute the traffic for that class. A FIFO queue is reserved for each class, and traffic belonging to a class is directed to the queue for that class.

Once a class has been defined according to its match criteria, you can assign it characteristics. To characterize a class, you assign it bandwidth, weight, and maximum packet limit. The bandwidth assigned to a class is the guaranteed bandwidth delivered to the class during congestion.

To characterize a class, you also specify the queue limit for that class, which is the maximum number of packets allowed to accumulate in the queue for the class. Packets belonging to a class are subject to the bandwidth and queue limits that characterize the class.

After a queue has reached its configured queue limit, enqueueing of additional packets to the class causes tail drop or packet drop to take effect, depending on how class policy is configured.

Tail drop is used for CBWFQ classes unless you explicitly configure policy for a class to use WRED to drop packets as a means of avoiding congestion. Note that if you use WRED packet drop instead of tail drop for one or more classes comprising a policy map, you must ensure that WRED is not configured for the interface to which you attach that service policy.

If a default class is configured with the **bandwidth** policy-map class configuration command, all unclassified traffic is put into a single FIFO queue and given treatment according to the configured bandwidth. If a default class is configured with the **fair-queue** command, all unclassified traffic is flow classified and given best-effort treatment. If no default class is configured, then by default the traffic that does not match any of the configured classes is flow classified and given best-effort treatment. Once a packet is classified, all of the standard mechanisms that can be used to differentiate service among the classes apply.

Flow classification is standard WFQ treatment. That is, packets with the same source IP address, destination IP address, source TCP or UDP port, or destination TCP or UDP port are classified as belonging to the same flow. WFQ allocates an equal share of bandwidth to each flow. Flow-based WFQ is also called fair queueing because all flows are equally weighted.

For CBWFQ, the weight specified for the class becomes the weight of each packet that meets the match criteria of the class. Packets that arrive at the output interface are classified according to the match criteria filters you define, then each one is assigned the appropriate weight. The weight for a packet belonging to a specific class is derived from the bandwidth you assigned to the class when you configured it; in this sense the weight for a class is user-configurable.

After the weight for a packet is assigned, the packet is enqueued in the appropriate class queue. CBWFQ uses the weights assigned to the queued packets to ensure that the class queue is serviced fairly.

Configuring a class policy--thus, configuring CBWFQ--entails these three processes:

- Defining traffic classes to specify the classification policy (class maps).

This process determines how many types of packets are to be differentiated from one another.

- Associating policies--that is, class characteristics--with each traffic class (policy maps).

This process entails configuration of policies to be applied to packets belonging to one of the classes previously defined through a class map. For this process, you configure a policy map that specifies the policy for each traffic class.

- Attaching policies to interfaces (service policies).

This process requires that you associate an existing policy map, or service policy, with an interface to apply the particular set of policies for the map to that interface.

CBWFQ Bandwidth Allocation

The sum of all bandwidth allocation on an interface cannot exceed 75 percent of the total available interface bandwidth. The remaining 25 percent is used for other overhead, including Layer 2 overhead, routing traffic, and best-effort traffic. Bandwidth for the CBWFQ class-default class, for instance, is taken from the remaining 25 percent. However, under aggressive circumstances in which you want to configure more than 75 percent of the interface bandwidth to classes, you can override the 75 percent maximum sum allocated to all classes or flows. If you want to override the default 75 percent, exercise caution and ensure that you allow enough remaining bandwidth to support best-effort and control traffic, and Layer 2 overhead.

When ATM is used you must account for the fact that ATM cell tax overhead is not included. For example, consider the case where a class needs guaranteed bandwidth on an ATM permanent virtual circuit (PVC). Suppose the average packet size for the class is 256 bytes and the class needs 100 kbps (which translates to 49 packets per second) of guaranteed bandwidth. Each 256-byte packet would be split into six cells to be sent on a VC, giving a total of 6 * 53 = 318 bytes. In this case, the ATM cell tax overhead would be 62 bytes or 49 * 62 * 8 = 24.34 kbps. When configuring CBWFQ in this example, ensure that the sum of all the configured class bandwidths is less than the VC bandwidth by at least 24.34 kbps to ensure desired payload guarantee for the configured classes (in this example, there is only one class). If you have several classes, the sum of all the class overheads should be estimated and added to the sum of all the configured class bandwidths. This total should be less than the VC bandwidth to ensure the required payload guarantees.

Why Use CBWFQ

Here are some general factors you should consider in determining whether you need to configure CBWFQ:

- Bandwidth allocation. CBWFQ allows you to specify the exact amount of bandwidth to be allocated for a specific class of traffic. Taking into account available bandwidth on the interface, you can configure up to 64 classes and control distribution among them, which is not the case with flow-based WFQ. Flow-based WFQ applies weights to traffic to classify it into conversations and determine how much bandwidth each conversation is allowed relative to other conversations. For flow-based WFQ, these weights, and traffic classification, are dependent on and limited to the seven IP Precedence levels.
- Coarser granularity and scalability. CBWFQ allows you to define what constitutes a class based on criteria that exceed the confines of flow. CBWFQ allows you to use ACLs and protocols or input interface names to define how traffic will be classified, thereby providing coarser granularity. You need not maintain traffic classification on a flow basis. Moreover, you can configure up to 64 discrete classes in a service policy.

CBWFQ and RSVP

RSVP can be used in conjunction with CBWFQ. When both RSVP and CBWFQ are configured for an interface, RSVP and CBWFQ act independently, exhibiting the same behavior that they would if each were running alone. RSVP continues to work as it does when CBWFQ is not present, even in regard to bandwidth availability assessment and allocation.

Restrictions

Configuring CBWFQ on a physical interface is only possible if the interface is in the default queueing mode. Serial interfaces at E1 (2.048 Mbps) and below use WFQ by default--other interfaces use FIFO by default. Enabling CBWFQ on a physical interface overrides the default interface queueing method. Enabling CBWFQ on an ATM PVC does not override the default queueing method.

If you configure a class in a policy map to use WRED for packet drop instead of tail drop, you must ensure that WRED is not configured on the interface to which you intend to attach that service policy.

Traffic shaping and policing are not currently supported with CBWFQ.

CBWFQ is supported on variable bit rate (VBR) and available bit rate (ABR) ATM connections. It is not supported on unspecified bit rate (UBR) connections.

CBWFQ is not supported on Ethernet subinterfaces.

Distributed Class-Based Weighted Fair Queueing

As explained earlier, WFQ offers dynamic, fair queueing that divides bandwidth across queues of traffic based on weights. WFQ ensures that all traffic is treated fairly, given its weight. For more information about WFQ, see the Weighted Fair Queueing, on page 6 section of this module.

The DCBWFQ feature extends the standard WFQ functionality to provide support for user-defined traffic classes on the VIP. These user-defined traffic classes are configured in the Modular Quality of Service Command-Line Interface (Modular QoS CLI) feature. For information on how to configure QoS with the Modular QoS CLI, see the "Applying QoS Features Using the MQC" module.

The maximum number of packets allowed to accumulate in a traffic class queue is called the queue limit and is specified with the **queue-limit** command when you create a service policy with the **policy-map** command. Packets belonging to a traffic class are subject to the guaranteed bandwidth allocation and the queue limits that characterize the traffic class.

After a queue has reached its configured queue limit, enqueuing of additional packets to the traffic class causes tail drop or WRED drop to take effect, depending on how the service policy 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 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 packet drop 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.

For information on how to configure DCBWFQ, see the "Configuring Weighted Fair Queueing" module.

RSVP Interaction with DCBWFQ

When RSVP and DCBWFQ are configured, RSVP and DCBWFQ act independently of one another. RSVP and DCBWFQ allocate bandwidth among their traffic classes and flows according to unallocated bandwidth available at the underlying point of congestion.

When an RSVP flow is created, the VIP queueing system reserves the unit of bandwidth allocation in an RSVP queue, similar to the way a traffic class queue is allotted to a DCBWFQ traffic class. DCBWFQ traffic classes are unaffected by the RSVP flows.

Benefits

Bandwidth Allocation

DCBWFQ allows you to specify the amount of guaranteed bandwidth to be allocated for a traffic class. Taking into account available bandwidth on the interface, you can configure up to 64 traffic classes and control bandwidth allocation among them. If excess bandwidth is available, the excess bandwidth is divided among the traffic classes in proportion to their configured bandwidths.

Flow-based WFQ allocates bandwidth equally among all flows.

Coarser Granularity and Scalability

DCBWFQ allows you to define what constitutes a traffic class based on criteria that exceed the confines of flow. DCBWFQ allows you to use ACLs and protocols or input interface names to define how traffic is classified, thereby providing coarser granularity. You need not maintain traffic classification on a flow basis. Moreover, you can configure up to 64 discrete traffic classes in a service policy.

Restrictions

Using the bandwidth Command on VIP Default Traffic Class

On a VIP, all traffic that does not match a user-defined traffic class is classified as part of the default traffic class. The implicit bandwidth allocated to the default traffic class on a VIP is equal to the link bandwidth minus all of the user-defined bandwidth given to the user-defined traffic classes (with the **bandwidth** command). At least 1 percent of the link bandwidth is always reserved for the default traffic class.

Because the bandwidth of the default traffic class for a VIP is implicit (the default traffic class receives all remaining bandwidth not given to the user-defined traffic classes), the **bandwidth** command cannot be used with the default traffic class when you configure a VIP.

Using the match protocol Command on a VIP

Do not use the **match protocol**command to create a traffic class with a non-IP protocol as a match criterion. The VIP does not support matching of non-IP protocols.

PA-A3-8T1IMA Modules

DCBWFQ is not supported on Cisco 7500 series routers with PA-A3-8T1IMA modules.

Prerequisites

WFQ

Attaching a service policy to an interface disables WFQ on that interface if WFQ is configured for the interface. For this reason, you should ensure that WFQ is not enabled on such an interface.

For information on WFQ, see the "Configuring Weighted Fair Queueing" module.

ACLs

You can specify a numbered access list as the match criterion for any traffic class that you create. For this reason, you should know how to configure access lists.

Modular QoS CLI

You can configure DCBWFQ using the Modular QoS CLI.

For information on configuring QoS features with the Modular QoS CLI, see the "Applying QoS Features Using the MQC" module.

IP RTP Priority

The IP RTP Priority feature provides a strict priority queueing scheme for delay-sensitive data such as voice. Voice traffic can be identified by its Real-Time Transport Protocol (RTP) port numbers and classified into a priority queue configured by the **ip rtp priority** command. The result is that voice is serviced as strict priority in preference to other nonvoice traffic.

Note Although this section focuses mainly on voice traffic, IP RTP Priority is useful for any RTP traffic.

The IP RTP Priority feature extends and improves on the functionality offered by the **ip rtp reserve** command by allowing you to specify a range of UDP/RTP ports whose traffic is guaranteed strict priority service over any other queues or classes using the same output interface. Strict priority means that if packets exist in the priority queue, they are dequeued and before packets in other queues are dequeued. We recommend that you use the **ip rtp priority** command instead of the **ip rtp reserve** command for voice configurations.

The IP RTP Priority feature does not require that you know the port of a voice call. Rather, the feature gives you the ability to identify a range of ports whose traffic is put into the priority queue. Moreover, you can specify the entire voice port range--16384 to 32767--to ensure that all voice traffic is given strict priority service. IP RTP Priority is especially useful on links whose speed is less than 1.544 Mbps.

This feature can be used in conjunction with either WFQ or CBWFQ on the same outgoing interface. In either case, traffic matching the range of ports specified for the priority queue is guaranteed strict priority over other CBWFQ classes or WFQ flows; packets in the priority queue are always serviced first. Note the following conditions of the **ip rtp priority** command:

- When used in conjunction with WFQ, the **ip rtp priority** command provides strict priority to voice, and WFQ scheduling is applied to the remaining queues.
- When used in conjunction with CBWFQ, the **ip rtp priority** command provides strict priority to voice. CBWFQ can be used to set up classes for other types of traffic (such as SNA) that needs dedicated bandwidth and needs to be treated better than best effort and not as strict priority; the nonvoice traffic is serviced fairly based on the weights assigned to the enqueued packets. CBWFQ can also support flow-based WFQ within the default CBWFQ class if so configured.

Because voice packets are small in size and the interface also can have large packets going out, the Link Fragmentation and Interleaving (LFI) feature should also be configured on lower speed interfaces. When you enable LFI, the large data packets are broken up so that the small voice packets can be interleaved between the data fragments that make up a large data packet. LFI prevents a voice packet from needing to wait until a large packet is sent. Instead, the voice packet can be sent in a shorter amount of time.

For information on how to configure IP RTP Priority, see the "Configuring Weighted Fair Queueing" module.

IP RTP Priority Bandwidth Allocation

If you want to understand its behavior and properly use the IP RTP Priority feature, it is important to consider its admission control and policing characteristics. When you use the **ip rtp priority** command to configure the priority queue for voice, you specify a strict bandwidth limitation. This amount of bandwidth is guaranteed to voice traffic enqueued in the priority queue. (This is the case whether you use the IP RTP Priority feature with CBWFQ or WFQ.)

Note IP RTP Priority does not have per-call admission control. The admission control is on an aggregate basis. For example, if configured for 96 kbps, IP RTP Priority guarantees that 96 kbps is available for reservation. It does not ensure that only four calls of 24 kbps are admitted. A fifth call of 24 kbps could be admitted, but because the five calls will only get 96 kbps, the call quality will be deteriorated. (Each call would get 96/5 = 19.2 kbps.) In this example, it is the responsibility of the user to ensure that only four calls are placed at one time.

IP RTP Priority closely polices use of bandwidth for the priority queue, ensuring that the allocated amount is not exceeded in the event of congestion. In fact, IP RTP Priority polices the flow every second. IP RTP Priority prohibits transmission of additional packets once the allocated bandwidth is consumed. If it discovers that the configured amount of bandwidth is exceeded, IP RTP Priority drops packets, an event that is poorly tolerated by voice traffic. (Enable debugging to watch for this condition.) Close policing allows for fair treatment of other data packets enqueued in other CBWFQ or WFQ queues. To avoid packet drop, be certain to allocate to the priority queue the most optimum amount of bandwidth, taking into consideration the type of codec used and interface characteristics. IP RTP Priority will not allow traffic beyond the allocated amount.

It is always safest to allocate to the priority queue slightly more than the known required amount of bandwidth. For example, suppose you allocated 24 kbps bandwidth, the standard amount required for voice transmission, to the priority queue. This allocation seems safe because transmission of voice packets occurs at a constant bit rate. However, because the network and the router or switch can use some of the bandwidth and introduce jitter and delay, allocating slightly more than the required amount of bandwidth (such as 25 kbps) ensures constancy and availability.

The IP RTP Priority admission control policy takes RTP header compression into account. Therefore, while configuring the *bandwidth* parameter of the **ip rtp priority** command you only need to configure for the bandwidth of the compressed call. For example, if a G.729 voice call requires 24 kbps uncompressed bandwidth (not including Layer 2 payload) but only 12 kbps compressed bandwidth, you only need to configure a bandwidth of 12 kbps. You need to allocate enough bandwidth for all calls if there will be more than one call.

The sum of all bandwidth allocation for voice and data flows on the interface cannot exceed 75 percent of the total available bandwidth. Bandwidth allocation for voice packets takes into account the payload plus the IP, RTP, and UDP headers, but again, not the Layer 2 header. Allowing 25 percent bandwidth for other overhead is conservative and safe. On a PPP link, for instance, overhead for Layer 2 headers assumes 4 kbps.

If you know how much bandwidth is required for additional overhead on a link, under aggressive circumstances in which you want to give voice traffic as much bandwidth as possible, you can override the 75 percent maximum allocation for the bandwidth sum allocated to all classes or flows. If you want to override the fixed amount of bandwidth, exercise caution and ensure that you allow enough remaining bandwidth to support best-effort and control traffic, and Layer 2 overhead.

As another alternative, if the importance of voice traffic far exceeds that of data, you can allocate most of the 75 percent bandwidth used for flows and classes to the voice priority queue. Unused bandwidth at any given point will be made available to the other flows or classes.

Restrictions

Because the **ip rtp priority** command gives absolute priority over other traffic, it should be used with care. In the event of congestion, if the traffic exceeds the configured bandwidth, then all the excess traffic is dropped.

The **ip rtp reserve** and **ip rtp priority** commands cannot be configured on the same interface.

Frame Relay IP RTP Priority

The Frame Relay IP RTP Priority feature provides a strict priority queueing scheme on a Frame Relay PVC for delay-sensitive data such as voice. Voice traffic can be identified by its RTP port numbers and classified into a priority queue configured by the **frame-relay ip rtp priority**command. The result of using this feature is that voice is serviced as strict priority in preference to other nonvoice traffic.

This feature extends the functionality offered by the **ip rtp priority** command by supporting Frame Relay PVCs. This feature allows you to specify a range of UDP ports whose voice traffic is guaranteed strict priority service over any other queues or classes using the same output interface. Strict priority means that if packets exist in the priority queue, they are dequeued and sent before packets in other queues are dequeued. This process is performed on a per-PVC basis, rather than at the interface level.

For information on how to configure Frame Relay IP RTP Priority, see the "Configuring Weighted Fair Queueing" module.

Frame Relay PVC Interface Priority Queueing

The Frame Relay PVC Interface Priority Queueing (PIPQ) feature provides an interface-level priority queueing scheme in which prioritization is based on destination PVC rather than packet contents. For example, Frame Relay (FR) PIPQ allows you to configure a PVC transporting voice traffic to have absolute priority over a PVC transporting signalling traffic, and a PVC transporting signalling traffic to have absolute priority over a PVC transporting data.

For information on how to configure Frame Relay PIPQ, see the "Configuring Weighted Fair Queueing" module. For information about Frame Relay, see the "Configuring Frame Relay" module.

Frame Relay PIPQ provides four levels of priority: high, medium, normal, and low. The Frame Relay packet is examined at the interface for the data-link connection identifier (DLCI) value. The packet is then sent to the correct priority queue based on the priority level configured for that DLCI.

Note When using Frame Relay PIPQ, configure the network so that different types of traffic are transported on separate PVCs. Frame Relay PIPQ is not meant to be used when an individual PVC carries different traffic types that have different QoS requirements.

You assign priority to a PVC within a Frame Relay map class. All PVCs using or inheriting that map class will be classed according to the configured priority. If a PVC does not have a map class associated with it, or if the map class associated with it does not have priority explicitly configured, then the packets on that PVC will be queued on the default "normal" priority queue.

If you do not enable Frame Relay PIPQ on the interface using the **frame-relay interface-queue priority**command in interface configuration mode, configuring PVC priority within a map class will not be effective. At this time you have the option to also set the size (in maximum number of packets) of the four priority queues.

Frame Relay PIPQ works with or without Frame Relay Traffic Shaping (FRTS) and FRF.12 (or higher). The interface-level priority queueing takes the place of the FIFO queueing or dual FIFO queueing normally used by FRTS and FRF.12 (or higher). PVC priority assigned within FR PIPQ takes precedence over FRF.12 priority, which means that all packets destined for the same PVC will be queued on the same interface queue whether they were fragmented or not.

Note Although high priority PVCs most likely will transport only small packets of voice traffic, you may want to configure FRF.12 (or higher) on these PVCs anyway to guard against any unexpectedly large packets.

Restrictions

The following restrictions apply to Frame Relay PIPQ:

- It is not supported on loopback or tunnel interfaces, or interfaces that explicitly disallow priority queueing.
- It is not supported with hardware compression.
- It cannot be enabled on an interface that is already configured with queueing other than FIFO queueing. FR PIPQ can be enabled if WFQ is configured, as long as WFQ is the default interface queueing method.

Prerequisites

The following prerequisites apply to Frame Relay PIPQ:

- PVCs should be configured to carry a single type of traffic.
- The network should be configured with adequate call admission control to prevent starvation of any of the priority queues.

Low Latency Queueing

The LLQ feature brings strict PQ to CBWFQ. Strict PQ allows delay-sensitive data such as voice to be dequeued and sent before packets in other queues are dequeued.

Without LLQ, CBWFQ provides WFQ based on defined classes with no strict priority queue available for real-time traffic. CBWFQ allows you to define traffic classes and then assign characteristics to that class. For example, you can designate the minimum bandwidth delivered to the class during congestion.

For CBWFQ, the weight for a packet belonging to a specific class is derived from the bandwidth you assigned to the class when you configured it. Therefore, the bandwidth assigned to the packets of a class determines the order in which packets are sent. All packets are serviced fairly based on weight; no class of packets may be granted strict priority. This scheme poses problems for voice traffic that is largely intolerant of delay, especially variation in delay. For voice traffic, variations in delay introduce irregularities of transmission manifesting as jitter in the heard conversation.

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

One of the ways in which the strict PQ used within CBWFQ differs from its use outside CBWFQ is in the parameters it takes. Outside CBWFQ, you can use the **ip rtp priority** command to specify the range of UDP ports whose voice traffic flows are to be given priority service. Using the **priority** command, you are no longer limited to a UDP port number to stipulate priority flows because you can configure the priority status for a class within CBWFQ. Instead, all of the valid match criteria used to specify traffic for a class now apply to priority traffic. These methods of specifying traffic for a class include matching on access lists, protocols, and input interfaces. Moreover, within an access list you can specify that traffic matches are allowed based on the IP differentiated services code point (DSCP) value that is set using the first six bits of the ToS byte in the IP header.

Although it is possible to enqueue various types of real-time traffic to the strict priority queue, we strongly recommend that you direct only voice traffic to it because voice traffic is well-behaved, whereas other types of real-time traffic are not. Moreover, voice traffic requires that delay be nonvariable in order to avoid jitter. Real-time traffic such as video could introduce variation in delay, thereby thwarting the steadiness of delay required for successful voice traffic transmission.

For information on how to configure LLQ, see the "Configuring Weighted Fair Queueing" module.

LLQ Bandwidth Allocation

When you specify the **priority** command for a class, it takes a *bandwidth* argument that gives maximum bandwidth in kbps. You use this parameter to specify the maximum amount of bandwidth allocated for packets belonging to the class configured with the **priority** command. The bandwidth parameter both guarantees bandwidth to the priority class and restrains the flow of packets from the priority class.

In the event of congestion, policing is used to drop packets when the bandwidth is exceeded. Voice traffic enqueued to the priority queue is UDP-based and therefore not adaptive to the early packet drop characteristic of WRED. Because WRED is ineffective, you cannot use the WRED **random-detect** command with the

priority command. In addition, because policing is used to drop packets and a queue limit is not imposed, the **queue-limit** command cannot be used with the **priority** command.

When congestion occurs, traffic destined for the priority queue is metered to ensure that the bandwidth allocation configured for the class to which the traffic belongs is not exceeded.

Priority traffic metering has the following qualities:

- It is much like the rate-limiting feature of CAR, except that priority traffic metering is only performed under congestion conditions. When the device is not congested, the priority class traffic is allowed to exceed its allocated bandwidth. When the device is congested, the priority class traffic above the allocated bandwidth is discarded.
- It is performed on a per-packet basis, and tokens are replenished as packets are sent. If not enough tokens are available to send the packet, it is dropped.
- It restrains priority traffic to its allocated bandwidth to ensure that nonpriority traffic, such as routing packets and other data, is not starved.

With metering, the classes are policed and rate-limited individually. That is, although a single policy map might contain four priority classes, all of which are enqueued in a single priority queue, they are each treated as separate flows with separate bandwidth allocations and constraints.

It is important to note that because bandwidth for the priority class is specified as a parameter to the **priority** command, you cannot also configure the **bandwidth** policy-map class configuration command for a priority class. To do so is a configuration violation that would only introduce confusion in relation to the amount of bandwidth to allocate.

The bandwidth allocated for a priority queue always includes the Layer 2 encapsulation header. However, it does not include other headers, such as ATM cell tax overheads. When you calculate the amount of bandwidth to allocate for a given priority class, you must account for the fact that Layer 2 headers are included. When ATM is used, you must account for the fact that ATM cell tax overhead is not included. You must also allow bandwidth for the possibility of jitter introduced by routers in the voice path.

Consider this case that uses ATM. Suppose a voice stream of 60 bytes emitting 50 packets per second is encoded using G.729. Prior to converting the voice stream to cells, the meter for the priority queue used for the voice stream assesses the length of the packet after the Layer 2 Logical Link Control (LLC) headers have been added.

Given the 8-byte Layer 2 LLC header, the meter will take into account a 68-byte packet. Because ATM cells are a standard 53 bytes long, before the 68-byte packet is emitted on the line, it is divided into two 53-byte ATM cells. Thus, the bandwidth consumed by this flow is 106 bytes per packet.

For this case, then, you must configure the bandwidth to be at least 27.2 kbps (68 * 50 * 8 = 27.2 kbps). However, recall that you must also allow for the ATM cell tax overhead, which is not accounted for by the configured bandwidth. In other words, the sum of the bandwidths for all classes must be less than the interface bandwidth by at least 15.2 kbps ([106 - 68] * 50 * 8 = 15.2 kbps). You should also remember to allow bandwidth for router-introduced jitter.

LLQ with IP RTP Priority

LLQ and IP RTP Priority can be configured at the same time, but IP RTP Priority takes precedence. To demonstrate how they work together, consider the following configuration:

```
policy-map llqpolicy
 class voice
 priority 50
```

```
ip rtp priority 16384 20000 40
service-policy output llqpolicy
```

In this example, packets that match the 16384 to 20000 port range will be given priority with 40 kbps bandwidth; packets that match the voice class will be given priority with 50 kbps bandwidth. In the event of congestion, packets that match the 16384 to 20000 port range will receive no more than 40 kbps of bandwidth, and packets that match the voice class will receive no more than 50 kbps of bandwidth.

If packets match both criteria (ports 16384 to 20000 and class voice), IP RTP Priority takes precedence. In this example, the packets will be considered to match the 16384 to 20000 port range and will be accounted for in the 40 kbps bandwidth.

LLQ and Committed Burst Size

The functionality of LLQ has been extended to allow you to specify the Committed Burst (Bc) size in LLQ. This functionality is provided with the Configuring Burst Size in Low Latency Queueing feature. With this new functionality, the network can now accommodate temporary bursts of traffic and handle network traffic more efficiently.

Note The default Bc size used by LLQ is intended to handle voice-like non-bursty traffic. If you want to configure LLQ to handle the traffic of non-voice applications, you may need to increase the burst size accordingly, based on the application in use on your network.

LLQ and per-VC Hold Queue Support for ATM Adapters

By default, the queueing mechanism in use determines the size of the hold queue, and, therefore, the number of packets contained in the queue. The Configurable per-VC Hold Queue Support for ATM Adapters feature allows you to expand the default hold queue size and change (or vary) the number of packets the queue can contain. With this new feature, the hold queue can contain a maximum of 1024 packets.

This feature allows you to specify the number of packets contained in the hold queue, per VC, on ATM adapters that support per-VC queueing.

Note This feature is supported only on the Cisco 7200 series routers, and on Cisco 2600 and 3600 series adapters that support per-VC queueing.

For related information about per-VC and ATM configurations, see the "IP to ATM Class of Service Overview" module and the "Configuring IP to ATM Class of Service" module.

Why Use LLQ

Here are some general factors you should consider in determining whether you need to configure LLQ:

- LLQ provides strict priority service on ATM VCs and serial interfaces. (The IP RTP Priority feature allows priority queueing only on interfaces.)
- LLQ is not limited to UDP port numbers. Because you can configure the priority status for a class within CBWFQ, you are no longer limited to UDP port numbers to stipulate priority flows. Instead, all of the valid match criteria used to specify traffic for a class now apply to priority traffic.

- By configuring the maximum amount of bandwidth allocated for packets belonging to a class, you can avoid starving nonpriority traffic.

Restrictions

The following restrictions apply to LLQ:

- If you use access lists to configure matching port numbers, this feature provides priority matching for all port numbers, both odd and even. Because voice typically exists on even port numbers, and control packets are generated on odd port numbers, control packets are also given priority when using this feature. On very slow links, giving priority to both voice and control packets may produce degraded voice quality. Therefore, if you are only assigning priority based on port numbers, you should use the **ip rtp priority** command instead of the **priority** command. (The **ip rtp priority** command provides priority only for even port numbers.)
- The **random-detect** command, **queue-limit** command,and **bandwidth** policy-map class configuration command cannot be used while the **priority** command is configured.
- The **priority** command can be configured in multiple classes, but it should only be used for voice-like, constant bit rate (CBR) traffic.

Distributed Low Latency Queueing

The Distributed LLQ feature provides the ability to specify low latency behavior for a traffic class on a VIP-based Cisco 7500 series router except one with a PA-A3-8T1IMA module. LLQ allows delay-sensitive data such as voice to be dequeued and sent before packets in other queues are dequeued.

The Distributed LLQ feature also introduces the ability to limit the depth of a device transmission ring. Before the introduction of Distributed LLQ, the maximum transmission ring depth was not a user-configurable parameter. Therefore, particles could accumulate on a transmission ring without limitation, which could result in unavoidable high latencies. The Distributed LLQ feature allows users to limit the number of particles that may exist on a transmission ring, effectively lowering the latency incurred by packets sitting on that transmission ring.

The **priority** command is used to allow delay-sensitive data to be dequeued and sent first. LLQ enables use of a single priority queue within which individual classes of traffic can be placed. To enqueue class traffic to the priority queue, you configure the **priority** command for the class after you specify the named class within a policy map. The amount of bandwidth available for the priority queue can be specified either as a set amount of bandwidth in kbps or as a percentage of all available bandwidth (beginning in Cisco IOS Release 12.1(5)T).

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

The **tx-ring-limit** command allows the user to specify the number of allowable particles on a transmission ring, effectively lowering the latency for that transmission ring. One packet can contain multiple particles, and a typical particle is 512 bytes in size (the size depends on the interface types. For some interface types, a typical particle size is 256 bytes.) These particles can no longer accumulate on a transmission ring and cause unavoidable high latencies.

Distributed LLQ is supported on the Cisco 7500 RSP series router with a VIP except when a PA-A3-8T 1IMA module is configured.

This feature also supports the *Class-Based Quality of Service* MIB.

For information on how to configure Distributed LLQ, see the "Configuring Weighted Fair Queueing" module.

Guaranteeing Bandwidth with the priority Command

One method of using the **priority** command for a traffic class is to specify a *bandwidth* argument that gives the maximum bandwidth in kpbs. The other method of using the **priority** command for a traffic class, which was introduced in Cisco IOS Release 12.1(5)T, is to specify a percentage of available bandwidth to be reserved for the priority queue. The *bandwidth* value or percentage guarantees the configured bandwidth to the priority class under worst-case congestion scenarios. If excess bandwidth is available, the priority class will be allowed to utilize the bandwidth. If no excess bandwidth is available, the priority traffic will be constrained to the configured rate via packet drops. Each individual class that is configured to a bandwidth value will have its traffic constrained to its individual rate. When a class is constrained to its individual rate, the traffic is permitted a certain amount of burstiness because of the token bucket mechanism policing the stream. This amount of burstiness is controlled by the optional *burst* parameter in the **priority**command (this burstiness cannot be specified when specifying a priority queue based on a percentage of available bandwidth). The *burst* parameter specifies, in bytes, the amount of traffic allowed to pass through the token bucket as a one-time burst in excess of the token bucket drop parameters. The default burst value is 200 milliseconds of traffic at the configured token bucket drop parameters.

It is important to note that because bandwidth for the priority class is specified as a parameter to the **priority** command, you cannot also configure the **bandwidth** command for a priority class. To do so is a configuration violation that introduces confusion in relation to the amount of bandwidth to allocate.

The bandwidth allocated for a priority queue always includes the Layer 2 encapsulation header. However, it does not include other headers, such as ATM cell tax overheads. When you calculate the amount of bandwidth to allocate for a given priority class, you must account for the fact that the Layer 2 headers are included. When ATM is used, you must account for the fact that ATM cell tax overhead is not included. You must also allow bandwidth for the possibility of jitter introduced by routers in the voice path.

Consider this case that uses ATM: Suppose a voice stream of 60 bytes emitting 50 packets per second is encoded using G.729. Prior to converting the voice stream to cells, the meter for the priority queue used for the voice stream assesses the length of the packet after the Layer logical link control (LLC) headers have been added.

Given the 8-byte Layer 2 LLC header, the meter will take into account a 68-byte packet. Because ATM cells are a standard 53 bytes long, before the 68-kbps packet is emitted on the line, it is divided into two 53-byte ATM cells. Thus, the bandwidth consumed by this flow is 106 bytes per packet.

For this case, then, you must configure the bandwidth to be at least 27.2 kbps (68 * 50 * 8 = 27.2 kbps). However, recall that you must also allow for the ATM cell tax overhead, which is not accounted for by the configured bandwidth. In other words, the sum of the bandwidths for all classes must be less than the interface bandwidth by at least 15.2 kbps ([106 - 68] * 50 * 8 = 15.2 kbps). You should also remember to allow bandwidth for router-introduced jitter.

Benefits

Provides Priority Service on ATM VCs and Serial Interface

The PQ scheme allows delay-sensitive data such as voice to be dequeued and sent before packets in other queues are dequeued. This feature provides PQ on ATM VCs.

Admission Control

By configuring the maximum amount of bandwidth allocated for packets belonging to a class, you can avoid starving nonpriority traffic.

Limiting Particles on a Transmission Ring

The Distributed LLQ feature also introduces particle limiting for transmission rings. Before the introduction of Distributed LLQ, the transmission ring depth was not user-configurable. Therefore, a user could experience unavoidable high latencies on a transmission ring.

The Distributed LLQ feature allows users to limit the number of particles on a transmission ring to a predefined limit, effectively lowering the latency on transmission rings.

Restrictions

The following restrictions apply to the Distributed LLQ feature:

- If you use access lists to configure matching port numbers, this feature provides priority matching for all port numbers. Because voice typically exists on even port numbers, and control packets are generated on odd port numbers, control packets are also given priority when using this feature. On very slow links, giving priority to both voice and control packets may produce degraded voice quality.
- The **priority** command can be used in conjunction with the **set** command. The **priority** command cannot be used in conjunction with any other command, including the **random-detect**, **queue-limit**, and **bandwidth** commands.
- The **priority** command can be configured in multiple traffic classes. If the traffic is not CBR traffic, you must configure a large enough *bandwidth-kbps* parameter to absorb the data bursts.
- Because 1 percent of the available bandwidth is reserved for the default traffic class, the sum of the percentage for the **bandwidth percent** and **priority percent** command reservations cannot exceed 99 percent.
- Priority queues can be reserved by either size or percentage values, but not both, in the same policy map. Therefore, if the **priority** command is used without the **percent** option in a policy map, the **bandwidth** command, if used, must also be used without the **percent** option, and vice versa. Similarly, if the **priority percent** command is used in a policy map, the **bandwidth percent** command must be used to specify bandwidth allocation for the class, and vice versa. The **priority** and **priority percent** commands also cannot be used in the same policy map.
- The **bandwidth** and **priority** commands cannot be used in the same class map. These commands can be used together in the same policy map, however.

The following commands cannot be used in the same class or policy map with the **priority** command:

- • **priority percent**

  • **bandwidth percent**

The following commands cannot be used in the same class or policy map with the **priority percentage** command:

- • **priority** (without the **percent** option)

  - **bandwidth** (without the **percent** option)

- The **tx-ring-limit**command can only affect a VBR VC on a PA-A3 port adapter. The **tx-ring-limit** command does not affect UBR VCs.
- DLLQ is not supported on Cisco 7500 series routers with PA-A3-8T1IMA modules.

Prerequisites

To use this feature, you should be familiar with the following features:

- ACLs
- ATM PVCs
- Bandwidth management
- CBWFQ
- LFI
- Virtual templates and virtual access interfaces

Low Latency Queueing for Frame Relay

LLQ for Frame Relay provides a strict priority queue for voice traffic and weighted fair queues for other classes of traffic. With this feature, LLQ is available at the Frame Relay VC level when FRTS is configured.

LLQ, also called PQ/CBWFQ, is a superset of and more flexible than previous Frame Relay QoS offerings, in particular RTP prioritization and PQ/WFQ.

With RTP prioritization and PQ/WFQ, traffic that matches a specified UDP/RTP port range is considered high priority and allocated to the priority queue (PQ). With LLQ for Frame Relay, you set up classes of traffic according to protocol, interface, or access lists, and then define policy maps to establish how the classes are handled in the priority queue and weighted fair queues.

Queues are set up on a per-PVC basis: each PVC has a PQ and an assigned number of fair queues. The fair queues are assigned weights proportional to the bandwidth requirements of each class; a class requiring twice the bandwidth of another will have half the weight. Oversubscription of the bandwidth is not permitted. The CLI will reject a change of configuration that would cause the total bandwidth to be exceeded. This functionality differs from that of WFQ, in which flows are assigned a weight based on IP precedence. WFQ allows higher precedence traffic to obtain proportionately more of the bandwidth, but the more flows there are, the less bandwidth is available to each flow.

The PQ is policed to ensure that the fair queues are not starved of bandwidth. When you configure the PQ, you specify in kbps the maximum amount of bandwidth available to that queue. Packets that exceed that maximum are dropped. There is no policing of the fair queues.

LLQ for Frame Relay is configured using a combination of **class-map**, **policy-map**, and Frame Relay map class commands. The **class-map** command defines traffic classes according to protocol, interface, or access list. The **policy-map** command defines how each class is treated in the queueing system according to bandwidth, priority, queue limit, or WRED. The **service-policy output** map class command attaches a policy map to a Frame Relay VC.

Policies not directly related to LLQ--for example, traffic shaping, setting IP precedence, and policing--are not supported by the **class-map** and **policy-map** commands for Frame Relay VCs. You must use other configuration mechanisms, such as map class commands, to configure these policies.

For information on how to configure LLQ for Frame Relay, see the "Configuring Weighted Fair Queueing" module.

Restrictions

Only the following class map and policy map commands are supported:

- The **match** class-map configuration command
- The **priority**, **bandwidth**, **queue-limit**, **random-detect**, and **fair-queue** policy-map configuration commands

Prerequisites

The following tasks must be completed before LLQ for Frame Relay can be enabled:

- FRTS must be enabled on the interface.
- An output service policy must be configured in the map class associated with the interface, subinterface, or DLCI.
- Any queue other than a FIFO queue that is configured in the map class must be removed. LLQ for Frame Relay cannot be configured if there is already a non-FIFO queue configured, except for the default queue that is created when fragmentation is enabled.

How It Works

LLQ for Frame Relay is used in conjunction with the features described in the following sections:

RTP Prioritization

RTP prioritization provides a strict PQ scheme for voice traffic. Voice traffic is identified by its RTP port numbers and classified into a priority queue configured by the **frame-relay ip rtp priority** map-class configuration command. You classify traffic as voice by specifying an RTP port number range. If traffic matches the specified range, it is classified as voice and queued in the LLQ PQ, and the interface priority queue. If traffic does not fall within the specified RTP port range, it is classified by the service policy of the LLQ scheme.

The **ip rtp priority**command is available in both interface configuration mode and map-class configuration mode. Only the **frame relay ip rtp priority**map-class configuration command is supported in this feature.

Voice over Frame Relay

Voice over Frame Relay (VoFR) uses the LLQ priority queue (PQ) rather than its own PQ mechanism. The **frame-relay voice bandwidth** map-class configuration command configures the total bandwidth available for VoFR traffic. The visible bandwidth made available to the other queues will be the minimum committed information rate (CIR) minus the voice bandwidth.

The **frame-relay voice bandwidth** map-class configuration command also configures a call admission control function, which ensures that sufficient VoFR bandwidth remains before allowing a call. There is no policing of the voice traffic once the call has been established.

For VoFR with no data, all voice and call control packets are queued in the LLQ priority queueing (PQ). For VoFR with data, a VoFR PVC may carry both voice and data packets in different subchannels. VoFR data packets are fragmented and interleaved with voice packets to ensure good latency bounds for voice packets and scalability for voice and data traffic.

Note that when VoFR is enabled, there is no need to configure a priority class map for voice. The only VoFR commands to be used with LLQ for Frame Relay are the **frame-relay voice bandwidth** map-class configuration command and the **vofr data** Frame Relay DLCI configuration command.

Note It is possible--though not recommended--to configure other traffic for the PQ at the same time as VoFR. Doing so could cause delays because interleaving non-VoFR packets in the PQ would not be possible, causing the PQ (and any VoFR packets on it) to be held up during fragmentation until the entire fragmented packet has been sent.

Frame Relay Fragmentation

The purpose of Frame Relay fragmentation (FRF.12) is to support voice and data packets on lower-speed links without causing excessive delay to the voice packets. Large data packets are fragmented and interleaved with the voice packets.

When FRF.12 is configured with LLQ, small packets classified for the PQ pass through unfragmented onto both the LLQ PQ and the high priority interface queue. Large packets destined for PQ are shaped and fragmented when dequeued.

Use the **frame-relay fragment** and **service-policy** map-class configuration commands to enable LLQ with FRF.12.

IP Cisco Express Forwarding Switching

IP CEF switching is not affected by LLQ functionality.

Custom Queueing

CQ allows you to specify a certain number of bytes to forward from a queue each time the queue is serviced, thereby allowing you to share the network resources among applications with specific minimum bandwidth or latency requirements. You can also specify a maximum number of packets in each queue.

For information on how to configure CQ, see the "Configuring Custom Queueing" module.

How It Works

CQ handles traffic by specifying the number of packets or bytes to be serviced for each class of traffic. It services the queues by cycling through them in round-robin fashion, sending the portion of allocated bandwidth for each queue before moving to the next queue. If one queue is empty, the router will send packets from the next queue that has packets ready to send.

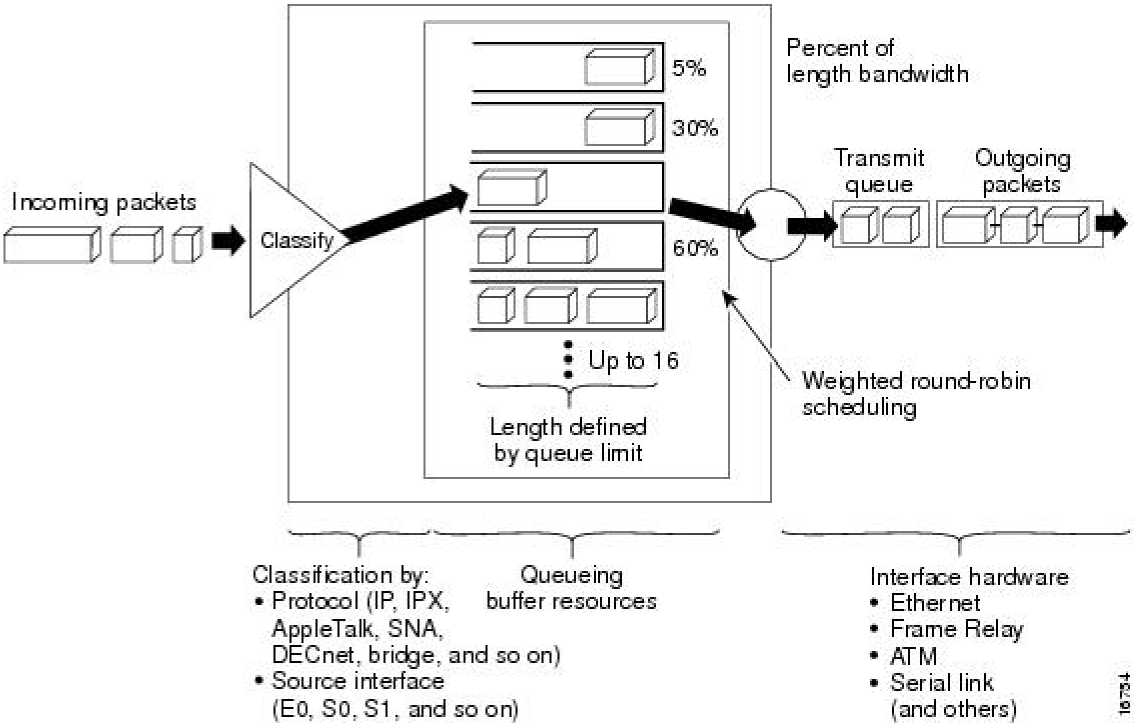
When CQ is enabled on an interface, the system maintains 17 output queues for that interface. You can specify queues 1 through 16. Associated with each output queue is a configurable byte count, which specifies how many bytes of data the system should deliver from the current queue before it moves on to the next queue.

Queue number 0 is a system queue; it is emptied before any of the queues numbered 1 through 16 are processed. The system queues high priority packets, such as keepalive packets and signalling packets, to this queue. Other traffic cannot be configured to use this queue.

For queue numbers 1 through 16, the system cycles through the queues sequentially (in a round-robin fashion), dequeueing the configured byte count from each queue in each cycle, delivering packets in the current queue before moving on to the next one. When a particular queue is being processed, packets are sent until the number of bytes sent exceeds the queue byte count or the queue is empty. Bandwidth used by a particular queue can be indirectly specified only in terms of byte count and queue length.

The figure below shows how CQ behaves.

Figure 2: Custom Queueing


Incoming packets
Classify
5%
30%
60%
Up to 16
Length defined by queue limit
Percent of length bandwidth
Transmit queue
Outgoing packets
Weighted round-robin scheduling
Classification by:
• Protocol (IP, IPX, AppleTalk, SNA, DECnet, bridge, and so on)
• Source interface (E0, S0, S1, and so on)
Queueing buffer resources
Interface hardware
• Ethernet
• Frame Relay
• ATM
• Serial link (and others)
16754

CQ ensures that no application or specified group of applications achieves more than a predetermined proportion of overall capacity when the line is under stress. Like PQ, CQ is statically configured and does not automatically adapt to changing network conditions.

On most platforms, all protocols are classified in the fast-switching path.

Determining Byte Count Values for Queues

In order to allocate bandwidth to different queues, you must specify the byte count for each queue.

How the Byte Count Is Used

The router sends packets from a particular queue until the byte count is exceeded. Once the byte count value is exceeded, the packet that is currently being sent will be completely sent. Therefore, if you set the byte count to 100 bytes and the packet size of your protocol is 1024 bytes, then every time this queue is serviced, 1024 bytes will be sent, not 100 bytes.

For example, suppose one protocol has 500-byte packets, another has 300-byte packets, and a third has 100-byte packets. If you want to split the bandwidth evenly across all three protocols, you might choose to specify byte counts of 200, 200, and 200 for each queue. However, this configuration does not result in a 33/33/33 ratio. When the router services the first queue, it sends a single 500-byte packet; when it services the second queue, it sends a 300-byte packet; and when it services the third queue, it sends two 100-byte packets. The effective ratio is 50/30/20.

Thus, setting the byte count too low can result in an unintended bandwidth allocation.

However, very large byte counts will produce a "jerky" distribution. That is, if you assign 10 KB, 10 KB, and 10 KB to three queues in the example given, each protocol is serviced promptly when its queue is the one being serviced, but it may be a long time before the queue is serviced again. A better solution is to specify 500-byte, 600-byte, and 500-byte counts for the queue. This configuration results in a ratio of 31/38/31, which may be acceptable.

In order to service queues in a timely manner and ensure that the configured bandwidth allocation is as close as possible to the required bandwidth allocation, you must determine the byte count based on the packet size of each protocol, otherwise your percentages may not match what you configure.

Note CQ was modified in Cisco IOS Release 12.1. When the queue is depleted early, or the last packet from the queue does not exactly match the configured byte count, the amount of deficit is remembered and accounted for the next time the queue is serviced. Beginning with Cisco IOS Release 12.1, you need not be as accurate in specifying byte counts as you did when using earlier Cisco IOS releases that did not take deficit into account.

Note Some protocols, such as Internetwork Packet Exchange (IPX), will negotiate the frame size at session startup time.

Determining the Byte Count

To determine the correct byte counts, perform the following steps:

SUMMARY STEPS

1. For each queue, divide the percentage of bandwidth you want to allocate to the queue by the packet size, in bytes. For example, assume the packet size for protocol A is 1086 bytes, protocol B is 291 bytes, and protocol C is 831 bytes. We want to allocate 20 percent for A, 60 percent for B, and 20 percent for C. The ratios would be:
2. Normalize the numbers by dividing by the lowest number:
3. A fraction in any of the ratio values means that an additional packet will be sent. Round up the numbers to the next whole number to obtain the actual packet count.
4. Convert the packet number ratio into byte counts by multiplying each packet count by the corresponding packet size.
5. To determine the bandwidth distribution this ratio represents, first determine the total number of bytes sent after all three queues are serviced:
6. Then determine the percentage of the total number of bytes sent from each queue:
7. If the actual bandwidth is not close enough to the desired bandwidth, multiply the original ratio of 1:11.2:1.3 by the best value, trying to get as close to three integer values as possible. Note that the multiplier you use need not be an integer. For example, if we multiply the ratio by two, we get 2:22.4:2.6. We would now send two 1086-byte packets, twenty-three 291-byte packets, and three 831-byte packets, or 2172/6693/2493, for a total of 11,358 bytes. The resulting ratio is 19/59/22 percent, which is much closer to the desired ratio that we achieved.

DETAILED STEPS

Step 1 For each queue, divide the percentage of bandwidth you want to allocate to the queue by the packet size, in bytes. For example, assume the packet size for protocol A is 1086 bytes, protocol B is 291 bytes, and protocol C is 831 bytes. We want to allocate 20 percent for A, 60 percent for B, and 20 percent for C. The ratios would be:
20/1086, 60/291, 20/831 or

0.01842, 0.20619, 0.02407

Step 2 Normalize the numbers by dividing by the lowest number:
1, 11.2, 1.3

The result is the ratio of the number of packets that must be sent so that the percentage of bandwidth that each protocol uses is approximately 20, 60, and 20 percent.

Step 3 A fraction in any of the ratio values means that an additional packet will be sent. Round up the numbers to the next whole number to obtain the actual packet count.
In this example, the actual ratio will be 1 packet, 12 packets, and 2 packets.

Step 4 Convert the packet number ratio into byte counts by multiplying each packet count by the corresponding packet size.
In this example, the number of packets sent is one 1086-byte packet, twelve 291-byte packets, and two 831-byte packets, or 1086, 3492, and 1662 bytes, respectively, from each queue. These are the byte counts you would specify in your CQ configuration.

Step 5 To determine the bandwidth distribution this ratio represents, first determine the total number of bytes sent after all three queues are serviced:
(1 * 1086) + (12 * 291) + (2 * 831) = 1086 + 3492 + 1662 = 6240

Step 6 Then determine the percentage of the total number of bytes sent from each queue:
1086/6240, 3492/6240, 1662/6240 = 17.4, 56, and 26.6 percent

This result is close to the desired ratio of 20/60/20.

Step 7 If the actual bandwidth is not close enough to the desired bandwidth, multiply the original ratio of 1:11.2:1.3 by the best value, trying to get as close to three integer values as possible. Note that the multiplier you use need not be an integer. For example, if we multiply the ratio by two, we get 2:22.4:2.6. We would now send two 1086-byte packets, twenty-three 291-byte packets, and three 831-byte packets, or 2172/6693/2493, for a total of 11,358 bytes. The resulting ratio is 19/59/22 percent, which is much closer to the desired ratio that we achieved.

What to Do Next

The bandwidth that a custom queue will receive is given by the following formula:

```
(queue byte count / total byte count of all queues) * bandwidth capacity of the interface
```

where bandwidth capacity is equal to the interface bandwidth minus the bandwidth for priority queues.

Window Size

Window size also affects the bandwidth distribution. If the window size of a particular protocol is set to one, then that protocol will not place another packet into the queue until it receives an acknowledgment. The CQ algorithm moves to the next queue if the byte count is exceeded or no packets are in that queue.

Therefore, with a window size of one, only one frame will be sent each time. If your frame count is set to 2 kilobytes, and your frame size is 256 bytes, then only 256 bytes will be sent each time this queue is serviced.

Why Use CQ

You can use the Cisco IOS QoS CQ feature to provide specific traffic guaranteed bandwidth at a potential congestion point, assuring the traffic a fixed portion of available bandwidth and leaving the remaining bandwidth to other traffic. For example, you could reserve half of the bandwidth for SNA data, allowing the remaining half to be used by other protocols.

If a particular type of traffic is not using the bandwidth reserved for it, then unused bandwidth can be dynamically allocated to other traffic types.

Restrictions

CQ is statically configured and does not adapt to changing network conditions. With CQ enabled, the system takes longer to switch packets than FIFO because the packets are classified by the processor card.

Priority Queueing

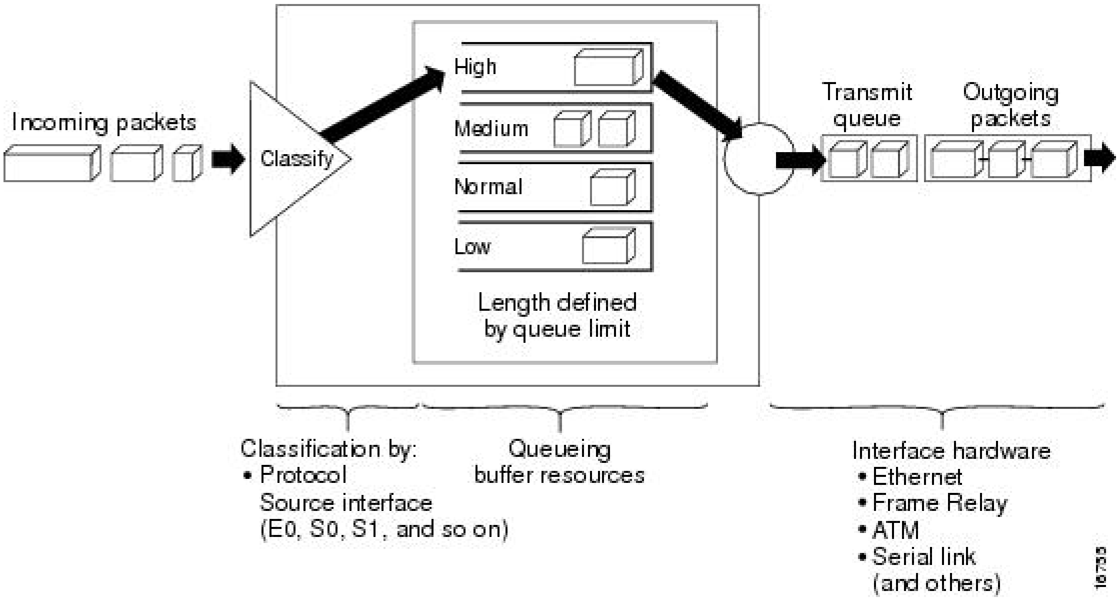
PQ allows you to define how traffic is prioritized in the network. You configure four traffic priorities. You can define a series of filters based on packet characteristics to cause the router to place traffic into these four queues; the queue with the highest priority is serviced first until it is empty, then the lower queues are serviced in sequence.

For information on how to configure PQ, see the "Configuring Priority Queueing" module.

How It Works

During transmission, PQ gives priority queues absolute preferential treatment over low priority queues; important traffic, given the highest priority, always takes precedence over less important traffic. Packets are classified based on user-specified criteria and placed into one of the four output queues--high, medium, normal, and low--based on the assigned priority. Packets that are not classified by priority fall into the normal queue. The figure below illustrates this process.

Figure 3: Priority Queueing



When a packet is to be sent out an interface, the priority queues on that interface are scanned for packets in descending order of priority. The high priority queue is scanned first, then the medium priority queue, and so on. The packet at the head of the highest queue is chosen for transmission. This procedure is repeated every time a packet is to be sent.

The maximum length of a queue is defined by the length limit. When a queue is longer than the queue limit, all additional packets are dropped.

Note The priority output queueing mechanism can be used to manage traffic from all networking protocols. Additional fine-tuning is available for IP and for setting boundaries on the packet size.

How Packets Are Classified for Priority Queueing

A priority list is a set of rules that describe how packets should be assigned to priority queues. A priority list might also describe a default priority or the queue size limits of the various priority queues.

Packets can be classified by the following criteria:

- Protocol or subprotocol type
- Incoming interface
- Packet size
- Fragments
- Access list

Keepalives sourced by the network server are always assigned to the high priority queue; all other management traffic (such as Interior Gateway Routing Protocol (IGRP) updates) must be configured. Packets that are not classified by the priority list mechanism are assigned to the normal queue.

Why Use Priority Queueing

PQ provides absolute preferential treatment to high priority traffic, ensuring that mission-critical traffic traversing various WAN links gets priority treatment. In addition, PQ provides a faster response time than do other methods of queueing.

Although you can enable priority output queueing for any interface, it is best used for low-bandwidth, congested serial interfaces.

Restrictions

When choosing to use PQ, consider that because lower priority traffic is often denied bandwidth in favor of higher priority traffic, use of PQ could, in the worst case, result in lower priority traffic never being sent. To avoid inflicting these conditions on lower priority traffic, you can use traffic shaping or CAR to rate-limit the higher priority traffic.

PQ introduces extra overhead that is acceptable for slow interfaces, but may not be acceptable for higher speed interfaces such as Ethernet. With PQ enabled, the system takes longer to switch packets because the packets are classified by the processor card.

PQ uses a static configuration and does not adapt to changing network conditions.

PQ is not supported on any tunnels.

Bandwidth Management

RSVP, CBWFQ, LLQ, IP RTP Priority, Frame Relay IP RTP Priority, and Frame Relay PIPQ can all reserve and consume bandwidth, up to a maximum of the reserved bandwidth on an interface.

To allocate bandwidth, you can use one of the following commands:

- For RSVP, use the **ip rsvp bandwidth** command.
- For CBWFQ, use the **bandwidth** policy-map class configuration command. For more information on CBWFQ bandwidth allocation, see the section Class-Based Weighted Fair Queueing, on page 11 in this module. For LLQ, you can allocate bandwidth using the **priority** command. For more information on LLQ bandwidth allocation, see the section Frame Relay PVC Interface Priority Queueing, on page 17 in this module.

- For IP RTP Priority, use the **ip rtp priority**command. For more information on IP RTP Priority bandwidth allocation, see the section IP RTP Priority, on page 15 in this module.
- For Frame Relay IP RTP Priority, use the **frame-relay ip rtp priority** command. For more information on Frame Relay IP RTP Priority, see the section Frame Relay IP RTP Priority, on page 17 in this module.
- For Frame Relay PVC Interface Priority Queueing, use the **frame-relay interface-queue priority** command. For more information on Frame Relay PIPQ, see the section Frame Relay PVC Interface Priority Queueing, on page 17 in this module.

When you configure these commands, be aware of bandwidth limitations and configure bandwidth according to requirements in your network. Remember, the sum of all bandwidths cannot exceed the maximum reserved bandwidth. The default maximum bandwidth is 75 percent of the total available bandwidth on the interface. The remaining 25 percent of bandwidth is used for overhead, including Layer 2 overhead, routing traffic, and best-effort traffic.

CHAPTER 2

IPv6 QoS: Queueing

Class-based and flow-based queueing are supported for IPv6.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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.

Information About IPv6 QoS: Queueing

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, queuing, 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 that are 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 that you establish for your network. In particular, if the same network is also carrying IPv4 traffic along with IPv6 traffic, 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**, **set precedence**, and **set 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 Management in IPv6 Networks

Once you have marked the traffic, you can use the markings to build a policy and classify traffic on the rest of the network segments. If you keep the policy simple (for example approximately four classes), it will be easier to manage. Class-based and flow-based queueing are supported for IPv6. The processes and tasks use the same commands and arguments to configure various queueing options for both IPv4 and IPv6.

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

Related Topic	Document Title
IPv6 commands	Cisco IOS IPv6 Command Reference
Cisco IOS IPv6 features	*Cisco IOS IPv6 Feature Mapping*
QoS Queuing features	“Congestion Management Overview” module

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: Queueing

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 IPv6 QoS: Queueing

Feature Name	Releases	Feature Information
IPv6 QoS: Queueing	12.2(33)SRA 12.2(18)SXE 12.2(13)T 12.3 12.3(2)T 12.4 12.4(2)T	Class-based and flow-based queueing are supported for IPv6.

CHAPTER 3

Configuring Weighted Fair Queueing

Feature History

Release	Modification
Cisco IOS	For information about feature support in Cisco IOS software, use Cisco Feature Navigator.

This module describes the tasks for configuring flow-based weighted fair queueing (WFQ), distributed WFQ (DWFQ), and class-based WFQ (CBWFQ), and distributed class-based WFQ (DCBWFQ) and the related features described in the following section, which provide strict priority queueing (PQ) within WFQ or CBWFQ:

- IP RTP Priority Queueing
- Frame Relay IP RTP Priority Queueing
- Frame Relay PVC Interface Priority Queueing
- Low Latency Queueing
- Distributed Low Latency Queueing
- Low Latency Queueing (LLQ) for Frame Relay
- Burst Size in Low Latency Queueing
- Per-VC Hold Queue Support for ATM Adapters

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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.

Flow-Based Weighted Fair Queueing Configuration Task List

WFQ provides traffic priority management that automatically sorts among individual traffic streams without requiring that you first define access lists. WFQ can also manage duplex data streams such as those between pairs of applications, and simplex data streams such as voice or video. There are two categories of WFQ sessions: high bandwidth and low bandwidth. Low-bandwidth traffic has effective priority over high-bandwidth traffic, and high-bandwidth traffic shares the transmission service proportionally according to assigned weights.

When WFQ is enabled for an interface, new messages for high-bandwidth traffic streams are discarded after the configured or default congestive messages threshold has been met. However, low-bandwidth conversations, which include control message conversations, continue to enqueue data. As a result, the fair queue may occasionally contain more messages than its configured threshold number specifies.

With standard WFQ, packets are classified by flow. Packets with the same source IP address, destination IP address, source TCP or User Datagram Protocol (UDP) port, or destination TCP or UDP port belong to the same flow. WFQ allocates an equal share of the bandwidth to each flow. Flow-based WFQ is also called fair queueing because all flows are equally weighted.

The Cisco IOS software provides two forms of flow-based WFQ:

- Standard WFQ, which is enabled by default on all serial interfaces that run at 2 Mbps or below, and can run on all Cisco serial interfaces.
- Distributed WFQ, which runs only on Cisco 7000 series routers with a Route Switch Processor (RSP)-based RSP7000 interface processor or Cisco 7500 series routers with a Versatile Interface Processor (VIP)-based VIP2-40 or greater interface processor. (A VIP2-50 interface processor is strongly recommended when the aggregate line rate of the port adapters on the VIP is greater than DS3. A VIP2-50 interface processor is required for OC-3 rates.) .

Flow-based WFQ uses a traffic data stream discrimination registry service to determine to which traffic stream a message belongs. Refer to the table accompanying the description of the **fair-queue** (WFQ) command in the Cisco IOS Quality of Service Solutions Command Reference for the attributes of a message that are used to classify traffic into data streams.

Defaults are provided for the congestion threshold after which messages for high-bandwidth conversations are dropped, and for the number of dynamic and reservable queues; however, you can fine-tune your network operation by changing these defaults. Refer to the tables accompanying the description of the **fair-queue** (WFQ) command in the Cisco IOS Quality of Service Solutions Command Reference for the default number of dynamic queues that WFQ and CBWFQ use when they are enabled on an interface or ATM VC. These values do not apply for DWFQ.

Note WFQ is the default queueing mode on interfaces that run at E1 speeds (2.048 Mbps) or below. It is enabled by default for physical interfaces that do not use Link Access Procedure, Balanced (LAPB), X.25, or Synchronous Data Link Control (SDLC) encapsulations. WFQ is not an option for these protocols. WFQ is also enabled by default on interfaces configured for Multilink PPP (MLP). However, if custom queueing (CQ) or priority queueing (PQ) is enabled for a qualifying link, it overrides fair queueing, effectively disabling it. Additionally, WFQ is automatically disabled if you enable autonomous or silicon switching.

If you enable flow-based DWFQ and then enable class-based DWFQ (either QoS-group based or ToS-based), class-based DWFQ will replace flow-based DWFQ.

If you enable class-based DWFQ and then want to switch to flow-based DWFQ, you must disable class-based DWFQ using the **no fair-queue class-based** command before enabling flow-based DWFQ.

If you enable one type of class-based DWFQ and then enable the other type, the second type will replace the first.

DWFQ runs only on Cisco 7000 series routers with an RSP-based RSP7000 interface processor or Cisco 7500 series routers with a VIP-based VIP2-40 or greater interface processor. (A VIP2-50 interface processor is strongly recommended when the aggregate line rate of the port adapters on the VIP is greater than DS3. A VIP2-50 interface processor is required for OC-3 rates.)

DWFQ can be configured on interfaces but not subinterfaces. It is not supported on Fast EtherChannel, tunnel, or other logical or virtual interfaces such as MLP.

For flow-based DWFQ, packets are classified by flow. Packets with the same source IP address, destination IP address, source TCP or UDP port, destination TCP or UDP port, and protocol belong to the same flow.

To configure flow-based WFQ, perform the tasks described in the following sections.

Flow-based WFQ is supported on unavailable bit rate (UBR), variable bit rate (VBR), and available bit rate (ABR) ATM connections.

Configuring WFQ

Command	Purpose
`Router(config-if)#` **fair-queue** [*congestive-discard-threshold* [*dynamic-queues* [*reservable-queues*]]]	Configures an interface to use WFQ.

Monitoring Fair Queueing

Command	Purpose
`Router#` **show interfaces** [*interface*]	Displays statistical information specific to an interface.
`Router#` **show queue** *interface-type interface-number*	Displays the contents of packets inside a queue for a particular interface or virtual circuit (VC).
`Router#` **show queueing fair**	Displays status of the fair queueing configuration.

Distributed Weighted Fair Queueing Configuration Task List

To configure DWFQ, perform one of the mutually exclusive tasks described in the following sections:

Configuring Flow-Based DWFQ

SUMMARY STEPS

1. Router(config-if)# **fair-queue**
2. Router(config-if)# **fair-queue aggregate-limit** *aggregate-packet*
3. Router(config-if)# **fair-queue individual-limit** *individual-packet*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **fair-queue**	Enables flow-based DWFQ.
Step 2	Router(config-if)# **fair-queue aggregate-limit** *aggregate-packet*	(Optional) Sets the total number of buffered packets before some packets may be dropped. Below this limit, packets will not be dropped. **Note** In general, you should not change the aggregate, individual, or class limit value from the default. Use the **fair-queue aggregate-limit**, **fair-queue individual-limit**, and **fair-queue limit** commands only if you have determined that you would benefit from using different values, based on your particular situation.
Step 3	Router(config-if)# **fair-queue individual-limit** *individual-packet*	(Optional) Sets the maximum queue size for individual per-flow queues during periods of congestion.

Configuring QoS-Group-Based DWFQ

SUMMARY STEPS

1. Router(config-if)# **fair-queue qos-group**
2. Router(config-if)# **fair-queue qos-group** *number* **weight** *weight*
3. Router(config-if)# **fair-queue aggregate-limit** *aggregate-packet*
4. Router(config-if)# **fair-queue individual-limit** *individual-packet*
5. Router(config-if)# **fair-queue qos-group** *number* **limit** *class-packet*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **fair-queue qos-group**	Enables QoS-group-based DWFQ.
Step 2	Router(config-if)# **fair-queue qos-group** *number* **weight** *weight*	For each QoS group, specifies the percentage of the bandwidth to be allocated to each class.
Step 3	Router(config-if)# **fair-queue aggregate-limit** *aggregate-packet*	(Optional) Sets the total number of buffered packets before some packets may be dropped. Below this limit, packets will not be dropped. **Note** In general, you should not change the aggregate, individual, or class limit value from the default. Use the **fair-queue aggregate-limit**, **fair-queue individual-limit**, and **fair-queue limit** commands only if you have determined that you would benefit from using different values, based on your particular situation.

	Command or Action	Purpose
Step 4	Router(config-if)# **fair-queue individual-limit** *individual-packet*	(Optional) Sets the maximum queue size for every per-flow queue during periods of congestion.
Step 5	Router(config-if)# **fair-queue qos-group** *number* **limit** *class-packet*	(Optional) Sets the maximum queue size for a specific QoS group queue during periods of congestion.

Configuring Type of Service-Based DWFQ

SUMMARY STEPS

1. Router(config-if)# **fair-queue tos**
2. Router(config-if)# **fair-queue tos** *number* **weight** *weight*
3. Router(config-if)# **fair-queue aggregate-limit** *aggregate-packet*
4. Router(config-if)# **fair-queue individual-limit** *individual-packet*
5. Router(config-if)# **fair-queue tos** *number* **limit** *class-packet*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **fair-queue tos**	Enables ToS-based DWFQ
Step 2	Router(config-if)# **fair-queue tos** *number* **weight** *weight*	(Optional) For each ToS class, specifies the percentage of the bandwidth to be allocated to each class.
Step 3	Router(config-if)# **fair-queue aggregate-limit** *aggregate-packet*	(Optional) Sets the total number of buffered packets before some packets may be dropped. Below this limit, packets will not be dropped. **Note** In general, you should not change the aggregate, individual, or class limit value from the default. Use the **fair-queue aggregate-limit**, **fair-queue individual-limit**, and **fair-queue limit**commands only if you have determined that you would benefit from using different values, based on your particular situation.
Step 4	Router(config-if)# **fair-queue individual-limit** *individual-packet*	(Optional) Sets the maximum queue size for every per-flow queue during periods of congestion.
Step 5	Router(config-if)# **fair-queue tos** *number* **limit** *class-packet*	(Optional) Sets the maximum queue size for a specific ToS queue during periods of congestion.

Monitoring DWFQ

Command	Purpose
`Router#` **show interfaces** [*interface*]	Displays the statistical information specific to an interface.
`Router#` **show queueing fair-queue**	Displays status of the fair queueing configuration.

Class-Based Weighted Fair Queueing Configuration Task List

CBWFQ is supported on VBR and ABR ATM connections. It is not supported on UBR connections.

Defining Class Maps

SUMMARY STEPS

1. Router(config)# **class-map** *class-map-name*
2. Do one of the following:
    - Router(config-cmap)# **match access-group** {*access-group*|**name** *access-group-name*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **class-map** *class-map-name*	Specifies the name of the class map to be created.
Step 2	Do one of the following: • Router(config-cmap)# **match access-group** {*access-group*|**name** *access-group-name*} **Example:** `Router(config-cmap)#` **match input-interface** *interface-name* **Example:** `Router(config-cmap)#` **match protocol** *protocol*	Specifies the name of the access control list (ACL) against whose contents packets are checked to determine if they belong to the class. CBWFQ supports numbered and named ACLs. Specifies the name of the input interface used as a match criterion against which packets are checked to determine if they belong to the class. Specifies the name of the protocol used as a match criterion against which packets are checked to determine if they belong to the class. Specifies the value of the EXP field to be used as a match criterion against which packets are checked to determine if they belong to the class. **Note** Other match criteria can be used when defining class maps. For additional match criteria, see "Applying QoS Features Using the MQC" module.

	Command or Action	Purpose
	Example: `Router(config-cmap)#` **match mpls experimental** *number*	

Configuring Class Policy in the Policy Map

To configure a policy map and create class policies that make up the service policy, use the **policy-map** command to specify the policy map name, then use one or more of the following commands to configure policy for a standard class or the default class:

- **class**
- **bandwidth** (policy-map class)
- **fair-queue** (for class-default class only)
- **queue-limit** or **random-detect**

For each class that you define, you can use one or more of the listed commands to configure class policy. For example, you might specify bandwidth for one class and both bandwidth and queue limit for another class.

The default class of the policy map (commonly known as the class-default class) is the class to which traffic is directed if that traffic does not satisfy the match criteria of other classes whose policy is defined in the policy map.

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 must not exceed 75 percent of the available bandwidth on the interface. The other 25 percent is used for control and routing traffic. (To override the 75 percent limitation, use the **max-reserved bandwidth** command.) If not all of the bandwidth is allocated, the remaining bandwidth is proportionally allocated among the classes, based on their configured bandwidth.

The class-default class is used to classify traffic that does not fall into one of the defined classes. Once a packet is classified, all of the standard mechanisms that can be used to differentiate service among the classes apply. The class-default class was predefined when you created the policy map, but you must configure it. If no default class is configured, then by default the traffic that does not match any of the configured classes is flow classified and given best-effort treatment.

To configure class policies in a policy map, perform the optional tasks described in the following sections. If you do not perform the steps in these sections, the default actions are used.

Configuring Class Policy Using Tail Drop

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class** *class-name*
3. Router(config-pmap-c)# **bandwidth**{*bandwidth-kbps* | **percent** *percent*}
4. Router(config-pmap-c)# **queue-limit** *number-of-packets*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a class to be created and included in the service policy. **Note** To configure policy for more than one class in the same policy map, repeat Configuring Class Policy Using Tail Drop through Configuring Class Policy Using Tail Drop. Note that because this set of commands uses the **queue-limit** command, the policy map uses tail drop, not Weighted Random Early Detection (WRED) packet drop.
Step 3	Router(config-pmap-c)# **bandwidth**{*bandwidth-kbps* | **percent** *percent*}	Specifies the amount of bandwidth, in kbps, or percentage of available bandwidth, to be assigned to the class. The amount of bandwidth configured should be large enough to also accommodate Layer 2 overhead.
Step 4	Router(config-pmap-c)# **queue-limit** *number-of-packets*	Specifies the maximum number of packets that can be queued for the class.

Configuring Class Policy Using WRED Packet Drop

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class** *class-name*
3. Router(config-pmap-c)# **bandwidth**{*bandwidth-kbps* | **percent** *percent*}
4. Router(config-pmap-c)# **random-detect**
5. Do one of the following:
   - Router(config-pmap-c)# **random-detect exponential-weighting-constant** *exponent*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a class to be created and included in the service policy. **Note** To configure policy for more than one class in the same policy map, repeat Configuring Class Policy Using WRED Packet Drop through Configuring Class Policy Using WRED Packet Drop. Note that this set of commands uses WRED packet drop, not tail drop.
Step 3	Router(config-pmap-c)# **bandwidth**{*bandwidth-kbps* | **percent** *percent*}	Specifies the amount of bandwidth, in kbps, or percentage of available bandwidth to be assigned to the class. The amount of bandwidth configured should be large enough to also accommodate Layer 2 overhead.
Step 4	Router(config-pmap-c)# **random-detect**	Enables WRED. The class policy will drop packets using WRED instead of tail drop. **Note** If you configure a class in a policy map to use WRED for packet drop instead of tail drop, you must ensure that WRED is not configured on the interface to which you intend to attach that service policy.
Step 5	Do one of the following: • Router(config-pmap-c)# **random-detect exponential-weighting-constant** *exponent* **Example:** `Router(config-pmap-c)#` **random-detect precedence** *precedence min-threshold max-threshold mark-prob-denominator*	Configures the exponential weight factor used in calculating the average queue length. Configures WRED parameters for packets with a specific IP precedence. Repeat this command for each precedence.

Configuring the Class-Default Class Policy for WFQ

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class class-default** *default-class-name*
3. Do one of the following:
   - Router(config-pmap-c)# **bandwidth**{*bandwidth-kbps* | **percent** *percent*}
4. Router(config-pmap-c)# **queue-limit** *number-of-packets*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-pmap)# **class class-default** *default-class-name*	Specifies the default class so that you can configure or modify its policy.
Step 3	Do one of the following: • Router(config-pmap-c)# **bandwidth**{*bandwidth-kbps* | **percent** *percent*} **Example:** `Router(config-pmap-c)#` **fair-queue** [*number-of-dynamic-queues*]	Specifies the amount of bandwidth, in kbps, or percentage of available bandwidth to be assigned to the class. The amount of bandwidth configured should be large enough to also accommodate Layer 2 overhead. Specifies the number of dynamic queues to be reserved for use by flow-based WFQ running on the default class. The number of dynamic queues is derived from the bandwidth of the interface. Refer to the tables accompanying the description of the **fair-queue** (WFQ) command in the Cisco IOS Quality of Service Solutions Command Reference for the default number of dynamic queues that WFQ and CBWFQ use when they are enabled on an interface or ATM VC.
Step 4	Router(config-pmap-c)# **queue-limit** *number-of-packets*	Specifies the maximum number of packets that the queue for the default class can accumulate.

Attaching the Service Policy and Enabling CBWFQ

Command	Purpose
`Router(config-if)#` **service-policy output** *policy-map*	Enables CBWFQ and attaches the specified service policy map to the output interface. **Note** Configuring CBWFQ on a physical interface is only possible if the interface is in the default queueing mode. Serial interfaces at E1 (2.048 Mbps) and below use WFQ by default--other interfaces use FIFO by default. Enabling CBWFQ on a physical interface overrides the default interface queueing method. Enabling CBWFQ on an ATM permanent virtual circuit (PVC) does not override the default queueing method.

Modifying the Bandwidth for an Existing Policy Map Class

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class** *class-name*
3. Router(config-pmap-c)# **bandwidth**{*bandwidth-kbps* | **percent** *percent*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the policy map containing the class to be modified.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a class whose bandwidth you want to modify.
Step 3	Router(config-pmap-c)# **bandwidth**{*bandwidth-kbps* | **percent** *percent*}	Specifies the new amount of bandwidth, in kbps, or percentage of available bandwidth to be used to reconfigure the class. The amount of bandwidth configured should be large enough to also accommodate Layer 2 overhead.

Modifying the Queue Limit for an Existing Policy Map Class

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class** *class-name*
3. Router(config-pmap-c)# **queue-limit** *number-of-packets*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the policy map containing the class to be modified.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a class whose queue limit you want to modify.
Step 3	Router(config-pmap-c)# **queue-limit** *number-of-packets*	Specifies the new maximum number of packets that can be queued for the class to be reconfigured. The default and maximum number of packets is 64.

Deleting Class Maps From Service Policy Maps

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **no class** *class-name*
3. Router(config-pmap-c)# **no class class-default**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the policy map containing the classes to be deleted.
Step 2	Router(config-pmap)# **no class** *class-name*	Specifies the name of the classes to be deleted.
Step 3	Router(config-pmap-c)# **no class class-default**	Deletes the default class.

Deleting Policy Maps

Command	Purpose
`Router(config)#` **no policy-map** *policy-map*	Specifies the name of the policy map to be deleted.

Verifying Configuration of Policy Maps and Their Classes

Command	Purpose
`Router#` **show policy-map** *policy-map*	Displays the configuration of all classes that make up the specified policy map.
`Router#` **show policy-map** *policy-map* **class** *class-name*	Displays the configuration of the specified class of the specified policy map.
`Router#` **show policy-map interface** *interface-name*	Displays the configuration of all classes configured for all policy maps on the specified interface. **Note** The counters displayed after issuing the **show policy-map interface** command are updated only if congestion is present on the interface.
`Router#` **show queue** *interface-type interface-number*	Displays queueing configuration and statistics for a particular interface.

Distributed Class-Based Weighted Fair Queueing Configuration Task List

To configure DCBWFQ, perform the tasks described in the following sections. Although all the tasks are listed as optional, you must complete the task in either the first or second section.

DCBWFQ is configured using user-defined traffic classes and service policies. Traffic classes and service policies are configured using the Modular Quality of Service Command-Line Interface (CLI) feature.

Modifying the Bandwidth for an Existing Traffic Class

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class** *class-name*
3. Router(config-pmap-c)# **bandwidth** *bandwidth-kbps*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the traffic policy to be created or modified.
Step 2	Router(config-pmap)# **class** *class-name* **Example:**	Specifies the name of a traffic class whose bandwidth you want to modify.
Step 3	Router(config-pmap-c)# **bandwidth** *bandwidth-kbps*	Specifies the amount of allocated bandwidth, in kbps, to be reserved for the traffic class in congested network environments. **Note** After configuring the traffic policy with the **policy-map** command, you must still attach the traffic policy to an interface before it is successfully enabled. For information on attaching a traffic policy to an interface, see the "Applying QoS Features Using the MQC" module.

Modifying the Queue Limit for an Existing Traffic Class

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class***class-name*
3. Router(config-pmap-c)# **queue-limit** *number-of-packets*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the traffic policy to be created or modified.
Step 2	Router(config-pmap)# **class***class-name*	Specifies the name of a traffic class whose queue limit you want to modify.
Step 3	Router(config-pmap-c)# **queue-limit** *number-of-packets*	Specifies the new maximum number of packets that can be queued for the traffic class to be reconfigured. The default and maximum number of packets is 64. **Note** After configuring the service policy with the **policy-map** command, you must still attach the traffic policy to an interface before it is successfully enabled. For information on attaching a traffic policy to an interface, see the "Applying QoS Features Using the MQC" module.

Monitoring and Maintaining DCBWFQ

Command	Purpose
`Router#` **show policy-map**	Displays all configured traffic policies.
`Router#` **show policy-map** *policy-map-name*	Displays the user-specified traffic policy.
`Router#` **show policy-map interface**	Displays statistics and configurations of all input and output policies attached to an interface.
`Router#` **show policy-map interface** *interface-spec*	Displays configuration and statistics of the input and output policies attached to a particular interface.
`Router#` **show policy-map interface** *interface-spec input*	Displays configuration and statistics of the input policy attached to an interface.
`Router#` **show policy-map interface** *interface-spec output*	Displays configuration statistics of the output policy attached to an interface.
`Router#` **show policy-map** [**interface** *interface-spec* [input | output] [**class** *class-name*]]]]	Displays the configuration and statistics for the class name configured in the policy.

IP RTP Priority Configuration Task List

Frame Relay Traffic Shaping (FRTS) and Frame Relay Fragmentation (FRF.12 or higher) must be configured before the Frame Relay IP RTP Priority feature is used.

Configuring IP RTP Priority

Command	Purpose
`Router(config-if)#` **ip rtp priority** *starting-rtp-port-number port-number-range bandwidth*	Reserves a strict priority queue for a set of RTP packet flows belonging to a range of UDP destination ports. **Note** The **ip rtp reserve** and **ip rtp priority** commands cannot be configured on the same interface. **Caution** Because the **ip rtp priority** command gives absolute priority over other traffic, it should be used with care. In the event of congestion, if the traffic exceeds the configured bandwidth, then all the excess traffic is dropped.

Verifying IP RTP Priority

Command	Purpose
`Router#` **show queue** *interface-type interface-number*	Displays queueing configuration and statistics for a particular interface.

Monitoring and Maintaining IP RTP Priority

Command	Purpose
`Router#` **debug priority**	Displays priority queueing output if packets are dropped from the priority queue.
`Router#` **show queue** *interface-type interface-number*	Displays queueing configuration and statistics for a particular interface.

Frame Relay IP RTP Priority Configuration Task List

Configuring Frame Relay IP RTP Priority

Command	Purpose
`Router(config-map-class)#` **frame-relay ip rtp priority** *starting-rtp-port-number port-number-range bandwidth*	Reserves a strict priority queue for a set of RTP packet flows belonging to a range of UDP destination ports. **Note** Because the **frame-relay ip rtp priority** command gives absolute priority over other traffic, it should be used with care. In the event of congestion, if the traffic exceeds the configured bandwidth, then all the excess traffic is dropped.

Verifying Frame Relay IP RTP Priority

Command	Purpose
`Router#` **show frame relay pvc**	Displays statistics about PVCs for Frame Relay interfaces.
`Router#` **show queue** *interface-type interface-number*	Displays fair queueing configuration and statistics for a particular interface.
`Router#` **show traffic-shape queue**	Displays information about the elements queued at a particular time at the VC data-link connection identifier (DLCI) level.

Monitoring and Maintaining Frame Relay IP RTP Priority

Command	Purpose
`Router#` **debug priority**	Displays priority queueing output if packets are dropped from the priority queue.

Frame Relay PVC Interface Priority Configuration Task List

Configuring PVC Priority in a Map Class

SUMMARY STEPS

1. Router(config)# **map-class frame-relay** *map-class-name*
2. Router(config-map-class)# **frame-relay interface-queue priority**{**high**| **medium**| **normal**| **low**}

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **map-class frame-relay** *map-class-name*	Specifies a Frame Relay map class.
Step 2	Router(config-map-class)# **frame-relay interface-queue priority**{**high**| **medium**| **normal**| **low**}	Assigns a PVC priority level to a Frame Relay map class.

Enabling Frame Relay PIPQ and Setting Queue Limits

SUMMARY STEPS

1. Router(config)# **interface** *type number* [*name-tag*]
2. Router(config-if)# **encapsulation frame-relay**[**cisco** | **ietf**]
3. Router(config-if)# **frame-relay interface-queue priority** [*high-limit medium-limit normal-limit low-limit*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface** *type number* [*name-tag*]	Configures an interface type and enters interface configuration mode.
Step 2	Router(config-if)# **encapsulation frame-relay**[**cisco** | **ietf**]	Enables Frame Relay encapsulation.
Step 3	Router(config-if)# **frame-relay interface-queue priority** [*high-limit medium-limit normal-limit low-limit*]	Enables Frame Relay PIPQ and sets the priority queue limits.

Assigning a Map Class to a PVC

SUMMARY STEPS

1. Router(config-if)# **frame-relay interface-dlci** *dlci*
2. Router(config-fr-dlci)# **class** *map-class-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config-if)# **frame-relay interface-dlci** *dlci*	Specifies a single PVC on a Frame Relay interface.
Step 2	Router(config-fr-dlci)# **class** *map-class-name*	Associates a map class with a specified PVC.

Verifying Frame Relay PIPQ

Command	Purpose
Router# **show frame-relay pvc** [**interface** *interface*] [*dlci*]	Displays statistics about PVCs for Frame Relay interfaces.
Router# **show interfaces** [*type number*] [*first*] [*last*]	Displays the statistical information specific to a serial interface.
Router# **show queueing** [**custom** | **fair** | **priority** | **random-detect** [**interface** *atm_subinterface* [**vc** [[**vpi**/] **vci**]]]]	Lists all or selected configured queueing strategies.

Monitoring and Maintaining Frame Relay PIPQ

Command	Purpose
`Router#` **debug priority**	Displays priority queueing output if packets are dropped from the priority queue.
`Router#` **show frame-relay pvc** `[`**interface** *interface*`]` `[`*dlci*`]`	Displays statistics about PVCs for Frame Relay interfaces.
`Router#` **show interfaces** `[`*type number*`]` `[`*first*`]` `[`*last*`]`	Displays the statistical information specific to a serial interface.
`Router#` **show queue** *interface-name interface-number* `[`**vc** `[`*vpi*`/]` *vci*`]` `[`*queue-number*`]`	Displays the contents of packets inside a queue for a particular interface or VC.
`Router#` **show queueing** `[`**custom** `|` **fair** `|` **priority** `|` **random-detect** `[`**interface** *atm_subinterface* `[`**vc** `[[`*vpi*`/]` *vci*`]]]]`	Lists all or selected configured queueing strategies.

Low Latency Queueing Configuration Task List

Configuring LLQ

Command	Purpose
`Router(config-pmap-c)#` **priority** *bandwidth*	Reserves a strict priority queue for this class of traffic.

Verifying LLQ

Command	Purpose
`Router#` **show queue** *interface-type interface-number*	Displays queueing configuration and statistics for a particular interface.

Monitoring and Maintaining LLQ

Command	Purpose
`Router#` **debug priority**	Displays priority queueing output if packets are dropped from the priority queue.
`Router#` **show queue** *interface-type interface-number*	Displays queueing configuration and statistics for a particular interface.
`Router#` **show policy-map interface** *interface-name*	Displays the configuration of all classes configured for all traffic policies on the specified interface. Displays if packets and bytes were discarded or dropped for the priority class in the traffic policy attached to the interface.

Distributed LLQ Configuration Task List

Configuring a Priority Queue for an Amount of Available Bandwidth

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-name*
2. Router(config-pmap)# **class** *class-name*
3. Router(config-pmap-c)# **priority** *kpbs* [*bytes*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-name*	Specifies the name of the policy map to configure. Enters policy-map configuration mode.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a predefined class included in the service policy. Enters policy-map class configuration mode.
Step 3	Router(config-pmap-c)# **priority** *kpbs* [*bytes*	Reserves a priority queue with a specified amount of available bandwidth for CBWFQ traffic. **Note** The traffic policy configured in this section is not yet attached to an interface. For information on attaching a traffic policy to an interface, see the "Applying QoS Features Using the MQC" module.

Configuring a Priority Queue for a Percentage of Available Bandwidth

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-name*
2. Router(config-pmap)# **class***class-name*
3. Router(config-pmap-c)# **priority percent** *percent*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-name* **Example:**	Specifies the name of the traffic policy to configure. Enters policy-map configuration mode.
Step 2	Router(config-pmap)# **class***class-name* **Example:**	Specifies the name of a predefined class included in the service policy. Enters policy-map class configuration mode.
Step 3	Router(config-pmap-c)# **priority percent** *percent* **Example:**	Reserves a priority queue with a specified percentage of available bandwidth for CBWFQ traffic. **Note** The traffic policy configured in this section is not yet attached to an interface. For information on attaching a traffic policy to an interface, see the "Applying QoS Features Using the MQC" module.

Configuring a Transmission Ring Limit on an ATM PVC

SUMMARY STEPS

1. Router(config)# **interface atm** *interface-name*
2. Router(config-if)# **atm pvc** *vcd-number vpi-number vci-number Encapsulation-type* **tx-ring-limit** *ring-limit*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface atm** *interface-name*	Specifies the name of the ATM interface to configure.

	Command or Action	Purpose
Step 2	Router(config-if)# **atm pvc** *vcd-number vpi-number vci-number Encapsulation-type* **tx-ring-limit** *ring-limit*	Specifies the ATM PVC to configure, the encapsulation type, and the transmission ring limit value.

Configuring a Transmission Ring Limit on an ATM Subinterface

SUMMARY STEPS

1. Router(config)# **interface atm** *subinterface name*
2. Router(config-subif)# **pvc** *pvc-name*
3. Router(config-if-atm-vc)# **tx-ring-limit** *ring-limit*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface atm** *subinterface name*	Specifies the name of the subinterface to configure.
Step 2	Router(config-subif)# **pvc** *pvc-name*	Specifies the name of the PVC to configure.
Step 3	Router(config-if-atm-vc)# **tx-ring-limit** *ring-limit*	Specifies the transmission ring limit value.

Verifying Distributed LLQ

Command	Purpose
`Router#` **show interfaces** `[`*interface-type interface-number*`]` **fair-queue**	Displays information and statistics about WFQ for a VIP-based interface.
`Router#` **show policy-map** *policy-map-name*	Displays the contents of a policy map, including the priority setting in a specific policy map.

Verifying a Transmission Ring Limit

Command	Purpose
`Router#` **show atm vc** *vc-name*	Displays the contents of a VC. The **show atm vc** command output will indicate the transmission ring limit value if the **tx-ring-limit** command is successfully enabled.

Monitoring and Maintaining Distributed LLQ

Command	Purpose
`Router#` **show interfaces** `[`*interface-type interface-number*`]` **fair-queue**	Displays information and statistics about WFQ for a VIP-based interface.
`Router#` **show policy-map** *policy-map-name*	Displays the contents of a traffic policy, including the priority setting in a specific policy map.
`Router#` **show policy interface** *interface-name*	Displays the configuration of all classes configured for all service policies on the specified interface. Displays if packets and bytes were discarded or dropped for the priority class in the service policy attached to the interface.
`Router#` **show atm vc** *vc-name*	Displays the contents of a VC. The **show atm vc** command output will indicate the transmission ring limit value if the **tx-ring-limit** command is successfully enabled.

Low Latency Queueing for Frame Relay Configuration Task List

Defining Class Maps

SUMMARY STEPS

1. Router(config)# **class-map***class-map-name*
2. Do one of the following:
   - Router(config-cmap)# **match access-group** {*a ccess-group*| *name access-group-name}*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **class-map***class-map-name*	Specifies the name of the class map to be created.
Step 2	Do one of the following: - Router(config-cmap)# **match access-group** {*access-group*| *name access-group-name}* **Example:** `Router(config-cmap)#` **match input-interface** *interface-name* **Example:** `Router(config-cmap)#` **match protocol** *protocol*	Specifies the name of the ACL against whose contents packets are checked to determine if they belong to the class. Specifies the name of the input interface used as a match criterion against which packets are checked to determine if they belong to the class. Specifies the name of the protocol used as a match criterion against which packets are checked to determine if they belong to the class.

Configuring Class Policy in the Policy Map

To configure a policy map and create class policies that make up the service policy, begin with the **policy-map** command to specify the policy map name. Then use one or more of the following commands to configure the policy for a standard class or the default class:

- **priority**
- **bandwidth**
- **queue-limit** or **random-detect**
- **fair-queue** (for class-default class only)

For each class that you define, you can use one or more of the commands listed to configure the class policy. For example, you might specify bandwidth for one class and both bandwidth and queue limit for another class.

The default class of the policy map (commonly known as the class-default class) is the class to which traffic is directed if that traffic does not satisfy the match criteria of the other classes defined in the policy map.

The class-default class is used to classify traffic that does not fall into one of the defined classes. Even though the class-default class is predefined when you create the policy map, you still have to configure it. If a default class is not configured, then traffic that does not match any of the configured classes is given best-effort treatment, which means that the network will deliver the traffic if it can, without any assurance of reliability, delay prevention, or throughput.

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 in a policy map must not exceed the minimum committed information rate (CIR) configured for the VC minus any bandwidth reserved by the **frame-relay voice bandwidth** and **frame-relay ip rtp priority** commands. If the minimum CIR is not configured, the bandwidth defaults to one half of the CIR. If all of the bandwidth is not allocated, the remaining bandwidth is allocated proportionally among the classes on the basis of their configured bandwidth.

To configure class policies in a policy map, perform the tasks described in the following sections.

Configuring Class Policy for a LLQ Priority Queue

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class** *class-name*
3. Router(config-pmap-c)# **priority** *bandwidth-kbps*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a class to be created and included in the service policy.
Step 3	Router(config-pmap-c)# **priority** *bandwidth-kbps*	Creates a strict priority class and specifies the amount of bandwidth, in kbps, to be assigned to the class.

Configuring Class Policy Using a Specified Bandwidth and WRED Packet Drop

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class** *class-name*
3. Router(config-pmap-c)# **bandwidth** *bandwidth-kbps*
4. Router(config-pmap-c)# **random-detect**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-pmap)# **class** *class-name*	Specifies the name of a class to be created and included in the service policy.
Step 3	Router(config-pmap-c)# **bandwidth** *bandwidth-kbps*	Specifies the amount of bandwidth to be assigned to the class, in kbps, or as a percentage of the available bandwidth. Bandwidth must

	Command or Action	Purpose
		be specified in kbps or as a percentage consistently across classes. (Bandwidth of the priority queue must be specified in kbps.)
Step 4	Router(config-pmap-c)# **random-detect**	Enables WRED.

Configuring the Class-Default Class Policy

SUMMARY STEPS

1. Router(config)# **policy-map** *policy-map*
2. Router(config-pmap)# **class class-default** *default-class-name*
3. Do one of the following:
   - Router(config-pmap-c)# **bandwidth** *bandwidth-kbps*
4. Router(config-pmap-c)# **queue-limit** *number-of-packets*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** *policy-map*	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-pmap)# **class class-default** *default-class-name*	Specifies the default class so that you can configure or modify its policy.
Step 3	Do one of the following: • Router(config-pmap-c)# **bandwidth** *bandwidth-kbps* **Example:** `Router(config-pmap-c)#` **fair-queue** `[`*number-of-dynamic-queues*`]`	Specifies the amount of bandwidth, in kbps, to be assigned to the class. Specifies the number of dynamic queues to be reserved for use by flow-based WFQ running on the default class. The number of dynamic queues is derived from the bandwidth of the interface.
Step 4	Router(config-pmap-c)# **queue-limit** *number-of-packets*	Specifies the maximum number of packets that the queue for the default class can accumulate.

Attaching the Service Policy and Enabling LLQ for Frame Relay

Command	Purpose
`Router(config-map-class)#` **service-policy output** *policy-map*	Attaches the specified service policy map to the output interface and enables LLQ for Frame Relay. **Note** When LLQ is enabled, all classes configured as part of the service policy map are installed in the fair queueing system.

Verifying Configuration of Policy Maps and Their Classes

Command	Purpose
`Router#` **show frame-relay pvc** *dlci*	Displays statistics about the PVC and the configuration of classes for the policy map on the specified DLCI.
`Router#` **show policy-map interface** *interface-name*	When FRTS is configured, displays the configuration of classes for all Frame Relay VC-level policy maps. When FRTS is not configured, displays the configuration of classes for the interface-level policy.
`Router#` **show policy-map interface** *interface-name* **dlci** *dlci*	When FRTS is configured, displays the configuration of classes for the policy map on the specified DLCI.

Monitoring and Maintaining LLQ for Frame Relay

For a list of commands that can be used to monitor LLQ for Frame Relay, see the previous section "Verifying Configuration of Policy Maps and Their Classes, on page 52."

Configuring Burst Size in LLQ Configuration Task List

Configuring the LLQ Bandwidth

Command	Purpose
`Router(config)#` **priority** *bandwidth*	Specifies the maximum amount of bandwidth, in kpbs, for the priority traffic.

Configuring the LLQ Burst Size

Command	Purpose
`Router(config)#` **priority** *bandwidth burst*	Specifies the burst size in bytes. The range is from 32 to 2 million.

Verifying the LLQ Burst Size

Command	Purpose
`Router#` **show policy-map**	Displays the configuration of all classes comprising the specified service policy map or all classes for all existing policy maps.
`Router#` **show policy-map interface**	Displays the configuration of classes configured for service polices on the specified interface or PVC.

Per-VC Hold Queue Support for ATM Adapters Configuration Task List

Configuring the per-VC Hold Queue on an ATM Adapter

Command	Purpose
`Router(config)#` **vc-hold-queue** *number-of-packets*	Specifies the number of packets contained in the per-VC hold queue. This can be a number from 5 to 1024.

Verifying the Configuration of the per-VC Hold Queue on an ATM Adapter

Command	Purpose
`Router#` **show queueing interface**	Displays the queueing statistics of an interface or VC.

Examples Flow-Based WFQ Configuration

The following example requests a fair queue with a congestive discard threshold of 64 messages, 512 dynamic queues, and 18 RSVP queues:

```
Router(config)# interface Serial 3/0
Router(config-if)# ip unnumbered Ethernet 0/
0
Router(config-if)# fair-queue 64 512 18
```

Examples DWFQ Configuration

Example Flow-Based DWFQ

The following example enables DWFQ on the HSSI interface 0/0/0:

```
Router(config)# interface Hssi0/0/0
Router(config-if)# description 45Mbps to R2
Router(config-if)# ip address 200.200.14.250 255.255.255.252
Router(config-if)# fair-queue
```

The following is sample output from the **show interfaces fair-queue** command for this configuration:

```
Router# show interfaces hssi 0/0/0 fair-queue
Hssi0/0/0 queue size 0
     packets output 35, drops 0
WFQ: global queue limit 401, local queue limit 200
```

Example QoS-Group-Based DWFQ

The following example configures QoS-group-based DWFQ. Committed access rate (CAR) policies are used to assign packets with an IP Precedence value of 2 to QoS group 2, and packets with an IP Precedence value of 6 are assigned to QoS group 6.

```
Router(config)# interface Hssi0/0/0
Router(config-if)# ip address 188.1.3.70 255.255.255.0
Router(config-if)# rate-limit output access-group rate-limit 6 155000000 2000000 8000000
conform-action set-qos-transmit 6 exceed-action drop
Router(config-if)# rate-limit output access-group rate-limit 2 155000000 2000000 8000000
conform-action set-qos-transmit 2 exceed-action drop
Router(config-if)# fair-queue qos-group
Router(config-if)# fair-queue qos-group 2 weight 10
Router(config-if)# fair-queue qos-group 2 limit 27
Router(config-if)# fair-queue qos-group 6 weight 30
Router(config-if)# fair-queue qos-group 6 limit 27
!
Router(config)# access-list rate-limit 2 2
Router(config)# access-list rate-limit 6 6
```

The following sample output shows how to view WFQ statistics using the **show interfaces fair-queue** command:

```
Router# show interfaces fair-queue
 Hssi0/0/0 queue size 0
        packets output 806232, drops 1
    WFQ: aggregate queue limit 54, individual queue limit 27
       max available buffers 54

        Class 0: weight 60 limit 27 qsize 0 packets output 654 drops 0
        Class 2: weight 10 limit 27 qsize 0 packets output 402789 drops 0
        Class 6: weight 30 limit 27 qsize 0 packets output 402789 drops 1
```

Example ToS-Based DWFQ

The following example configures type of service (ToS)-based DWFQ using the default parameters:

```
Router# configure terminal
Router(config)# interface Hssi0/0/0
Router(config-if)# fair-queue tos
Router(config-if)# end
```

The following is output of the **show running-config** command for the HSSI interface 0/0/0. Notice that the router automatically adds the default weights and limits for the ToS classes to the configuration.

```
interface Hssi0/0/0
 ip address 188.1.3.70 255.255.255.0
 fair-queue tos
 fair-queue tos 1 weight 20
 fair-queue tos 1 limit 27
 fair-queue tos 2 weight 30
 fair-queue tos 2 limit 27
 fair-queue tos 3 weight 40
 fair-queue tos 3 limit 27
```

The following sample output shows how to view DWFQ statistics using the **show interfaces fair-queue** command:

```
Router# show interfaces fair-queue
 Hssi0/0/0 queue size 0
        packets output 1417079, drops 2
 WFQ: aggregate queue limit 54, individual queue limit 27
    max available buffers 54

     Class 0: weight 10 limit 27 qsize 0 packets output 1150 drops 0
     Class 1: weight 20 limit 27 qsize 0 packets output 0 drops 0
     Class 2: weight 30 limit 27 qsize 0 packets output 775482 drops 1
     Class 3: weight 40 limit 27 qsize 0 packets output 0 drops 0
```

Examples CBWFQ Configuration

Example Class Map Configuration

In the following example, ACLs 101 and 102 are created. Next, two class maps are created and their match criteria are defined. For the first map class, called class1, the numbered 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. Packets are checked against the contents of these ACLs to determine if they belong to the class.

```
Router(config)# access-list 101 permit udp host 10.10.10.10 host 10.10.10.20 range 16384
20000
Router(config# access-list 102 permit udp host 10.10.10.10 host 10.10.10.20 range 53000
56000
Router(config)# class-map class1
Router(config-cmap)# match access-group 101
Router(config-cmap)# exit
Router(config-cmap)# class-map class2
Router(config-cmap)# match access-group 102
Router(config-cmap)# exit
```

Example Policy Creation

In the following example, a policy map called policy1 is defined to contain policy specification for the two classes, class1 and class2. The match criteria for these classes were defined in the previous "Example Class Map Configuration, on page 70" section.

For class1, the policy specifies the bandwidth allocation request and the maximum number of packets that the queue for this class can accumulate. For class2, the policy specifies only the bandwidth allocation request, so the default queue limit of 64 packets is assumed.

```
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth 3000

Router(config-pmap-c)# queue-limit 30
Router(config-pmap-c)# exit
Router(config-pmap)# class class2
Router(config-pmap-c)# bandwidth 2000

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

Example Policy Attachment to Interfaces

The following example shows how to attach an existing policy map. After you define a policy map, you can attach it to one or more interfaces to specify the service policy for those interfaces. Although you can assign the same policy map to multiple interfaces, each interface can have only one policy map attached at the input and one policy map attached at the output.

The policy map in this example was defined in the previous section, "Example Policy Creation, on page 71."

```
Router(config)# interface e1/1
Router(config-if)# service output policy1

Router(config-if)# exit
Router(config)# interface fa1/0/0
Router(config-if)# service output policy1

Router(config-if)# exit
```

Example CBWFQ Using WRED Packet Drop

In the following example, the class map called class1 is created and defined to use the input FastEthernet interface 0/1 as a match criterion to determine if packets belong to the class. Next, the policy map policy1 is defined to contain policy specification for class1, which is configured for WRED packet drop.

```
Router(config)# class-map class1
Router(config-cmap)# match input-interface FastEthernet0/1
!
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth 1000
Router(config-pmap-c)# random-detect
!
```

```
Router(config)# interface serial0/0
Router(config-if)# service-policy output policy1
!
```

Examples Display Service Policy Map Content

The following examples show how to display the contents of service policy maps. Four methods can be used to display the contents.

All Classes for a Specified Service Policy Map

The following example displays the contents of the service policy map called pol1:

```
Router# show policy-map po1
Policy Map po1
 Weighted Fair Queueing
    Class class1
       Bandwidth 937 (kbps) Max thresh 64 (packets)
    Class class2
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class3
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class4
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class5
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class6
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class7
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class8
         Bandwidth 937 (kbps)  Max thresh 64 (packets)
```

All Classes for All Service Policy Maps

The following example displays the contents of all policy maps on the router:

```
Router# show policy-map

Policy Map poH1
 Weighted Fair Queueing
    Class class1
       Bandwidth 937 (kbps) Max thresh 64 (packets)
    Class class2
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class3
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class4
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class5
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class6
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class7
        Bandwidth 937 (kbps)  Max thresh 64 (packets)
    Class class8
         Bandwidth 937 (kbps)  Max thresh 64 (packets)
Policy Map policy2
 Weighted Fair Queueing
    Class class1
       Bandwidth 300 (kbps) Max thresh 64 (packets)
    Class class2
        Bandwidth 300  (kbps)  Max thresh 64 (packets)
    Class class3
        Bandwidth 300 (kbps)  Max thresh 64 (packets)
    Class class4
        Bandwidth 300 (kbps)  Max thresh 64 (packets)
    Class class5
        Bandwidth 300 (kbps)  Max thresh 64 (packets)
    Class class6
        Bandwidth 300 (kbps)  Max thresh 64 (packets)
```

Specified Class for a Service Policy Map

The following example displays configurations for the class called class7 that belongs to the policy map called po1:

```
Router# show policy-map po1 class class7


Class class7
 Bandwidth 937 (kbps) Max Thresh 64 (packets)
```

All Classes for All Service Policy Maps on a Specified Interface

The following example displays configurations for classes on the output Ethernet interface 2/0. The numbers shown in parentheses are for use with the Management Information Base (MIB).

```
Router# show policy-map interface
 e2/0
Ethernet2/0
  Service-policy output:p1 (1057)
    Class-map:c1 (match-all) (1059/2)
      19 packets, 1140 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:ip precedence 0  (1063)
      Weighted Fair Queueing
        Output Queue:Conversation 265
        Bandwidth 10 (%) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
    Class-map:c2 (match-all) (1067/3)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:ip precedence 1  (1071)
      Weighted Fair Queueing
        Output Queue:Conversation 266
        Bandwidth 10 (%) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0

   Class-map:class-default (match-any) (1075/0)
      8 packets, 2620 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:any  (1079)
```

Examples Distributed CBWFQ Configuration

Example Traffic Class Configuration

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

```
Router(config)# class-map class1

Router(config-cmap)# match access-group 101
Router(config-cmap)# exit
Router(config)# class-map class2

Router(config-cmap)# match access-group 102
Router(config-cmap)# exit
```

Example Traffic Policy Creation

In the following example, a traffic policy called policy1 is defined to associate QoS features with the two traffic classes, class1 and class2. The match criteria for these traffic classes were defined in the previous "Example Class Map Configuration, on page 70" section.

For class1, the QoS policies include bandwidth allocation request and maximum packet count limit for the queue reserved for the traffic class. For class2, the policy specifies only a bandwidth allocation request, so the default queue limit of 64 packets is assumed.

```
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth 3000
Router(config-pmap-c)# queue-limit 30
Router(config-pmap)# exit
Router(config-pmap)# class class2
Router(config-pmap-c)# bandwidth 2000
Router(config-pmap)# exit
```

Example Traffic Policy Attachment to an Interface

The following example shows how to attach an existing traffic policy to an interface. After you define a traffic policy, you can attach it to one or more interfaces to specify a traffic policy for those interfaces. Although you can assign the same traffic policy to multiple interfaces, each interface can have only one traffic policy attached at the input and one policy map attached at the output at one time.

```
Router(config)# interface fe1/0/0

Router(config-if)# service output policy1

Router(config-if)# exit
```

Examples IP RTP Priority Configuration

Example CBWFQ Configuration

The following example first defines a CBWFQ configuration and then reserves a strict priority queue:

```
! The following commands define a class map:
Router(config)# class-map class1
Router(config-cmap)# match access-group 101
Router(config-cmap)# exit
! The following commands create and attach a policy map:
Router(config)# policy-map policy1
Router(config-pmap)# class class1
Router(config-pmap-c)# bandwidth 3000
Router(config-pmap-c)# queue-limit 30
Router(config-pmap-c)# random-detect
Router(config-pmap-c)# random-detect precedence 0 32 256 100
Router(config-pmap-c)# exit
Router(config)# interface Serial1
Router(config-if)# service-policy output policy1
! The following command reserves a strict priority queue:
Router(config-if)# ip rtp priority 16384 16383 40
```

The **queue-limit**and **random-detect**commands are optional commands for CBWFQ configurations. The **queue-limit**command is used for configuring tail drop limits for a class queue. The **random-detect**command is used for configuring RED drop limits for a class queue, similar to the **random-detect** command available on an interface.

Example Virtual Template Configuration

The following example configures a strict priority queue in a virtual template configuration with CBWFQ.

```
Router(config)# multilink virtual-template 1
Router(config)# interface virtual-template 1
Router(config-if)# ip address 172.16.1.1 255.255.255.0
Router(config-if)# no ip directed-broadcast
Router(config-if)# ip rtp priority 16384 16383 25
Router(config-if)# service-policy output policy1
Router(config-if)# ppp multilink
Router(config-if)# ppp multilink fragment-delay 20
Router(config-if)# ppp multilink interleave
Router(config-if)# end
Router(config)# interface Serial0/1
Router(config-if)# bandwidth 64
Router(config-if)# ip address 1.1.1.2 255.255.255.0
Router(config-if)# no ip directed-broadcast
Router(config-if)# encapsulation ppp
Router(config-if)# ppp multilink
Router(config-if)# end
```

Note To make the virtual access interface function properly, the **bandwidth** policy-map class configuration command should not be configured on the virtual template. It needs to be configured on the actual interface, as shown in the example.

Example Multilink Bundle Configuration

The following example configures a strict priority queue in a multilink bundle configuration with WFQ. The advantage to using multilink bundles is that you can specify different ip rtp priority parameters on different interfaces.

The following commands create multilink bundle 1, which is configured for a maximum ip rtp priority bandwidth of 200 kbps.

```
Router(config)# interface multilink 1
Router(config-if)# ip address 172.17.254.161 255.255.255.248
Router(config-if)# no ip directed-broadcast
Router(config-if)# ip rtp priority 16384 16383 200
Router(config-if)# no ip mroute-cache
Router(config-if)# fair-queue 64 256 0
Router(config-if)# ppp multilink
Router(config-if)# ppp multilink fragment-delay 20
Router(config-if)# ppp multilink interleave
```

The following commands create multilink bundle 2, which is configured for a maximum ip rtp priority bandwidth of 100 kbps:

```
Router(config)# interface multilink 2
Router(config-if)# ip address 172.17.254.162 255.255.255.248
Router(config-if)# no ip directed-broadcast
Router(config-if)# ip rtp priority 16384 16383 100
Router(config-if)# no ip mroute-cache
Router(config-if)# fair-queue 64 256 0
Router(config-if)# ppp multilink
Router(config-if)# ppp multilink fragment-delay 20
Router(config-if)# ppp multilink interleave
```

In the next part of the example, the **multilink-group** command configures serial interface 2/0 to be part of multilink bundle 1:

```
Router(config)# interface serial 2/0
Router(config-if)# bandwidth 256
Router(config-if)# no ip address
Router(config-if)# no ip directed-broadcast
Router(config-if)# encapsulation ppp
Router(config-if)# no ip mroute-cache
Router(config-if)# no fair-queue
Router(config-if)# clockrate 256000
Router(config-if)# ppp multilink
Router(config-if)# multilink-group 1
```

Next, serial interface 2/1 is configured to be part of multilink bundle 2.

```
Router(config)# interface serial 2/1
Router(config-if)# bandwidth 128
Router(config-if)# no ip address
Router(config-if)# no ip directed-broadcast
Router(config-if)# encapsulation ppp
Router(config-if)# no ip mroute-cache
Router(config-if)# no fair-queue
Router(config-if)# clockrate 128000
Router(config-if)# ppp multilink
Router(config-if)# multilink-group 2
```

Example Debug

The following example shows sample output from the **debug priority** command. In this example, 64 indicates the actual priority queue depth at the time the packet was dropped.

```
Router# debug priority
*Feb 28 16:46:05.659:WFQ:dropping a packet from the priority queue 64
*Feb 28 16:46:05.671:WFQ:dropping a packet from the priority queue 64
*Feb 28 16:46:05.679:WFQ:dropping a packet from the priority queue 64
*Feb 28 16:46:05.691:WFQ:dropping a packet from the priority queue 64
*Feb 28 16:46:05.699:WFQ:dropping a packet from the priority queue 64
*Feb 28 16:46:05.711:WFQ:dropping a packet from the priority queue 64
*Feb 28 16:46:05.719:WFQ:dropping a packet from the priority queue 64
```

Examples Frame Relay IP RTP Priority Configuration

Example Strict Priority Service to Matching RTP Packets

The following example first configures the Frame Relay map class called voip and then applies the map class to PVC 100 to provide strict priority service to matching RTP packets. In this example, RTP packets on PVC 100 with UDP ports in the range 16384 to 32764 will be matched and given strict priority service.

```
map-class frame-relay voip
 frame-relay cir 256000
 frame-relay bc 2560
 frame-relay be 600
 frame-relay mincir 256000
 no frame-relay adaptive-shaping
 frame-relay fair-queue
 frame-relay fragment 250
 frame-relay ip rtp priority 16384 16380 210
interface Serial5/0
 ip address 10.10.10.10 255.0.0.0
 no ip directed-broadcast
 encapsulation frame-relay
 no ip mroute-cache
 load-interval 30
 clockrate 1007616
 frame-relay traffic-shaping
 frame-relay interface-dlci 100
  class voip
 frame-relay ip rtp header-compression
 frame-relay intf-type dce
```

Examples Frame Relay PVC Interface PQ Configuration

This section provides configuration examples for Frame Relay PIPQ.

This example shows the configuration of four PVCs on serial interface 0. DLCI 100 is assigned high priority, DLCI 200 is assigned medium priority, DLCI 300 is assigned normal priority, and DLCI 400 is assigned low priority.

The following commands configure Frame Relay map classes with PVC priority levels:

```
Router(config)# map-class frame-relay HI
Router(config-map-class)# frame-relay interface-queue priority high
Router(config-map-class)# exit
Router(config)# map-class frame-relay MED
Router(config-map-class)# frame-relay interface-queue priority medium
Router(config-map-class)# exit
Router(config)# map-class frame-relay NORM
Router(config-map-class)# frame-relay interface-queue priority normal
Router(config-map-class)# exit
Router(config)# map-class frame-relay LOW
Router(config-map-class)# frame-relay interface-queue priority low
Router(config-map-class)# exit
```

The following commands enable Frame Relay encapsulation and Frame Relay PIPQ on serial interface 0. The sizes of the priority queues are set at a maximum of 20 packets for the high priority queue, 40 for the medium priority queue, 60 for the normal priority queue, and 80 for the low priority queue.

```
Router(config)# interface Serial0
Router(config-if)# encapsulation frame-relay
Router(config-if)# frame-relay interface-queue priority 20 40 60 80
```

The following commands assign priority to four PVCs by associating the DLCIs with the configured map classes:

```
Router(config-if)# frame-relay interface-dlci 100
Router(config-fr-dlci)# class HI
Router(config-fr-dlci)# exit
Router(config-if)# frame-relay interface-dlci 200
Router(config-fr-dlci)# class MED
Router(config-fr-dlci)# exit
Router(config-if)# frame-relay interface-dlci 300
Router(config-fr-dlci)# class NORM
Router(config-fr-dlci)# exit
Router(config-if)# frame-relay interface-dlci 400
Router(config-fr-dlci)# class LOW
Router(config-fr-dlci)# exit
```

Examples LLQ Configuration

Example ATM PVC Configuration

In the following example, a strict priority queue with a guaranteed allowed bandwidth of 50 kbps is reserved for traffic that is sent from the source address 10.10.10.10 to the destination address 10.10.10.20, in the range of ports 16384 through 20000 and 53000 through 56000.

First, the following commands configure access list 102 to match the desired voice traffic:

```
Router(config)# access-list 102 permit udp host 10.10.10.10 host 10.10.10.20 range 16384
20000
Router(config)# access-list 102 permit udp host 10.10.10.10 host 10.10.10.20 range 53000
56000
```

Next, the class map voice is defined, and the policy map called policy1 is created; a strict priority queue for the class voice is reserved, a bandwidth of 20 kbps is configured for the class bar, and the default class is configured for WFQ. The **service-policy** command then attaches the policy map to the PVC interface 0/102 on the subinterface atm1/0.2.

```
Router(config)# class-map voice
Router(config-cmap)# match access-group 102
Router(config)# policy-map policy1
Router(config-pmap)# class voice
Router(config-pmap-c)# priority 50
Router(config-pmap)# class bar
Router(config-pmap-c)# bandwidth 20
Router(config-pmap)# class class-default
Router(config-pmap-c)# fair-queue
Router(config)# interface atm1/0.2
Router(config-subif)# pvc 0/102
Router(config-subif-vc)# service-policy output policy1
```

Example Virtual Template Configuration

The following example configures a strict priority queue in a virtual template configuration with CBWFQ. Traffic on virtual template 1 that is matched by access list 102 will be directed to the strict priority queue.

First, the class map voice is defined, and the policy map called policy1 is created. A strict priority queue (with a guaranteed allowed bandwidth of 50 kbps) is reserved for the class called voice.

```
Router(config)# class-map voice
Router(config-cmap)# match access-group 102
Router(config)# policy-map policy1
Router(config-pmap)# class voice
Router(config-pmap-c)# priority 50
```

Next, the **service-policy** command attaches the policy map called policy1 to virtual template 1.

```
Router(config)# multilink virtual-template 1
Router(config)# interface virtual-template 1
Router(config-if)# ip address 172.16.1.1 255.255.255.0
Router(config-if)# no ip directed-broadcast
Router(config-if)# service-policy output policy1
Router(config-if)# ppp multilink
Router(config-if)# ppp multilink fragment-delay 20
Router(config-if)# ppp multilink interleave
Router(config-if)# end
Router(config)# interface serial 2/0
Router(config-if)# bandwidth 256
Router(config-if)# no ip address
Router(config-if)# no ip directed-broadcast
Router(config-if)# encapsulation ppp
Router(config-if)# no fair-queue
Router(config-if)# clockrate 256000
Router(config-if)# ppp multilink
```

Example Multilink Bundle Configuration

The following example configures a strict priority queue in a multilink bundle configuration with CBWFQ. Traffic on serial interface 2/0 that is matched by access list 102 will be directed to the strict priority queue. The advantage to using multilink bundles is that you can specify different **priority** parameters on different interfaces. To specify different **priority** parameters, you would configure two multilink bundles with different parameters.

First, the class map voice is defined, and the policy map called policy1 is created. A strict priority queue (with a guaranteed allowed bandwidth of 50 kbps) is reserved for the class called voice.

```
Router(config)# class-map voice
Router(config-cmap)# match access-group 102
Router(config)# policy-map policy1
Router(config-pmap)# class voice
Router(config-pmap-c)# priority 50
```

The following commands create multilink bundle 1. The policy map called policy1 is attached to the bundle by the **service-policy** command.

```
Router(config)# interface multilink 1
Router(config-if)# ip address 172.17.254.161 255.255.255.248
Router(config-if)# no ip directed-broadcast
Router(config-if)# no ip mroute-cache
Router(config-if)# service-policy output policy1
Router(config-if)# ppp multilink
Router(config-if)# ppp multilink fragment-delay 20
Router(config-if)# ppp multilink interleave
```

In the next part of the example, the **multilink-group** command configures serial interface 2/0 to be part of multilink bundle 1, which effectively directs traffic on serial interface 2/0 that is matched by access list 102 to the strict priority queue:

```
Router(config)# interface serial 2/0
Router(config-if)# bandwidth 256
Router(config-if)# no ip address
Router(config-if)# no ip directed-broadcast
Router(config-if)# encapsulation ppp
Router(config-if)# no fair-queue
Router(config-if)# clockrate 256000
Router(config-if)# ppp multilink
Router(config-if)# multilink-group 1
```

Examples Distributed LLQ Configuration

Example Enabling PQ for an Amount of Available Bandwidth on an ATM Subinterface

The **priority** command can be enabled on an ATM subinterface, and that subinterface must have only one enabled ATM PVC. This configuration provides a sufficient amount of ATM PVC support.

In the following example, a priority queue with a guaranteed allowed bandwidth of 50 kbps is reserved for traffic that is sent from the source address 10.10.10.10 to the destination address 10.10.10.20, in the range of ports 16384 through 20000 and 53000 through 56000.

First, the following commands configure access list 102 to match the desired voice traffic:

```
Router(config)# access-list 102 permit udp host 10.10.10.10 host 10.10.10.20 range 16384
20000
Router(config)# access-list 102 permit udp host 10.10.10.10 host 10.10.10.20 range 53000
56000
```

Next, the traffic class called voice is defined, and the policy map called policy1 is created; a priority queue for the class voice is reserved with a guaranteed allowed bandwidth of 50 kpbs and an allowable burst size of 60 bytes, a bandwidth of 20 kbps is configured for the class called bar, and the default class is configured for flow-based fair queuing. The **service-policy** command then attaches the policy map to the PVC interface 0/102 on the subinterface atm1/0.

```
Router(config)# class-map voice
Router(config-cmap)# match access-group 102
Router(config)# policy-map policy1
Router(config-pmap)# class voice
Router(config-pmap-c)# priority 50 60
Router(config-pmap)# class bar
Router(config-pmap-c)# bandwidth 20
Router(config-pmap)# class class-default
Router(config-pmap-c)# fair-queue
Router(config)# interface atm1/0
Router(config-subif)# pvc 0/102
Router(config-subif)# service-policy output policy1
```

Example Enabling PQ for a Percentage of Available Bandwidth on an ATM Subinterface

The **priority percent**command can be enabled on an ATM subinterface, and that subinterface must have only one enabled ATM PVC. This configuration provides a sufficient amount of ATM PVC support.

In the following example, a priority queue with a guaranteed allowed bandwidth percentage of 15 percent is reserved for traffic that is sent from the source address 10.10.10.10 to the destination address 10.10.10.20, in the range of ports 16384 through 20000 and 53000 through 56000.

First, the following commands configure access list 102 to match the desired voice traffic:

```
Router(config)# access-list 102 permit udp host 10.10.10.10 host 10.10.10.20 range 16384
20000
Router(config)# access-list 102 permit udp host 10.10.10.10 host 10.10.10.20 range 53000
56000
```

Next, the traffic class called voice is defined, and the policy map called policy1 is created; a priority queue for the class voice is reserved with a guaranteed allowed bandwidth percentage of 15 percent, a bandwidth percentage of 20 percent is configured for the class called bar, and the default class is configured for flow-based fair queueing. The **service-policy** command then attaches the policy map to the ATM subinterface 1/0.2.

```
Router(config)# class-map voice
Router(config-cmap)# match access-group 102
Router(config)# policy-map policy1
Router(config-pmap)# class voice
Router(config-pmap-c)# priority percent 15
Router(config-pmap)# class bar
Router(config-pmap-c)# bandwidth percent 20
Router(config-pmap)# class class-default
Router(config-pmap-c)# fair-queue
Router(config)# interface atm1/0.2
Router(config-subif)# service-policy output policy1
```

Example Limiting the Transmission Ring Limit on an ATM Interface

In the following example, the number of particles on the transmission ring of an ATM interface is limited to seven particles:

```
Router(config)# interface atm 1/0/0
Router(config-if)# atm pvc 32 0 32 tx-ring-limit 7
```

Example Limiting the Transmission Ring Limit on an ATM PVC Subinterface

In the following example, the number of particles on the transmission ring of an ATM PVC subinterface is limited to ten particles:

```
Router(config)#
interface ATM1/0/0.1 point-to-point
Router(config-subif)#
```

```
pvc 2/200
Router(config-if-atm-vc)#
tx-ring-limit 10
```

The **tx-ring-limit** command can be applied to several ATM PVC subinterfaces on a single interface. Every individual PVC can configure a transmission ring limit.

Examples LLQ for Frame Relay Configuration

The following example shows how to configure a PVC shaped to a 64K CIR with fragmentation. The shaping queue is configured with a class for voice, two data classes for IP precedence traffic, and a default class for best-effort traffic. WRED is used as the drop policy on one of the data classes.

The following commands define class maps and the match criteria for the class maps:

```
!
class-map voice
 match access-group 101
!
class-map immediate-data
 match access-group 102
!
class-map priority-data
 match access-group 103
!
access-list 101 permit udp any any range 16384 32767
access-list 102 permit ip any any precedence immediate
access-list 103 permit ip any any precedence priority
```

The following commands create and define a policy map called mypolicy:

```
!
policy-map mypolicy
 class voice
  priority 16
 class immediate-data
  bandwidth 32
  random-detect
 class priority-data
  bandwidth 16
 class class-default
  fair-queue 64
  queue-limit 20
```

The following commands enable Frame Relay fragmentation and attach the policy map to DLCI 100:

```
!
interface Serial1/0.1 point-to-point
 frame-relay interface-dlci 100
   class fragment
!
map-class frame-relay fragment
 frame-relay cir 64000
 frame-relay mincir 64000
 frame-relay bc 640
 frame-relay fragment 50
 service-policy output mypolicy
```

Examples Burst Size in LLQ Configuration

The following example configures the burst parameter to 1250 bytes for the class called Voice, which has an assigned bandwidth of 1000 kbps:

```
policy policy1
  class Voice
   priority 1000 1250
```

Examples Per-VC Hold Queue Support for ATM Adapters

The following example sets the per-VC hold queue to 55:

```
interface atm2/0.1
 pvc 1/101
  vc-hold-queue 55
```

CHAPTER 4

Low Latency Queueing with Priority Percentage Support

This feature allows you to configure bandwidth as a percentage within low latency queueing (LLQ). Specifically, you can designate a percentage of the bandwidth to be allocated to an entity (such as a physical interface, a shaped ATM permanent virtual circuit (PVC), or a shaped Frame Relay PVC to which a policy map is attached). Traffic associated with the policy map will then be given priority treatment.

This feature also allows you to specify the percentage of bandwidth to be allocated to nonpriority traffic classes. It modifies two existing commands--**bandwidth** and **priority**--and provides additional functionality to the way that bandwidth can be allocated using these two commands.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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.

Restrictions for LLQ with Priority Percentage Support

If the incoming high priority traffic exceeds the bandwidth percentage calculated by the **priority percent**command, and there is congestion in the network, the excess traffic is dropped. This is identical to the behavior demonstrated when the **priority**command uses bandwidth in kbps. In both cases, if the high priority traffic exceeds the bandwidth, and there is congestion in the network, excess traffic is dropped.

By default, when the **bandwidth percent**and **priority percent** commands are used to allocate bandwidth, the sum of the bandwidth percentage allocated to the high priority traffic and the bandwidth percentage allocated to the nonpriority traffic cannot exceed 75 percent of the total bandwidth available on the interface.

The remaining 25 percent of the total bandwidth available on the interface is kept in reserve for the unclassified traffic and routing traffic, if any, and proportionally divided among the defined traffic classes. To override the 75 percent limitation, use the **max-reserved bandwidth** command in interface configuration mode.

Note The **max-reserved bandwidth** command is intended for use on main interfaces only; it has no effect on virtual circuits (VCs) or ATM permanent virtual circuits (PVCs).

Information About LLQ with Priority Percentage Support

Benefits of LLQ with Priority Percentage Support

This feature allows the Cisco Software to accommodate networks with a large number of interfaces, all with differing bandwidths. This feature is useful when all of those interfaces with differing bandwidths need to be associated with a policy map that allocates proportional bandwidths to multiple classes.

Additionally, configuring bandwidth in percentages is most useful when the underlying link bandwidth is unknown or the relative class bandwidth distributions are known. For interfaces that have adaptive shaping rates (such as available bit rate [ABR] virtual circuits), CBWFQ can be configured by configuring class bandwidths in percentages.

Changes to the bandwidth Command for LLQ with Priority Percentage Support

This feature adds a new keyword to the **bandwidth** command--**remaining percent**. The feature also changes the functionality of the existing **percent** keyword. These changes result in the following commands for bandwidth: **bandwidth percent**and **bandwidth remaining percent**.

The **bandwidth percent** command configures bandwidth as an absolute percentage of the total bandwidth on the interface.

The **bandwidth remaining percent**command allows you to allocate bandwidth as a relative percentage of the total bandwidth available on the interface. This command allows you to specify the relative percentage of the bandwidth to be allocated to the classes of traffic. For instance, you can specify that 30 percent of the available bandwidth be allocated to class1, and 60 percent of the bandwidth be allocated to class2. Essentially, you are specifying the ratio of the bandwidth to be allocated to the traffic class. In this case, the ratio is 1 to 2 (30 percent allocated to class1 and 60 percent allocated to class2). The sum of the numbers used to indicate this ratio cannot exceed 100 percent. This way, you need not know the total amount of bandwidth available, just the relative percentage you want to allocate for each traffic class.

Each traffic class gets a minimum bandwidth as a relative percentage of the remaining bandwidth. The remaining bandwidth is the bandwidth available after the priority queue, if present, is given its required bandwidth, and after any Resource Reservation Protocol (RSVP) flows are given their requested bandwidth.

Because this is a relative bandwidth allocation, the packets for the traffic classes are given a proportionate weight only, and no admission control is performed to determine whether any bandwidth (in kbps) is actually available. The only error checking that is performed is to ensure that the total bandwidth percentages for the classes do not exceed 100 percent.

Changes to the priority Command for LLQ with Priority Percentage Support

This feature also adds the **percent** keyword to the **priority** command. The **priority percent** command indicates that the bandwidth will be allocated as a percentage of the total bandwidth of the interface. You can then specify the percentage (that is, a number from 1 to 100) to be allocated by using the *percentage* argument with the **priority percent**command.

Unlike the **bandwidth** command, the **priority** command provides a strict priority to the traffic class, which ensures low latency to high priority traffic classes.

Bandwidth Calculations in LLQ with Priority Percentage Support

When the **bandwidth** and **priority** commands calculate the total amount of bandwidth available on an entity, the following guidelines are invoked:

- If the entity is a physical interface, the total bandwidth is the bandwidth on the physical interface.
- If the entity is a shaped ATM PVC, the total bandwidth is calculated as follows:
    - For a variable bit rate (VBR) VC, the average shaping rate is used in the calculation.
    - For an available bit rate (ABR) VC, the minimum shaping rate is used in the calculation.
- If the entity is a shaped Frame Relay PVC, the total bandwidth is calculated as follows:
    - If a minimum acceptable committed information rate (minCIR) is not configured, the CIR divided by two is used in the calculation.
    - If a minimum acceptable CIR is configured, the minCIR setting is used in the calculation.

How to Configure LLQ with Priority Percentage Support

Specifying the Bandwidth Percentage

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map*
4. **class** {*class-name* | **class-default**}
5. **priority** {*bandwidth-kbps* | **percent** *percentage*}[*burst*]
6. **bandwidth** {*bandwidth-kbps* | **percent** *percentage* | **remaining percent** *percentage*}
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* **Example:** `Router(config)# policy-map policy1`	Specifies the name of the policy map to be created or modified. Enters policy-map configuration mode. • Enter the policy map name. Names can be a maximum of 40 alphanumeric characters.

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.

Step 5	**priority** {*bandwidth-kbps*	**percent** *percentage*}[*burst*]	Gives priority to a class of traffic belonging to the policy map. • Enter the priority percentage.

	Command or Action	Purpose

	Example:
	`Router(config-pmap-c)#`
	`priority percent 10`	

Step 6	**bandwidth** {*bandwidth-kbps* | **percent** *percentage* | **remaining percent** *percentage*}
	Example:
	`Router(config-pmap-c)#`
	`bandwidth percent 30`
		Specifies the bandwidth for a class of traffic belonging to the policy map.
		• Enter the bandwidth percentage.

Step 7	**end**
	Example:
	Example:
	`Router(config-pmap-c)#`
	`end`
		(Optional) Exits policy-map class configuration mode and returns to privileged EXEC mode.

Verifying the Bandwidth Percentage

SUMMARY STEPS

1. **enable**
2. **show policy-map** *policy-map*
3. **show policy-map** *policy-map* **class** *class-name*
4. **show policy-map interface** *type number*
5. **exit**

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 policy-map** *policy-map*
	Example:
	`Router#`
	`show policy-map policy1`
		(Optional) Displays the configuration of all classes for a specified service policy map or the configuration of all classes for all existing policy maps
		• Enter the name of the policy map whose complete configuration is to be displayed. The name can be a maximum of 40 alphanumeric characters.

Step 3	**show policy-map** *policy-map* **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 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 5	**exit** **Example:** `Router# exit`	(Optional) Exits privileged EXEC mode.

Configuration Examples for LLQ with Priority Percentage Support

Example Specifying the Bandwidth Percentage

The following example uses the **priority percent** command to specify a bandwidth percentage of 10 percent for the class called voice-percent. Then the **bandwidth remaining percent** command is used to specify a bandwidth percentage of 30 percent for the class called data1, and a bandwidth percentage of 20 percent for the class called data2.

```
Router> enable
Router# configure terminal
Router(config)# policy-map policy1
Router(config-pmap)# class voice-percent
Router(config-pmap-c)# priority percent 10
Router(config-pmap-c)# exit
Router(config-pmap)# class data1
Router(config-pmap-c)# bandwidth remaining percent 30
Router(config-pmap-c)# exit
Router(config-pmap)# class data2
Router(config-pmap-c)# bandwidth remaining percent 20
Router(config-pmap-c)# end
```

As a result of this configuration, 10 percent of the interface bandwidth is guaranteed for the class called voice-percent. The classes called data1 and data2 get 30 percent and 20 percent of the remaining bandwidth, respectively.

Example Mixing the Units of Bandwidth for Nonpriority Traffic

If a particular unit (that is, kbps or percentages) is used when specifying the bandwidth for a specific class of nonpriority traffic, the same bandwidth unit must be used when specifying the bandwidth for the other nonpriority classes in that policy map. The bandwidth units within the same policy map must be identical. However, the unit for the **priority** command in the priority class can be different from the bandwidth unit of the nonpriority class. The same configuration can contain multiple policy maps, however, which in turn can use different bandwidth units.

The following sample configuration contains three policy maps--policy1, policy2, and policy3. In the policy map called policy1 and the policy map called policy2, the bandwidth is specified by percentage. However, in the policy map called policy3, bandwidth is specified in kbps.

```
Router> enable
Router# configure terminal
Router(config)# policy-map policy1
Router(config-pmap)# class voice-percent
Router(config-pmap-c)# priority percent 10
Router(config-pmap-c)# exit
Router(config-pmap)# class data1
Router(config-pmap-c)# bandwidth percent 30
Router(config-pmap-c)# exit
Router(config-pmap)# class data2
Router(config-pmap-c)# bandwidth percent 20
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# policy-map policy2
Router(config-pmap)# class voice-percent
Router(config-pmap-c)# priority percent 10
Router(config-pmap-c)# exit
Router(config-pmap)# class data1
Router(config-pmap-c)# bandwidth remaining percent 30
Router(config-pmap-c)# exit
Router(config-pmap)# class data2
Router(config-pmap-c)# bandwidth remaining percent 20
Router(config-pmap-c)# exit
Router(config-pmap)# exit
Router(config)# policy-map policy3
Router(config-pmap)# class voice-percent
Router(config-pmap-c)# priority 500
Router(config-pmap-c)# exit
Router(config-pmap)# class data1
Router(config-pmap-c)# bandwidth 30
Router(config-pmap-c)# exit
Router(config-pmap)# class data2
Router(config-pmap-c)# bandwidth 20
Router(config-pmap-c)# end
```

Example Verifying the Bandwidth Percentage

The following sample output from the **show policy-map interface**command shows that 50 percent of the interface bandwidth is guaranteed for the class called class1, and 25 percent is guaranteed for the class called class2. The output displays the amount of bandwidth as both a percentage and a number of kbps.

```
Router# show policy-map interface
 serial3/2
 Serial3/2
  Service-policy output:policy1
    Class-map:class1 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:none
      Weighted Fair Queueing
        Output Queue:Conversation 265
        Bandwidth 50 (%)
        Bandwidth 772 (kbps) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
Class-map:class2 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:none
      Weighted Fair Queueing
        Output Queue:Conversation 266
        Bandwidth 25 (%)
        Bandwidth 386 (kbps) Max Threshold 64 (packets)
        (pkts matched/bytes matched) 0/0
        (depth/total drops/no-buffer drops) 0/0/0
    Class-map:class-default (match-any)
      0 packets, 0 bytes
      5 minute offered rate 0 bps, drop rate 0 bps
      Match:any
```

In this example, interface s3/2 has a total bandwidth of 1544 kbps. During periods of congestion, 50 percent (or 772 kbps) of the bandwidth is guaranteed to the class called class1, and 25 percent (or 386 kbps) of the link bandwidth is guaranteed to the class called class2.

Note The counters displayed for classes configured with **bandwidth** or **priority** after using the **show policy-map interface** command are updated only if congestion is present on the interface.

Additional References

The following sections provide references related to the Low Latency Queueing with Priority Percentage 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*
Congestion management concepts and related topics	"Congestion Management Overview" module
LLQ, bandwidth allocation	"Configuring Weighted Fair Queueing" module

Standards

Standards	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 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

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

Feature Information for LLQ with Priority Percentage 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 4: Feature Information for Low Latency Queueing with Priority Percentage Support

Feature Name	Releases	Feature Information
Low Latency Queueing with Priority Percentage Support	12.2(2)T 12.0(28)S 12.2(28)SB	This feature allows you to configure bandwidth as a percentage within low latency queueing (LLQ). Specifically, you can designate a percentage of the bandwidth to be allocated to an entity (such as a physical interface, a shaped ATM permanent virtual circuit (PVC), or a shaped Frame Relay PVC to which a policy map is attached). Traffic associated with the policy map will then be given priority treatment. In 12.2(2)T, this feature was introduced. In 12.0(28)S, this feature was integrated into Cisco IOS Release 12.0(28)S. In 12.2(28)SB, this feature was integrated into Cisco IOS Release 12.2(28)SB. The following commands were introduced or modified: **bandwidth (policy-map class)**, **priority**.

CHAPTER 5

Low Latency Queueing for IPSec Encryption Engines

Feature History

Release	Modification
12.2(13)T	This feature was introduced.
12.2(14)S	This feature was integrated into Cisco IOS Release 12.2(14)S.

This feature module describes the Low Latency Queueing (LLQ) for IPSec encryption engines feature in Cisco IOS Release 12.2(13)T and 12.2(14)S. It includes the following sections:

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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 Overview

Low Latency Queueing (LLQ) for IPSec encryption engines helps reduce packet latency by introducing the concept of queueing before crypto engines. Prior to this, the crypto processing engine gave data traffic and voice traffic equal status. Administrators now designate voice traffic as priority. Data packets arriving at a router interface are directed into a data packet inbound queue for crypto engine processing. This queue is called the best effort queue. Voice packets arriving on a router interface are directed into a priority packet inbound queue for crypto engine processing. This queue is called the priority queue. The crypto engine undertakes packet processing in a favorable ratio for voice packets. Voice packets are guaranteed a minimum processing bandwidth on the crypto engine.

Benefits

The Low Latency Queueing (LLQ) for IPSec encryption engines feature guarantees a certain level of crypto engine processing time for priority designated traffic.

Note On the Cisco 2600 platform, with the exception of the Cisco 2691 router, the CPU utilization maximizes out before the crypto engine becomes congested, so latency is not improved.

Better Voice Performance

Voice packets can be identified as priority, allowing the crypto engine to guarantee a certain percentage of processing bandwidth. This feature impacts the end user experience by assuring voice quality if voice traffic is directed onto a congested network.

Improved Latency and Jitters

Predictability is a critical component of network performance. The Low Latency Queueing (LLQ) for IPSec encryption engines feature delivers network traffic predictability relating to VPN. With this feature disabled, an end user employing an IP phone over VPN might experience jitter or latency, both symptoms of overall network latency and congestion. With this feature enabled, these undesirable characteristics are dissipated.

Restrictions

- No per-tunnel QoS policy. An interface QoS policy represents all tunnels.
- Assume the same IP precedence/DSCP marking for inbound and outbound voice packets.
- Assume the IP precedence/DSCP marking for voice packets are done at the source.
- Limited match criteria for voice traffic in the interface QoS policy.
- Assume call admission control is enforced within the enterprise.

- No strict error checking when aggregate policy's bandwidth exceeds crypto engine bandwidth. Only a warning is displayed but configuration is allowed.
- Assume voice packets are either all encrypted or unencrypted.

Related Features and Technologies

- CBWFQ
- Priority Queueing
- Weighted Fair Queueing

Related Documents

- Quality of Service Solutions Command Reference
- "Configuring Weighted Fair Queueing" module

Supported Platforms

12.2(14)S and higher

The LLQ for IPSec encryption engines feature is supported on the following platform:

- Cisco 7200 series

12.2(13)T

The LLQ for IPSec encryption engines feature is supported on all platforms using Cisco IOS Release 12.2(13)T or later, including:

- Cisco 2600 series
- Cisco 3600 series
- Cisco 7100 series
- Cisco 7200 series

Determining Platform Support Through Cisco Feature Navigator

Cisco IOS software is packaged in feature sets that are supported on specific platforms. To get updated information regarding platform support for this feature, access Cisco Feature Navigator. Cisco Feature Navigator dynamically updates the list of supported platforms as new platform support is added for the feature.

Cisco Feature Navigator is a web-based tool that enables you to quickly determine which Cisco IOS software images support a specific set of features and which features are supported in a specific Cisco IOS image. You can search by feature or release. Under the release section, you can compare releases side-by-side to display both the features unique to each software release and the features in common.

To access Cisco Feature Navigator, 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 http://www.cisco.com/register

Cisco Feature Navigator is updated regularly when major Cisco IOS software releases and technology releases occur. For the most current information, go to the Cisco Feature Navigator home page at the following URL:

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

Availability of Cisco IOS Software Images

Platform support for particular Cisco IOS software releases is dependent on the availability of the software images for those platforms. Software images for some platforms may be deferred, delayed, or changed without prior notice. For updated information about platform support and availability of software images for each Cisco IOS software release, see the online release notes or, if supported, Cisco Feature Navigator.

Supported Standards MIBs and RFCs

Standards

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

MIBs

- No new or modified standards are supported 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://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

- No new or modified RFCs are supported by this feature.

Prerequisites

To use this feature, you should be familiar with the following:

- Access control lists
- Bandwidth management
- CBWFQ

Configuration Tasks

Defining Class Maps

SUMMARY STEPS

1. Router(config)# **class-map**class-map-name
2. Do one of the following:
   - Router(config-cmap)# **match access-group** {*access-group* | *name access-group-name*}

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **class-map**class-map-name	Specifies the name of the class map to be created.
Step 2	Do one of the following: • Router(config-cmap)# **match access-group** {*access-group* | *name access-group-name*} **Example:** `Router(config-cmap)#` **match input-interface** *interface-name* **Example:** `or`	Specifies the name of the access control list (ACL) against whose contents packets are checked to determine if they belong to the class. Specifies the name of the input interface used as a match criterion against which packets are checked to determine if they belong to the class. Specifies the name of the protocol used as a match criterion against which packets are checked to determine if they belong to the class.

	Command or Action	Purpose
	Example: `Router(config-cmap)#` **match protocol** *protocol*	

Configuring Class Policy in the Policy Map

To configure a policy map and create class policies that make up the service policy, begin with the **policy-map** command to specify the policy map name. Then use one or more of the following commands to configure the policy for a standard class or the default class:

- **priority**
- **bandwidth**
- **queue-limit** or **random-detect**
- **fair-queue** (for class-default class only)

For each class that you define, you can use one or more of the commands listed to configure the class policy. For example, you might specify bandwidth for one class and both bandwidth and queue limit for another class.

The default class of the policy map (commonly known as the class-default class) is the class to which traffic is directed if that traffic does not satisfy the match criteria of the other classes defined in the policy map.

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 in a policy map must not exceed the minimum committed information rate (CIR) configured for the virtual circuit (VC) minus any bandwidth reserved by the **frame-relay voice bandwidth** and **frame-relay ip rtp priority** commands. If the minimum CIR is not configured, the bandwidth defaults to one half of the CIR. If all of the bandwidth is not allocated, the remaining bandwidth is allocated proportionally among the classes on the basis of their configured bandwidth.

To configure class policies in a policy map, perform the tasks described in the following sections. The task in the first section is required; the tasks in the remaining sections are optional.

Configuring Class Policy for a Priority Queue

SUMMARY STEPS

1. Router(config)# **policy-map** policy-map
2. Router(config-cmap)# **class** class-name
3. Router(config-pmap-c)# **priority** bandwidth-kbps

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** policy-map	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-cmap)# **class** class-name	Specifies the name of a class to be created and included in the service policy.
Step 3	Router(config-pmap-c)# **priority** bandwidth-kbps	Creates a strict priority class and specifies the amount of bandwidth, in kbps, to be assigned to the class.

Configuring Class Policy Using a Specified Bandwidth

SUMMARY STEPS

1. Router(config)# **policy-map** policy-map
2. Router(config-cmap)# **class** class-name
3. Router(config-pmap-c)# **bandwidth** bandwidth-kbps

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** policy-map	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-cmap)# **class** class-name	Specifies the name of a class to be created and included in the service policy.
Step 3	Router(config-pmap-c)# **bandwidth** bandwidth-kbps	Specifies the amount of bandwidth to be assigned to the class, in kbps, or as a percentage of the available bandwidth. Bandwidth must be specified in kbps or as a percentage consistently across classes. (Bandwidth of the priority queue must be specified in kbps.)

Configuring the Class-Default Class Policy

SUMMARY STEPS

1. Router(config)# **policy-map** policy-map
2. Router(config-cmap)# **class class-default** *default-class-name*
3. Router(config-pmap-c)# **bandwidth** bandwidth-kbps

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **policy-map** policy-map	Specifies the name of the policy map to be created or modified.
Step 2	Router(config-cmap)# **class class-default** *default-class-name*	Specifies the default class so that you can configure or modify its policy. **Note** The class-default class is used to classify traffic that does not fall into one of the defined classes. Even though the class-default class is predefined when you create the policy map, you still have to configure it. If a default class is not configured, then traffic that does not match any of the configured classes is given best-effort treatment, which means that the network will deliver the traffic if it can, without any assurance of reliability, delay prevention, or throughput.

Step 3	Router(config-pmap-c)# **bandwidth** bandwidth-kbps
	Example:
	Example:
	`or`
	Example:
	Example:
	`Router(config-pmap-c)#` **fair-queue** `[`*number-of-dynamic-queues*`]`
		Specifies the amount of bandwidth, in kbps, to be assigned to the class. Specifies the number of dynamic queues to be reserved for use by flow-based WFQ running on the default class. The number of dynamic queues is derived from the bandwidth of the interface.

Attaching the Service Policy

SUMMARY STEPS

1. Router(config)# **interface**type number
2. Router(config-if)# **service-policy output**policy-map

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface**type number	Specifies the interface using the LLQ for IPSec encryption engines.
Step 2	Router(config-if)# **service-policy output**policy-map	Attaches the specified service policy map to the output interface and enables LLQ for IPSec encryption engines.

Verifying Configuration of Policy Maps and Their Classes

SUMMARY STEPS

1. Router# **show frame-relay pvc dlci**
2. Router# **show policy-map interface** *interface-name*
3. Router# **show policy-map interface** *interface-name dlci* **dlci**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router# **show frame-relay pvc dlci**	Displays statistics about the PVC and the configuration of classes for the policy map on the specified data-link connection identifier (DLCI).
Step 2	Router# **show policy-map interface** *interface-name*	When LLQ is configured, displays the configuration of classes for all policy maps.
Step 3	Router# **show policy-map interface** *interface-name dlci* **dlci**	When LLQ is configured, displays the configuration of classes for the policy map on the specified DLCI.

Monitoring and Maintaining LLQ for IPSec Encryption Engines

SUMMARY STEPS

1. Router# **show crypto eng qos**

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router# **show crypto eng qos**	Displays quality of service queueing statistics for LLQ for IPSec encryption engines.

Configuration Examples

Example LLQ for IPSec Encryption Engines

In the following example, a strict priority queue with a guaranteed allowed bandwidth of 50 kbps is reserved for traffic that is sent from the source address 10.10.10.10 to the destination address 10.10.10.20, in the range of ports 16384 through 20000 and 53000 through 56000.

First, the following commands configure access list 102 to match the desired voice traffic:

```
Router(config)# access-list 102 permit udp host 10.10.10.10 host 10.10.10.20 range 16384
20000
Router(config)# access-list 102 permit udp host 10.10.10.10 host 10.10.10.20 range 53000
56000
```

Next, the class map voice is defined, and the policy map called policy1 is created; a strict priority queue for the class voice is reserved, a bandwidth of 20 kbps is configured for the class bar, and the default class is configured for WFQ. The service-policy command then attaches the policy map to the fas0/0.

```
Router(config)# class-map voice
Router(config-cmap)# match access-group 102
Router(config)# policy-map policy1
Router(config-pmap)# class voice
Router(config-pmap-c)# priority 50
Router(config-pmap)# class bar
Router(config-pmap-c)# bandwidth 20
Router(config-pmap)# class class-default
Router(config-pmap-c)# fair-queue
Router(config)# interface fastethernet0/0
Router(config-if)# service-policy output policy1
```

CHAPTER 6

Configuring Custom Queueing

This module describes the tasks for configuring QoS custom queueing (CQ) on a router.

Note CQ is not supported on any tunnels.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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.

Custom Queueing Configuration Task List

You must follow certain required, basic steps to enable CQ for your network. In addition, you can choose to assign packets to custom queues based on protocol type, interface where the packets enter the router, or other criteria you specify.

CQ allows a fairness not provided with priority queueing (PQ). With CQ, you can control the available bandwidth on an interface when it is unable to accommodate the aggregate traffic enqueued. Associated with each output queue is a configurable byte count, which specifies how many bytes of data should be delivered from the current queue by the system before the system moves on to the next queue. When a particular queue is being processed, packets are sent until the number of bytes sent exceeds the queue byte count defined by the **queue-list queue byte-count** command (see the following section Specifying the Maximum Size of the Custom Queues, on page 108), or until the queue is empty.

To configure CQ, perform the tasks described in the following sections.

Specifying the Maximum Size of the Custom Queues

Command	Purpose
`Router(config)#` **queue-list** *list-number* **queue** *queue-number* **limit** *limit-number*	Specifies the maximum number of packets allowed in each of the custom queues. The *limit-numbe r* argument specifies the number of packets that can be queued at any one time. The range is from 0 to 32767. The default is 20.
`Router(config)#` **queue-list** *list-number* **queue** *queue-number* **byte-count** *byte-count-number*	Designates the average number of bytes forwarded per queue. The *byte-count-number* argument specifies the average number of bytes the system allows to be delivered from a given queue during a particular cycle.

Assigning Packets to Custom Queues

Command	Purpose
`Router(config)#` **queue-list** *list-number* **protocol** *protocol-name queue-number queue-keyword keyword-value*	Establishes queueing priorities based on the protocol type. **Note** All protocols supported by Cisco are allowed. The *queue-keyword* variable provides additional options, including byte count, TCP service and port number assignments, and AppleTalk, IP, IPX, VINES, or XNS access list assignments. **Note** When you use multiple rules, remember that the system reads the **queue-list** commands in order of appearance.
`Router(config)#` **queue-list** *list-number* **interface** *interface-type interface-number queue-number*	Establishes CQ based on packets entering from a given interface.
`Router(config)#` **queue-list** *list-number* **default** *queue-number*	Assigns a queue number for those packets that do not match any other rule in the custom queue list.

Defining the Custom Queue List

SUMMARY STEPS

1. Router(config)# **interface***interface-type interface-number*
2. Router(config-if)# **custom-queue-list***list*

DETAILED STEPS

	Command or Action	Purpose
Step 1	Router(config)# **interface***interface-type interface-number*	Specifies the interface, and then enters interface configuration mode.
Step 2	Router(config-if)# **custom-queue-list***list*	Assigns a custom queue list to the interface. The list argument is any number from 1 to 16. There is no default assignment. **Note** Use the **custom-queue-list**command in place of the **priority-list** command. Only one queue list can be assigned per interface.

Monitoring Custom Queue Lists

Command	Purpose
`Router#` **show queue** *`interface-type interface-number`*	Displays the contents of packets inside a queue for a particular interface or virtual circuit (VC).
`Router#` **show queueing custom**	Displays the status of the CQ lists.
`Router#` **show interfaces** *`interface-type interface-number`*	Displays the current status of the custom output queues when CQ is enabled.

Custom Queueing Configuration Examples

Example Custom Queue List Defined

The following example illustrates how to assign custom queue list number 3 to serial interface 0:

```
interface serial 0
custom-queue-list 3
```

Examples Maximum Specified Size of the Custom Queues

The following example specifies the maximum number of packets allowed in each custom queue. The queue length of queue 10 is increased from the default 20 packets to 40 packets.

```
queue-list 3 queue 10 limit 40
```

The queue length limit is the maximum number of packets that can be enqueued at any time, with the range being from 0 to 32767 queue entries.

The following example decreases queue list 9 from the default byte count of 1500 to 1400 for queue number 10:

```
queue-list 9 queue 10 byte-count 1400
```

The byte count establishes the lowest number of bytes the system allows to be delivered from a given queue during a particular cycle.

Examples Packets Assigned to Custom Queues

The following examples assign packets to custom queues by either protocol type or interface type, and the default assignment for unmatched packets.

Protocol Type

The following example assigns traffic that matches IP access list 10 to queue number 1:

```
queue-list 1 protocol ip 1 list 10
```

The following example assigns Telnet packets to queue number 2:

```
queue-list 4 protocol ip 2 tcp 23
```

The following example assigns User Datagram Protocol (UDP) Domain Name Service (DNS) packets to queue number 3:

```
queue-list 4 protocol ip 3 udp 53
```

Interface Type

In this example, queue list 4 establishes queueing priorities for packets entering on serial interface 0. The queue number assigned is 10.

```
queue-list 4 interface serial 0 10
```

Default Queue

You can specify a default queue for packets that do not match other assignment rules. In this example, the default queue for list 10 is set to queue number 2:

```
queue-list 10 default 2
```

CHAPTER 7

Configuring Priority Queueing

This module describes the tasks for configuring priority queueing (PQ) on a device.

A priority list contains the definitions for a set of priority queues. The priority list specifies which queue a packet will be placed in and, optionally, the maximum length of the different queues.

In order to perform queueing using a priority list, you must assign the list to an interface. The same priority list can be applied to multiple interfaces. Alternatively, you can create many different priority policies to apply to different interfaces.

Assign packets to priority queues based on the following qualities:

- Protocol type
- Interface where the packets enter the device

You can specify multiple assignment rules. The **priority-list** commands are read in order of appearance until a matching protocol or interface type is found. When a match is found, the packet is assigned to the appropriate queue and the search ends. Packets that do not match other assignment rules are assigned to the default queue.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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.

How to Configure Priority Queueing

Defining the Priority List

Assigning Packets to Priority Queues

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **priority-list** *list-number* **protocol** *protocol-name* {**high** | **medium** | **normal** | **low**} *queue-keyword keyword-value*
4. **priority-list** *list-number* **interface** *interface-type interface-number* {**high** | **medium** | **normal**| **low**}
5. **priority-list** *list-number* **default** {**high** | **medium** | **normal** | **low**}
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	**priority-list** *list-number* **protocol** *protocol-name* {**high** | **medium** | **normal** | **low**} *queue-keyword keyword-value* **Example:** `Device(config)# priority-list 1 protocol ip high list 10`	Establishes queueing priorities based on the protocol type. **Note** All protocols supported by Cisco are allowed. The *queue-keyword* argument provides additional options including byte count, TCP service and port number assignments, and AppleTalk, IP, IPX, VINES, or XNS access list assignments. Refer to the **priority-list protocol** command syntax description in the *Cisco IOS Quality of Service Solutions Command Reference*.

	Command or Action	Purpose

Step 4	**priority-list** *list-number* **interface** *interface-type interface-number* {**high** | **medium** | **normal**| **low**}
	Example:
	`Device(config)# priority-list 3 interface ethernet 0 medium`
		Establishes queueing priorities for packets entering from a given interface.

Step 5	**priority-list** *list-number* **default** {**high** | **medium** | **normal** | **low**}
	Example:
	`Device(config)# priority-list 3 default high`
		Assigns a priority queue for those packets that do not match any other rule in the priority list.

Step 6	**end**
	Example:
	`Device(config)# end`
		Exits global configuration mode and returns to privileged EXEC mode.

Specifying the Maximum Size of the Priority Queues

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **priority-list**
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.

	Command or Action	Purpose
Step 3	**priority-list** **Example:** `Device(config)# policy-list`	Specifies the maximum number of packets allowed in each of the priority queues: • high-limit--20 • medium-limit--40 • normal-limit--60 • low-limit--80

Step 4	**end**
	Example:
	`Device(config)# end`
		(Optional) Exits global configuration mode.

Assigning the Priority List to an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-type interface-number*
4. **priority-group** *list-numbe*r
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** *interface-type interface-number* **Example:** `Device(config)# interface ethernet 0`	Specifies the interface, and then enters interface configuration mode.
Step 4	**priority-group** *list-numbe*r **Example:** `Device(config-if)# priority-group 3`	Assigns a priority list number to the interface.

Step 5	**end**
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Monitoring Priority Queueing Lists

SUMMARY STEPS

1. **enable**
2. **show queue interface-type interface-number**
3. **show queueing priority**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show queue interface-type interface-number** **Example:** `Device# show queue interface-type interface-number`	Displays the contents of packets inside a queue for a particular interface or VC.

	Command or Action	Purpose
Step 3	**show queueing priority** **Example:** `Device# show queueing priority`	Displays the status of the priority queueing lists.

Configuration Examples for Priority Queueing

Example: Priority Queueing Based on Protocol Type

The following example establishes queueing based on protocol type. The example assigns 1 as the arbitrary priority list number, specifies IP as the protocol type, and assigns a high-priority level to traffic that matches IP access list 10.

```
access-list 10 permit 239.1.1.0 0.0.0.255
priority-list 1 protocol ip high list 10
```

Example: Priority Queueing Based on Interface

The following example establishes queueing based on interface. The example sets any packet type entering on Ethernet interface 0 to a medium priority.

```
priority-list 3 interface ethernet 0 medium
```

Example: Maximum Specified Size of the Priority Queue

The following example changes the maximum number of packets in the high-priority queue to 10. The medium-limit, normal, and low-limit queue sizes remain at their default 40-, 60-, and 80-packet limits.

```
priority-list 4 queue-limit 10 40 60 80
```

Example: Priority List Assigned to an Interface

The following example assigns priority group list 4 to serial interface 0:

```
interface serial 0
  priority-group 4
```

Note The **priority-group** *list-number* command is not available on ATM interfaces that do not support fancy queueing.

Example: Priority Queueing Using Multiple Rules

When classifying a packet, the system searches the list of rules specified by **priority-list** commands for a matching protocol type. The following example specifies four rules:

- DECnet packets with a byte count less than 200 are assigned a medium-priority queue level.
- IP packets originating or destined to TCP port 23 are assigned a medium-priority queue level.
- IP packets originating or destined to User Datagram Protocol (UDP) port 53 are assigned a medium-priority queue level.
- All IP packets are assigned a high-priority queue level.

Remember that when using multiple rules for a single protocol, the system reads the priority settings in the order of appearance.

```
priority-list 4 protocol decnet medium lt 200
priority-list 4 protocol ip medium tcp 23
priority-list 4 protocol ip medium udp 53
priority-list 4 protocol ip high
```

Additional References for Configuring Priority Queueing

Related Documents

Related Topic	Document Title
Cisco 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 Configuring Priority Queueing

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 5: Feature Information for Configuring Priority Queueing

Feature Name	Releases	Feature Information
Priority Queueing (PQ)	11.2(1) 12.2(27)SBB 12.2(33)XNA Cisco IOS XE Release 3.2SE	The Priority Queueing (PQ) feature allows you to configure priority queueing on a device with the use of priority lists. The following commands were introduced or modified by this feature: **priority-group**, **priority list default**, **priority list interface**, **priority list protocol**, **priority list queue-limit**, **show queue**, **show queueing priority**.

CHAPTER 8

Per-Flow Admission

The Per-Flow Admission feature provides explicit controls to limit packet flow into a WAN edge in order to protect already admitted flows on the routing/WAN edge.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table.

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.

Prerequisites for Per-Flow Admission

A class must have bandwidth or priority defined before configuring the Per-Flow Admission feature.

Restrictions for Per-Flow Admission

Per-flow admission is currently supported only on Ethernet and serial interfaces, and Dynamic Multipoint Virtual Private Network (DMVPN) tunnels.

Information About Per-Flow Admission

Overview of Per-Flow Admission

Application (mainly voice and video) quality drops when they are connected from a branch to head quarters and data centers over a WAN because the WAN interface bandwidth is limited and always comes at a premium cost. There are no well-defined controls to restrict flows through a WAN link and no explicit controls to limit the flows to protect already admitted flows. This limitation leads to quality degradation of already admitted flows.

The Per-Flow Admission feature allows operators to understand the number of flows that can be accommodated into an interface without quality degradation. In most deployments, the N+1st flow affects the quality of all existing valid first N flows. The Per-Flow Admission feature enables nodes to automatically learn about flows and their bandwidth as they get accommodated into the interface where bandwidth is at a premium. The network node accommodates only flows that the interface can handle, and it drops flows thereafter.

Benefits of Per-Flow Admission

The following are benefits of integrating the Per-Flow Admission feature to Quality of Service (QoS):

- Makes QoS networks more predictable and robust.
- Requires no end-to-end coordination because per-flow admission is a per-hop decision and each hop makes decision independently.
- Does not require the source to predict the flow rate.
- Ensures a higher probability of getting a reservation in the network.
- Works well with rate adaption because certain parts of the flow may be elastic.
- Promotes better selection of admitted traffic.
- Works at the IP layer.
- Works transparently with other network technologies such as Network Address Translation (NAT).
- Does not allow the source to hog the network.
- Provides benefits for certain endpoints by selecting only certain parts of the flow as admitted.

How to Configure Per-Flow Admission

Configuring a Class Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **metadata flow**
4. **class-map** [**match-all** | **match-any**] *class-map-name*
5. **match metadata cac status** {**admitted** | **un-admitted**}
6. exit
7. **class-map** [**match-all** | **match-any**] *class-map-name*
8. **match dscp** *dscp-value*
9. **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	**metadata flow** **Example:** `Device(config)# metadata flow`	Enables metadata on all interfaces.

Step 4	**class-map** [**match-all** | **match-any**] *class-map-name*
	Example:
	`Device(config)# class-map match-all admitted`
		Creates a class map for matching traffic to a specified class, and enters class-map configuration mode.
		• Enter the class map name.

	Command or Action	Purpose
Step 5	**match metadata cac status** {**admitted** | **un-admitted**} **Example:** `Device(config-cmap)# match metadata cac status admitted`	Creates a filter to tag a flow as either admitted or non-admitted.
Step 6	exit **Example:** `Device(config-cmap)# exit`	Exits the class-map configuration mode and returns to global configuration mode.

Step 7	**class-map** [**match-all** | **match-any**] *class-map-name*
	Example:
	`Device(config-cmap)# class-map match-all af4`
		Creates a class map to be used for matching traffic to a specified class.
		• Enter the class map name.

Step 8	**match dscp** *dscp-value* **Example:** `Device(config-cmap)# match dscp af41 af42 af43`	Identifies a specific IP Differentiated Services Code Point (DSCP) value as a match criterion.

Step 9	**end**
	Example:
	`Device(config-cmap)#end`
		Exits class-map configuration mode and returns to privilged EXEC mode.

Configuring a Child Policy Map

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **set dscp** *dscp-value*
6. **class** {*class-name* | **class-default**}
7. **set dscp** *dscp-value*
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	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map child`	Creates a policy map using the specified name and enters policy-map configuration mode. - Enter the name of the policy map that you want to create.

Step 4	**class** {*class-name* | **class-default**}
	Example:
	`Device(config-pmap)# class admitted`
		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.

Step 5	**set dscp** *dscp-value* **Example:** `Device(config-pmap-c)# set dscp af41`	Sets the differentiated services code point (DSCP) value in the type of service (ToS) byte and assigns higher priority to admitted traffic by marking up the admitted flow and marking down the un-admitted flow. - Enter the DSCP value.

Step 6	**class** {*class-name* | **class-default**}
	Example:
	`Device(config-pmap-c)# class un-admitted`
		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.
		- Enter the name of the class or enter the **class-default** keyword.
		This class will be matched against the **match metadata cac status un-admitted** command.

Step 7	**set dscp** *dscp-value* **Example:** `Device(config-pmap-c)# set dscp af42`	Sets the DSCP value in the ToS byte. Sets higher priority to admitted traffic by marking up the admitted flow and marking down the un-admitted flow. - Enter the DSCP value.

Step 8	**end**
	Example:
	`Device(config-pmap-c)# end`
		Exits policy-map class configuration mode and returns to privileged EXEC mode.

Configuring Per-Flow Admission for a Class

Before You Begin

A class must have bandwidth or priority defined before configuring per-flow admission.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **bandwidth** {*kilobits* | **percent** *percentage*}
6. **admit cac local**
7. **rate** {*kbps* | **percent** *percentage*}
8. **flow rate fixed** *kbps flow-bit-rate*
9. **flow idle-timeout** *timeout-value*
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	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map test`	Creates a policy map using the specified name and enters policy-map configuration mode. • Enter the name of the policy map that you want to create.

Step 4	**class** {*class-name*	**class-default**}
		Example:
		`Device(config-pmap)# class af4`
		Note To divide packets into admitted and un-admitted buckets, you must assign the policy map created earlier, under the **class** command that is defined here as a child policy.
			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.

	Command or Action	Purpose
	Note To give preference to admitted packets, enable the weighted tail drop feature.	

Step 5	**bandwidth** {*kilobits* | **percent** *percentage*}
	Example:
	`Device(config-pmap-c)# bandwidth 200`
		Specifies the bandwidth for a class of traffic belonging to the policy map.
		- Enter the bandwidth in kbps.

Step 6	**admit cac local** **Example:** `Device(config-pmap-c)# admit cac local`	Enables per-flow admission for this class and enters per-flow admission configuration mode.

Step 7	**rate** {*kbps* | **percent** *percentage*}
	Example:
	`Device(config-pmap-admit-cac)# rate percent 80`
		Configures the size of the bandwidth pool in kbps or as a percentage of output class bandwidth.

Step 8	**flow rate fixed** *kbps flow-bit-rate* **Example:** `Device(config-pmap-admit-cac)# flow rate fixed 100`	Specifies how much bandwidth to allocate for each flow.
Step 9	**flow idle-timeout** *timeout-value* **Example:** `Device(config-pmap-admit-cac)#flow idle-timeout 50`	Sets the timeout period for the flow in seconds.

Step 10	**end**
	Example:
	`Device(config-pmap-admit-cac)# end`
		Exits per-flow admission configuration mode and returns to privileged EXEC mode.

Attaching a Per-Flow Admission Policy to an Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **service-policy** *policy-map*
6. **end**
7. **configure terminal**
8. **interface** *type number*
9. **bandwidth** *kilobits*
10. **ip address** *ip-address mask*
11. **load-interval** *seconds*
12. **service-policy output** *policy-map-name*
13. **no shutdown**
14. 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 test`	Creates a policy map using the specified name and enters policy-map configuration mode. • Enter the name of the policy map that you want to create.

Step 4	**class** {*class-name* | **class-default**}
	Example:
	`Device(config-pmap)# class af4`
		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.

	Command or Action	Purpose
Step 5	**service-policy** *policy-map* **Example:** `Device(config-pmap-c)# service-policy child`	Attaches the policy map to a class.

Step 6	**end**
	Example:
	`Device(config-pmap-c)# end`
		Exits policy-map class configuration mode and returns to privileged EXEC mode.

Step 7	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 8	**interface** *type number* **Example:** `Device(config)# interface Serial2/0`	Configures the specified interface and enters interface configuration mode. • Enter the interface type and number.
Step 9	**bandwidth** *kilobits* **Example:** `Device(config-if)# bandwidth 384`	Sets a bandwidth value for the interface. • Enter the bandwidth value in kbps.
Step 10	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 10.10.100.1 255.255.255.0`	Sets an IP address for an interface.
Step 11	**load-interval** *seconds* **Example:** `Device(config-if)# load-interval 30`	Specifies the interval for load calculation of an interface.
Step 12	**service-policy output** *policy-map-name* **Example:** `Device(config-if)# service-policy output test`	Attaches a policy map to an interface.
Step 13	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Enables the interface.

	Command or Action	Purpose

Step 14	end
	Example:
	`Device(config-if)# end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Verifying Per-flow Admission

SUMMARY STEPS

1. **enable**
2. **show policy-map interface** *interface-name*

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 interface** *interface-name* **Example:** `Device# show policy-map interface serial2/0`	Displays the configuration of all classes configured for all service policies on the specified interface. • Enter the name of the policy map whose complete configuration is to be displayed.

Configuration Examples for Per-Flow Admission

Example: Configuring a Class Map

```
Device> enable
Device# configure terminal
Device(config)# metadata flow
Device(config)# class-map match-all admitted
Device(config-cmap)# match metadata cac status admitted
Device(config-cmap)# class-map match-all af4
Device(config-cmap)# match dscp af41 af42 af43
Device(config-cmap)# end
```

Example: Configuring a Policy Map

```
Device> enable
Device# configure terminal
Device(config)# policy-map child
Device(config-pmap)# class admitted
Device(config-pmap-c)# set dscp af41
Device(config-pmap-c)# class class-default
Device(config-pmap-c)# set dscp af42
Device(config-pmap-c)# end
```

Example: Configuring Per-Flow Admission for a Class

```
Device> enable
Device# configure terminal
Device(config)# policy-map test
Device(config-pmap)# class af4
Device(config-pmap-c)# bandwidth 200
Device(config-pmap-c)# admit cac local
Device(config-pmap-admit-cac)# rate percent 80
Device(config-pmap-admit-cac)# flow rate fixed 100
Device(config-pmap-admit-cac)# flow idle-timeout 50Device(config-pmap-c)# exit
```

Example: Attaching a Per-Flow Admission Policy to an Interface

```
Device> enable
Device# configure terminal
Device(config-pmap-c)# service-policy child
Device(config-pmap-c)# end
Device# configure terminal
Device(config)# interface Serial2/0
Device(config-if)# bandwidth 384
Device(config-if)# ip address 10.10.100.1 255.255.255.0
Device(config-if)# load-interval 30
Device(config-if)# service-policy output test
Device(config-if)# no shutdown
Device(config-if)# end
```

Example: Verifying Per-Flow Admission

```
Device# show policy-map interface

 Serial2/0

  Service-policy output: test

    Class-map: af4 (match-all)
```

```
      269 packets, 336250 bytes
      30 second offered rate 90000 bps, drop rate 13000 bps
      Match:  dscp af41 (34) af42 (36) af43 (38)
      Queueing
      queue limit 100 ms/ 2500 bytes
 queue-limit dscp 34 150 ms/ 3750 bytes
       (pkts output/bytes output) 179/223750
       (pkt drops/byte drops) 0/0

      (queue depth/total drops/no-buffer drops) 2500/39/0
      (pkts output/bytes output) 230/287500
      bandwidth 200 kbps

      cac local rate 200 kbps, reserved 200 kbps
      flow rate fixed 100 kbps
      flow idle-timeout 5 sec
      All flows:
        Number of admitted flows: [2]
        Number of non-admitted flows: [1]

      Service-policy : child

        Class-map: admitted (match-all)
          178 packets, 222500 bytes
          30 second offered rate 60000 bps, drop rate 0000 bps
          Match:  metadata cac status admitted
          QoS Set
            dscp af41
              Packets marked 194

        Class-map: unadmitted (match-all)
          88 packets, 110000 bytes
          30 second offered rate 30000 bps, drop rate 0000 bps
          Match:  metadata cac status un-admitted
          QoS Set
            dscp af42
              Packets marked 96

        Class-map: class-default (match-any)
          3 packets, 3750 bytes
          30 second offered rate 1000 bps, drop rate 0000 bps
          Match: any

    Class-map: class-default (match-any)
      181 packets, 115396 bytes
      30 second offered rate 31000 bps, drop rate 0000 bps
      Match: any

      queue limit 64 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 181/115396
```

Additional References for Per-Flow Admission

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 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 Per-Flow Admission

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 6: Feature Information for Per-Flow Admission

Feature Name	Releases	Feature Information
Per-Flow Admission	15.4(2)T	The Per-Flow Admission feature provides explicit controls to limit flows in order to protect already admitted flows on the routing/WAN edge. The following commands were introduced by this feature: **admit cac local**, **flow idle-timeout**, **flow rate fixed**, **rate**.

CHAPTER 9

Remote Per-Flow Admission

The Remote Per-Flow Admission (PFA) feature enables the nodes to check for the available bandwidth at the remote side interface and the local side interface before admitting a flow. This ensures the quality of admitted flows (calls).

Information About Remote PFA

Remote PFA

In most deployments, the N+1st flow affects the quality of all existing valid first N flows. The Per-Flow Admission feature enables nodes to automatically learn about flows and their bandwidth as they get accommodated into the local interface side. The network node accommodates only flows that the interface can handle, and it throttles or drops flows thereafter. Thus flows are admitted on the local interface side, on first-come first-served basis, without knowing if the flow can be admitted on the remote interface side. This may cause the forthcoming flows to be rejected on the remote interface side, although flows can be admitted on the local interface side.

The Remote-PFA feature enables the nodes to learn about the available bandwidth at the remote interface side as well as the local interface side before admitting a flow. If a flow can be accommodated on both the remote and local side interfaces, the node adjusts the bandwidth availability for the next flow on the remote as well as the local side interfaces.

Prerequisites for Remote PFA

- Before you configure the Remote PFA feature, you must configure the Per-Flow Admission (local) feature.

Restrictions for Remote PFA

- The Remote PFA feature is currently supported only with Dynamic Multipoint Virtual Private Network (DMVPN) tunnels.
- The Remote PFA feature is currently supported only on global Virtual Routing and Forwarding (VRF) environment.

How to Configure Remote PFA

Configuring Sender (Output) Side Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. Execute one of the following:
   - **bandwidth** {*kilobits* | **percent** *percentage*}
   - **priority** *bandwidth-kbps*
6. **admit cac local**
7. **rate** {*kbps* | **percent** *percentage*}
8. **trigger remote signal**
9. **flow rate fixed** *flow-bit-rate*
10. **flow idle-timeout** *timeout-value*
11. **end**

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	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map child`	Creates a policy map using the specified name and enters policy-map configuration mode. • Enter the name of the policy map that you want to create. **Note** On the sender (output) side, the policy-map is applied to the multiple Generic Routing Encapsulation (mGRE) tunnel group using **nhrp map group** command.

Step 4	**class** {*class-name* | **class-default**}
	Example:
	`Device(config-pmap)# class admitted`
		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.

Step 5	Execute one of the following:
	• **bandwidth** {*kilobits* | **percent** *percentage*}
	• **priority** *bandwidth-kbps*
	Example:
	`Device(config-pmap-c)# bandwidth 200`
	Example:
	`Device(config-pmap-c)# priority 48`
		Specifies the bandwidth for a class of traffic belonging to the policy map.
		• Enter the bandwidth in kbps.

Step 6	**admit cac local** **Example:** `Device(config-pmap-c)# admit cac local`	Enables per-flow admission for this class and enters per-flow admission configuration mode.

Step 7	**rate** {*kbps* | **percent** *percentage*}
	Example:
	`Device(config-pmap-admit-cac)# rate percent 100`
		(Optional) Configures the size of the bandwidth pool in kbps or as a percentage of output class bandwidth.

Step 8	**trigger remote signal** **Example:** `Device(config-pmap-admit-cac)# trigger remote signal`	Triggers remote message to the peer once flows are seen in the configured class.

	Command or Action	Purpose
Step 9	**flow rate fixed** *flow-bit-rate* **Example:** `Device(config-pmap-admit-cac)# flow rate fixed 100`	(Optional) Specifies how much bandwidth to allocate for each flow. **Note** If you do not specify the flow rate, the rate is calculated dynamically per flow.
Step 10	**flow idle-timeout** *timeout-value* **Example:** `Device(config-pmap-admit-cac)#flow idle-timeout 50`	(Optional) Sets the timeout period for the flow in seconds. **Note** The default idle-timeout is 10 sec.

Step 11	**end**
	Example:
	`Device(config-pmap-admit-cac)# end`
		Exits per-flow admission configuration mode and returns to privileged EXEC mode.

Configuring Receiver (Input) Side Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name* | **class-default**}
5. **admit cac local**
6. **rate** {*kbps* | **percent** *percentage*}
7. **flow rate remote**
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	**policy-map** *policy-map-name* **Example:** `Device(config)# policy-map child`	Creates a policy map using the specified name and enters policy-map configuration mode. • Enter the name of the policy map that you want to create. **Note** On the receiver (input) side, the policy-map is attached to the tunnel aggregation point (physical interface).

Step 4	**class** {*class-name* | **class-default**}
	Example:
	`Device(config-pmap)# class admitted`
		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.

Step 5	**admit cac local** **Example:** `Device(config-pmap-c)# admit cac local`	Enables per-flow admission for this class and enters per-flow admission configuration mode.

Step 6	**rate** {*kbps* | **percent** *percentage*}
	Example:
	`Device(config-pmap-admit-cac)# rate 200`
		Configures the size of the bandwidth pool in kbps.
		Note Bandwidth pool on the remote side must be configured for every Differentiated Services Code Point (DSCP) class.

Step 7	**flow rate remote** **Example:** `Device(config-pmap-admit-cac)# flow rate remote`	Notifies the flow rate at the peer side to the local side.

Step 8	**end**
	Example:
	`Device(config-pmap-admit-cac)# end`
		Exits per-flow admission configuration mode and returns to privileged EXEC mode.

Verifying Remote PFA

SUMMARY STEPS

1. **enable**
2. **show policy-map multipoint** [**tunnel** *tunnel-interface-number*]
3. **show policy-map interface** *interface-type slot/subslot/port*

DETAILED STEPS

Step 1 **enable**

Example:

```
Device> enable
```

Enables the privileged EXEC mode.

- Enter your password if prompted.

Step 2 **show policy-map multipoint** [**tunnel** *tunnel-interface-number*]

Example:

```
Device# show policy-map multipoint Tunnel1

Interface Tunnel1 <--> 12.1.1.1

  Service-policy output: parent

    Class-map: class-default (match-any)
      7 packets, 1155 bytes
      30 second offered rate 0000 bps, drop rate 0000 bps
      Match: any
      Queueing
      queue limit 1750 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 7/1610
      shape (average) cir 7000000, bc 175000, be 175000
      target shape rate 7000000

      Service-policy : test

        Class-map: cac (match-all)
          0 packets, 0 bytes
          30 second offered rate 0000 bps, drop rate 0000 bps
          Match:  dscp af41 (34)
          Queueing
          queue limit 50 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 0/0
          bandwidth 200 kbps
          cac local rate 200 kbps, reserved 0 kbps
          remote signal: enabled
          flow rate fixed 100 kbps


        Class-map: class-default (match-any)
          7 packets, 1155 bytes
          30 second offered rate 0000 bps, drop rate 0000 bps
          Match: any

          queue limit 1700 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 1/230
```

Displays QoS policy details applied to multipoint tunnels, at the sender (output) side.

Step 3 **show policy-map interface** *interface-type slot/subslot/port*

Example:

```
Device# show policy-map interface Ethernet0/0

  Service-policy input: test

    Class-map: af4 (match-any)
      0 packets, 0 bytes
      30 second offered rate 0000 bps, drop rate 0000 bps
      Match:  dscp af41 (34)
        0 packets, 0 bytes
        30 second rate 0 bps
      Match:  dscp af41 (34) af42 (36) af43 (38)
        0 packets, 0 bytes
        30 second rate 0 bps
      cac local rate 200 kbps, reserved 0 kbps
      flow rate remote


    Class-map: class-default (match-any)
      4 packets, 356 bytes
      30 second offered rate 0000 bps, drop rate 0000 bps
      Match: any
```

Displays the statistics status and the configured policy maps on the Ethernet interface, at the receiver (input) side.

Configuration Examples for Remote PFA

Example: Configuring Sender (Output) Side Interface

The following example shows how to configure the sender (output) side interface.

```
policy-map child
 class admitted
   bandwidth 200
   admit cac local
      rate percent 100
      trigger remote signal
      flow rate fixed 100
      flow idle-timeout 5
```

Example: Configuring Receiver (Input) Side Interface

The following example shows how to configure a receiver (input) side interface.

```
policy-map child
 class admitted
   admit cac local
     rate 200
       flow rate remote
```

Troubleshooting Remote PFA

SUMMARY STEPS

1. Execute one of the following debug commands:
    - **debug qos cac local flow**
    - **debug qos cac remote flow**

DETAILED STEPS

Execute one of the following debug commands:

- **debug qos cac local flow**
- **debug qos cac remote flow**

Example:

```
Device# debug qos cac remote flow

QoS CAC flow display debugging is on
Device#
Device#show policy-map multipoint

Interface Tunnel1 <--> 12.1.1.1

  Service-policy output: parent

    Class-map: class-default (match-any)
      402255 packets, 31514364 bytes
      30 second offered rate 61000 bps, drop rate 0000 bps
      Match: any
      Queueing
      queue limit 1750 packets
      (queue depth/total drops/no-buffer drops) 0/0/0
      (pkts output/bytes output) 402255/54048186
      shape (average) cir 7000000, bc 175000, be 175000
      target shape rate 7000000

      Service-policy : test

        Class-map: af4 (match-all)
          400051 packets, 31203978 bytes
          30 second offered rate 61000 bps, drop rate 0000 bps
          Match:  dscp af41 (34) af42 (36) af43 (38)
          Queueing
          queue limit 100 ms/ 2500 bytes
          queue-limit dscp 34 150 ms/ 3750 bytes
            (pkts output/bytes output) 155882/20888188
            (pkt drops/byte drops) 0/0
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 400051/53606834
          bandwidth 200 kbps

          cac local rate 200 kbps, reserved 200 kbps
          remote signal: enabled
          flow rate fixed 100 kbps

          All flows:
            100.1.1.1/150.1.1.1/17/1000/1000, 100/100 kbps (admitted/current)
            100.1.1.1/150.1.1.1/17/2000/2000, 100/100 kbps (admitted/current)
            100.1.1.1/150.1.1.1/17/3000/3000, 100/100 kbps (non-admitted/current)
            Number of admitted flows: [2]
            Number of non-admitted flows: [1]
            Number of flows in RCAC learning: [0]

        Class-map: class-default (match-any)
          2204 packets, 310386 bytes
          30 second offered rate 0000 bps, drop rate 0000 bps
          Match: any

          queue limit 1700 packets
          (queue depth/total drops/no-buffer drops) 0/0/0
          (pkts output/bytes output) 2204/441352
```

Displays the flow level information on local or remote side interface.

Additional References for Remote PFA

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Per-Flow Admission	*Per-Flow Admission*

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/c/en/us/support/index.html

Feature Information for Remote PFA

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 Remote PFA

Feature Name	Releases	Feature Information
Remote PFA	15.5(3)M	The Remote Per-Flow Admission (PFA) feature enables the nodes to check for the available bandwidth at the remote side interface and the local side interface before admitting a flow. This ensures the quality of admitted flows (calls). The following commands were introduced or modified by this feature: **trigger remote signal, flow rate remote, show policy-map interface**