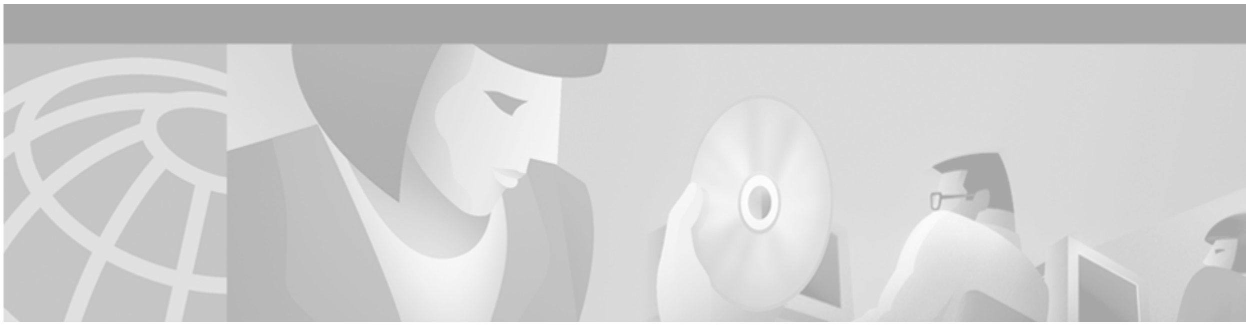

Configuring Weighted Fair Queueing

This chapter 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

For complete conceptual information, see the section "Weighted Fair Queueing" in the chapter "Congestion Management Overview" in this book.

For a complete description of the QoS commands in this chapter, refer to the *Cisco IOS Quality of Service Solutions Command Reference*. To locate documentation of other commands that appear in this chapter, use the command reference master index or search online.

To identify the hardware platform or software image information associated with a feature, use the Feature Navigator on Cisco.com to search for information about the feature or refer to the software release notes for a specific release. For more information, see the "Identifying Supported Platforms" section in the "Using Cisco IOS Software" chapter in this book.

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.) For configuration information on DWFQ, see the section "Distributed Weighted Fair Queueing Configuration Task List" later in this chapter.

To configure flow-based WFQ, perform the tasks described in the following sections. The task in the first section is required; the task in the remaining section is optional.

- Configuring WFQ (Required)
- Monitoring Fair Queueing (Optional)

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

See the end of this chapter for the section "Flow-Based WFQ Configuration Examples."

Configuring WFQ

To configure flow-based WFQ on an interface, use the following command in interface configuration mode:

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

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 (PQ0 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.

Monitoring Fair Queueing

To monitor flow-based fair queueing services in your network, use the following commands in EXEC mode, as needed:

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
- Configuring QoS-Group-Based DWFQ
- Configuring Type of Service-Based DWFQ
- Monitoring DWFQ (Optional)

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.

See the end of this chapter for the section "DWFQ Configuration Examples."

Configuring Flow-Based DWFQ

To configure flow-based DWFQ, use the following commands in interface configuration mode:

	Command	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.
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.

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.

In general, you should not change the aggregate or individual limit value from the default. Use the **fair-queue aggregate-limit** and **fair-queue individual-limit** commands only if you have determined that you would benefit from using different values, based on your particular situation.

Configuring QoS-Group-Based DWFQ

To configure QoS-group-based DWFQ, use the following commands in interface configuration mode:

	Command	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.
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.

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.

Configuring Type of Service-Based DWFQ

To configure type of service (ToS)-based DWFQ, use the following commands in interface configuration mode:

	Command	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.
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.

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.

Monitoring DWFQ

To monitor DWFQ, use the following commands in EXEC mode, as needed:

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

To configure CBWFQ, perform the tasks described in the following sections. The tasks in the first three sections are required; the tasks in the remaining sections are optional.

- Defining Class Maps (Required)
- Configuring Class Policy in the Policy Map (Required)
- Attaching the Service Policy and Enabling CBWFQ (Required)
- Modifying the Bandwidth for an Existing Policy Map Class (Optional)
- Modifying the Queue Limit for an Existing Policy Map Class (Optional)
- Configuring the Bandwidth Limiting Factor (Optional)
- Deleting Classes (Optional)

- Deleting Policy Maps (Optional)
- Verifying Configuration of Policy Maps and Their Classes (Optional)

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

See the end of this chapter for the section "CBWFQ Configuration Examples."

For information on how to configure per-VC WFQ and CBWFQ, see the chapter "Configuring IP to ATM Class of Service" in this book.

Defining Class Maps

To create a class map containing match criteria against which a packet is checked to determine if it belongs to a class—and to effectively create the class whose policy can be specified in one or more policy maps—use the first command in global configuration mode to specify the class map name, then use one of the following commands in class-map configuration mode, as needed:

	Command	Purpose
Step 1	`Router(config)# class-map class-map-name`	Specifies the name of the class map to be created.
Step 2	`Router(config-cmap)# match access-group {access-group | name access-group-name}`	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.
	or	
	`Router(config-cmap)# match input-interface interface-name`	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.
	or	
	`Router(config-cmap)# match protocol protocol`	Specifies the name of the protocol used as a match criterion against which packets are checked to determine if they belong to the class.
	or	
	`Router(config-cmap)# match mpls experimental number`	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.

Other match criteria can be used when defining class maps. For additional match criteria, see the section "Creating a Traffic Class" in the chapter "Configuring the Modular Quality of Service Command-Line Interface" in this book.

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.

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 (Optional)
- Configuring Class Policy Using WRED Packet Drop (Optional)
- Configuring the Class-Default Class Policy (Optional)

Configuring Class Policy Using Tail Drop

To configure a policy map and create class policies that make up the service policy, use the first command in global configuration mode to specify the policy map name, then use the following commands in policy-map class configuration mode, as needed, to configure policy for a standard class. To configure policy for the default class, see the section "Configuring the Class-Default Class Policy" in this chapter.

	Command	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`* `|` **`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.

To configure policy for more than one class in the same policy map, repeat Step 2 through Step 4. 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.

Configuring Class Policy Using WRED Packet Drop

To configure a policy map and create class policies comprising the service policy, use the first command in global configuration mode, as needed, to specify the policy map name, then use the following commands in policy-map class configuration mode, as needed, to configure policy for a standard class. To configure policy for the default class, see the section "Configuring the Class-Default Class Policy" in this chapter.

	Command	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`* `|` **`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.
Step 5	`Router(config-pmap-c)#` **`random-detect exponential-weighting-constant`** *`exponent`*	Configures the exponential weight factor used in calculating the average queue length.
	or	
	`Router(config-pmap-c)#` **`random-detect precedence`** *`precedence min-threshold max-threshold mark-prob-denominator`*	Configures WRED parameters for packets with a specific IP precedence. Repeat this command for each precedence.

To configure policy for more than one class in the same policy map, repeat Step 2 through Step 5. Note that this set of commands uses WRED packet drop, not 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.

Configuring the Class-Default Class Policy

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.

By default, the class-default class is defined as flow-based WFQ. However, configuring the default class with the **bandwidth** policy-map class configuration command disqualifies the default class as flow-based WFQ.

To configure a policy map and configure the class-default class to use tail drop, use the first command in global configuration mode to specify the policy map name, then to configure policy for the default class use the following commands in policy-map class configuration mode:

	Command	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	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.
	or	
	Router(config-pmap-c)# **fair-queue** [*number-of-dynamic-queues*]	Specifies the number of dynamic queues to be reserved for use by flow-based WFQ running on the default class. 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.

To configure a policy map and configure the class-default class to use WRED packet drop, use the first command in global configuration mode to specify the policy map name, then to configure policy for the default class use the following commands in policy-map class configuration mode:

	Command	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.

	Command	Purpose

Step 3	`Router(config-pmap-c)#` **`bandwidth`** `{`*`bandwidth-kbps`*` |` **`percent`** *`percent`*`}`
	
	or
	
	`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)#` **`random-detect`**	Enables WRED. The class policy will drop packets using WRED instead of tail drop.
Step 5	`Router(config-pmap-c)#` **`random-detect exponential-weighting-constant`** *`exponent`* or `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.

Attaching the Service Policy and Enabling CBWFQ

To attach a service policy to the output interface and enable CBWFQ on the interface, use the following command in interface configuration mode. When CBWFQ is enabled, all classes configured as part of the service policy map are installed in the fair queueing system.

Command	Purpose
`Router(config-if)#` **`service-policy output`** *`policy-map`*	Enables CBWFQ and attaches the specified service policy map to the output interface.

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

To change the amount of bandwidth allocated for an existing class, use the following commands beginning in global configuration mode:

	Command	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

To change the maximum number of packets that can accrue in a queue reserved for an existing class, use the following commands beginning in global configuration mode:

	Command	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.

Configuring the Bandwidth Limiting Factor

To change the maximum reserved bandwidth allocated for Resource Reservation Protocol (RSVP), CBWFQ, LLQ, IP RTP Priority, Frame Relay IP RTP Priority, and Frame Relay PVC Interface Priority Queueing (PIPQ), use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)#` **max-reserved-bandwidth** *percent*	Changes the maximum configurable bandwidth for RSVP, CBWFQ, LLQ, IP RTP Priority, Frame Relay IP RTP Priority, and Frame Relay PVC Interface Priority Queueing. The default is 75 percent.

Deleting Classes

To delete one or more class maps from a service policy map, use the following commands beginning in global configuration mode:

	Command	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

To delete a policy map, use the following command in global configuration mode:

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

To display the contents of a specific policy map, a specific class from a specific policy map, or all policy maps configured on an interface, use the following commands in EXEC mode, as needed:

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

The counters displayed after issuing the **show policy-map interface** command are updated only if congestion is present on the 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.

- Modifying the Bandwidth for an Existing Traffic Class (Optional)
- Modifying the Queue Limit for an Existing Traffic Class (Optional)
- Monitoring and Maintaining DCBWFQ (Optional)

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. For information on how to configure QoS with the Modular QoS CLI, see the chapter "Configuring the Modular Quality of Service Command-Line Interface" in this book.

See the end of this chapter for the section "Verifying Configuration of Policy Maps and Their Classes."

Modifying the Bandwidth for an Existing Traffic Class

To change the amount of bandwidth allocated for an existing traffic class in congested environments, use the following commands beginning in global configuration mode:

	Command	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 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.

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 chapter "Configuring the Modular Quality of Service Command-Line Interface" in this book.

Modifying the Queue Limit for an Existing Traffic Class

To change the maximum number of packets that can accrue in a queue reserved for an existing traffic class, use the following commands beginning in global configuration mode:

	Command	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.

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 chapter "Configuring the Modular Quality of Service Command-Line Interface" in this book.

Monitoring and Maintaining DCBWFQ

To display the configuration of a traffic policy and its associated traffic classes, use the following commands in EXEC mode, as needed:

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

To configure IP RTP Priority, 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 IP RTP Priority (Required)
- Configuring the Bandwidth Limiting Factor (Optional)
- Verifying IP RTP Priority (Optional)
- Monitoring and Maintaining IP RTP Priority (Optional)

See the end of this chapter for the section "IP RTP Priority Configuration Examples."

Frame Relay Traffic Shaping (FRTS) and Frame Relay Fragmentation (FRF.12) must be configured before the Frame Relay IP RTP Priority feature is used. For information about configuring FRTS and FRF.12, refer to the *Cisco IOS Wide-Area Networking Configuration Guide*.

Configuring IP RTP Priority

To reserve a strict priority queue for a set of RTP packet flows belonging to a range of UDP destination ports, use the following command in interface configuration mode:

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.

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.

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

The **frame-relay ip rtp priority** command provides strict PQ for Frame Relay PVCs. For more information about this command, refer to the *Cisco IOS Quality of Service Solutions Command Reference*.

Configuring the Bandwidth Limiting Factor

To change the maximum reserved bandwidth allocated for CBWFQ, LLQ, and the IP RTP Priority feature, use the following command in interface configuration mode:

Command	Purpose
Router(config-if)# **max-reserved-bandwidth** *percent*	Changes the maximum configurable bandwidth for CBWFQ, LLQ, and IP RTP Priority. The default is 75 percent.

Verifying IP RTP Priority

To display the contents of the priority queue (such as queue depth and the first packet queued), use the following command in EXEC mode:

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

Monitoring and Maintaining IP RTP Priority

To tune your RTP bandwidth or decrease RTP traffic if the priority queue is experiencing drops, use the following commands in EXEC mode, as needed:

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

To configure Frame Relay IP RTP Priority, 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 Frame Relay IP RTP Priority (Required)
- Verifying Frame Relay IP RTP Priority (Optional)
- Monitoring and Maintaining Frame Relay IP RTP Priority (Optional)

See the end of this chapter for the section "Frame Relay IP RTP Priority Configuration Examples."

Configuring Frame Relay IP RTP Priority

To reserve a strict priority queue on a Frame Relay PVC for a set of RTP packet flows belonging to a range of UDP destination ports, use the following command in map-class configuration mode:

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.

> **Caution** 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

To verify the Frame Relay IP RTP Priority feature, use the following commands in EXEC mode, as needed:

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

To tune your RTP bandwidth or decrease RTP traffic if the priority queue is experiencing drops, use the following command in EXEC mode:

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

To configure the Frame Relay PVC Interface Priority feature, perform the tasks described in the following sections. The tasks in the first three sections are required; the tasks in the remaining sections are optional.

- Configuring PVC Priority in a Map Class (Required)
- Enabling Frame Relay PIPQ and Setting Queue Limits (Required)
- Assigning a Map Class to a PVC (Required)
- Verifying Frame Relay PIPQ (Optional)
- Monitoring and Maintaining Frame Relay PIPQ (Optional)

See the end of this chapter for the section "Frame Relay PVC Interface PQ Configuration Examples."

Configuring PVC Priority in a Map Class

To configure PVC priority within a map class, use the following commands beginning in global configuration mode:

	Command	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

To enable Frame Relay (FR) PIPQ and set the priority queue sizes, use the following commands beginning in global configuration mode:

	Command	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

To assign a map class to a specific PVC, use the following commands beginning in interface configuration mode:

	Command	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

To verify the configuration of Frame Relay (FR) PIPQ, use the following commands in privileged EXEC mode, as needed:

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

To monitor and maintain Frame Relay (FR) PIPQ, use the following commands in privileged EXEC mode, as needed:

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

To configure LLQ, 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 LLQ (Required)
- Configuring the Bandwidth Limiting Factor (Optional)
- Verifying LLQ (Optional)
- Monitoring and Maintaining LLQ (Optional)

See the end of this chapter for the section "LLQ Configuration Examples."

Configuring LLQ

To give priority to a class within a policy map, use the following command in policy-map class configuration mode:

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

Configuring the Bandwidth Limiting Factor

To change the maximum reserved bandwidth allocated for CBWFQ, LLQ, and IP RTP Priority, use the following command in interface configuration mode:

Command	Purpose
`Router(config-if)#` **`max-reserved-bandwidth`** *`percent`*	Changes the maximum configurable bandwidth for CBWFQ, LLQ, and IP RTP Priority. The default is 75 percent.

Verifying LLQ

To display the contents of the priority queue, such as queue depth and the first packet queued, use the following command in EXEC mode:

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

The priority queue is the queue whose conversation ID is equal to the number of dynamic queues plus 8. The packets in the priority queue have a weight of 0.

Monitoring and Maintaining LLQ

To tune your RTP bandwidth or decrease RTP traffic if the priority queue is experiencing drops, use the following commands in EXEC mode, as needed:

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

To configure Distributed LLQ, perform the tasks described in the following sections. The tasks in the first two sections are required; the tasks in the remaining sections are optional.

- Configuring a Priority Queue for an Amount of Available Bandwidth (Required)
- Configuring a Priority Queue for a Percentage of Available Bandwidth (Required)
- Configuring a Transmission Ring Limit (Optional)
- Verifying Distributed LLQ (Optional)
- Verifying a Transmission Ring Limit (Optional)
- Monitoring and Maintaining Distributed LLQ (Optional)

See the end of this chapter for the section "Distributed LLQ Configuration Examples."

Configuring a Priority Queue for an Amount of Available Bandwidth

To give priority to a traffic class based on the amount of available bandwidth within a traffic policy, use the following commands beginning in global configuration mode:

	Command	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.

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 chapter "Modular Quality of Service Command-Line Interface Overview" in this book.

Configuring a Priority Queue for a Percentage of Available Bandwidth

To give priority to a class based on a percentage of available bandwidth, use the following commands beginning in global configuration mode:

	Command	Purpose
Step 1	`Router(config)#` **`policy-map`** *`policy-name`*	Specifies the name of the traffic policy 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 percent`** *`percent`*	Reserves a priority queue with a specified percentage of available bandwidth for CBWFQ traffic.

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 chapter "Modular Quality of Service Command-Line Interface Overview" in this book.

Configuring a Transmission Ring Limit

To limit the number of allowable particles on a transmission ring on an ATM PVC, use the following commands beginning in global interface configuration mode:

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

To limit the number of allowable particles on a transmission ring on an ATM subinterface, use the following commands beginning in global configuration mode:

	Command	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

To view the contents of the priority queue, such as queue depth and the first packet queued, use the following commands in EXEC mode, as needed:

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.

The priority queue is the queue in which the conversation ID is equal to the number of dynamic queues plus 8. The packets in the priority queue have a weight of 0.

Verifying a Transmission Ring Limit

To display the contents of the interface or the PVC, use the following command in EXEC mode:

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

To tune your Real-Time Transport Protocol (RTP) bandwidth or to decrease RTP traffic if the priority queue is experiencing drops, use the following commands in EXEC mode, as needed:

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

To configure LLQ for Frame Relay, perform the tasks described in the following sections. The tasks in the first three sections are required; the tasks in the remaining section are optional.

- Defining Class Maps (Required)
- Configuring Class Policy in the Policy Map (Required)
- Attaching the Service Policy and Enabling LLQ for Frame Relay (Required)
- Verifying Configuration of Policy Maps and Their Classes (Optional)
- Monitoring and Maintaining LLQ for Frame Relay (Optional)

See the end of this chapter for the section "LLQ for Frame Relay Configuration Examples."

Defining Class Maps

To create a class map containing match criteria against which a packet is checked to determine if it belongs to a class, use the following commands beginning in global configuration mode:

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

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. The task in the first section is required; the tasks in the remaining sections are optional.

- Configuring Class Policy for a LLQ Priority Queue (Required)
- Configuring Class Policy Using a Specified Bandwidth and WRED Packet Drop (Optional)
- Configuring the Class-Default Class Policy (Optional)

Configuring Class Policy for a LLQ Priority Queue

To configure a policy map and give priority to a class within the policy map, use the following commands beginning in global configuration mode:

	Command	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

To configure a policy map and create class policies that make up the service policy, use the following commands beginning in global configuration mode:

	Command	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.

	Command	Purpose
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.)
Step 4	`Router(config-pmap-c)#` **`random-detect`**	Enables WRED.

To configure policy for more than one class in the same policy map, repeat Steps 2 through 4.

Configuring the Class-Default Class Policy

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.

To configure a policy map and the class-default class, use the following commands beginning in global configuration mode:

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

To attach a service policy to the output interface and enable LLQ for Frame Relay, use the following command in map-class configuration mode. When LLQ is enabled, all classes configured as part of the service policy map are installed in the fair queueing system.

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.

Verifying Configuration of Policy Maps and Their Classes

To display the contents of a specific policy map or all policy maps configured on an interface, use the following commands in EXEC mod, as needed:

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."

Configuring Burst Size in LLQ Configuration Task List

To configure the burst size in LLQ, perform the tasks described in the following sections. The tasks in the first two sections are required; the task in the remaining section is optional.

- Configuring the LLQ Bandwidth (Required)
- Configuring the LLQ Burst Size (Required)
- Verifying the LLQ Burst Size (Optional)

See the end of this chapter for "Burst Size in LLQ Configuration Examples."

Configuring the LLQ Bandwidth

To configure the LLQ bandwidth, use the following command in policy-map class configuration mode:

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

Configuring the LLQ Burst Size

To configure the LLQ burst size, use the following command in policy-map class configuration mode:

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

To verify the LLQ burst size, use the following commands in EXEC mode, as needed:

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

To configure the per-VC hold queue support for ATM adapters, perform the tasks described in the following sections. The task in the first section is required; the task in the remaining section is optional.

- Configuring the per-VC Hold Queue on an ATM Adapter (Required)
- Verifying the Configuration of the per-VC Hold Queue on an ATM Adapter (Optional)

See the end of this chapter for "Per-VC Hold Queue Support for ATM Adapters Examples."

For related information about per-VC and ATM configurations, see the chapters "IP to ATM Class of Service Overview" and "Configuring IP to ATM Class of Service" later in this book.

Configuring the per-VC Hold Queue on an ATM Adapter

To configure the per-VC hold queue on an ATM adapter, use the following command in global configuration mode:

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

To verify the configuration of the per-VC hold queue on an ATM adapter, use the following command in EXEC mode:

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

Flow-Based WFQ Configuration Examples

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
```

For information on how to configure WFQ, see the section "Flow-Based Weighted Fair Queueing Configuration Task List" in this chapter.

DWFQ Configuration Examples

The following sections provide DWFQ configuration examples:

- Flow-Based DWFQ Example
- QoS-Group-Based DWFQ Example
- ToS-Based DWFQ Example

For information on how to configure DWFQ, see the section "Distributed Weighted Fair Queueing Configuration Task List" in this chapter.

Flow-Based DWFQ Example

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
```

QoS-Group-Based DWFQ Example

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
```

ToS-Based DWFQ Example

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
```

CBWFQ Configuration Examples

The following sections provide CBWFQ configuration examples:

- Class Map Configuration Example
- Policy Creation Example
- Policy Attachment to Interfaces Example
- CBWFQ Using WRED Packet Drop Example
- Display Service Policy Map Content Examples

For information on how to configure CBWFQ, see the section "Class-Based Weighted Fair Queueing Configuration Task List" in this chapter.

Class Map Configuration Example

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
```

Policy Creation Example

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 "Class Map Configuration Example" 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
```

Policy Attachment to Interfaces Example

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, "Policy Creation Example."

```
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
```

CBWFQ Using WRED Packet Drop Example

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
!
```

Display Service Policy Map Content Examples

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

- Display all classes that make up a specified service policy map
- Display all classes configured for all service policy maps
- Display a specified class of a service policy map
- Display all classes configured for all service policy maps on a specified interface

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)
```

Distributed CBWFQ Configuration Examples

The following sections provide DCBWFQ configuration examples:

- Traffic Class Configuration Example
- Traffic Policy Creation Example
- Traffic Policy Attachment to an Interface Example

For information on how to configure DCBWFQ, see the section "Distributed Class-Based Weighted Fair Queueing Configuration Task List" in this chapter.

Traffic Class Configuration Example

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
```

For additional information on traffic classes, see the chapter "Configuring the Modular Quality of Service Command-Line Interface" in this book.

Traffic Policy Creation Example

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 "Class Map Configuration Example" 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
```

For additional information on traffic policy configurations, see the chapter "Configuring the Modular Quality of Service Command-Line Interface" in this book.

Traffic Policy Attachment to an Interface Example

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
```

For additional information on attaching traffic policy configurations to interfaces, see the chapter "Configuring the Modular Quality of Service Command-Line Interface" in this book.

IP RTP Priority Configuration Examples

The following sections provide IP RTP Priority configuration examples:

- CBWFQ Configuration Example
- Virtual Template Configuration Example
- Multilink Bundle Configuration Example
- Debug Example

For information on how to configure IP RTP Priority, see the section "IP RTP Priority Configuration Task List" in this chapter.

CBWFQ Configuration Example

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.

Virtual Template Configuration Example

The following example configures a strict priority queue in a virtual template configuration with CBWFQ. The **max-reserved-bandwidth** command changes the maximum reserved bandwidth allocated for CBWFQ and IP RTP Priority from the default (75 percent) to 80 percent.

```
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)# max-reserved-bandwidth 80
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.

Multilink Bundle Configuration Example

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. The **max-reserved-bandwidth** command changes the maximum reserved bandwidth allocated for WFQ and IP RTP Priority.

```
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
Router(config-if)# max-reserved-bandwidth 80
```

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
```

Debug Example

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
```

Frame Relay IP RTP Priority Configuration Examples

This "Strict Priority Service to Matching RTP Packets Example" section provides a configuration example.

For information on how to configure Frame Relay IP RTP Priority queueing, see the section "Frame Relay IP RTP Priority Configuration Task List" in this chapter.

Strict Priority Service to Matching RTP Packets Example

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
```

Frame Relay PVC Interface PQ Configuration Examples

This section provides configuration examples for Frame Relay PIPQ.

For information on how to configure Frame Relay PIPQ, see the section "Frame Relay PVC Interface Priority Configuration Task List" in this chapter.

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
```

LLQ Configuration Examples

The following sections provide LLQ configuration examples:

- ATM PVC Configuration Example
- Virtual Template Configuration Example
- Multilink Bundle Configuration Example

For information on how to configure LLQ, see the section "Low Latency Queueing Configuration Task List" in this chapter.

ATM PVC Configuration Example

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
```

Virtual Template Configuration Example

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
```

Multilink Bundle Configuration Example

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
```

Distributed LLQ Configuration Examples

The following sections provide distributed LLQ configuration examples:

- Enabling PQ for an Amount of Available Bandwidth on an ATM Subinterface Example
- Enabling PQ for a Percentage of Available Bandwidth on an ATM Subinterface Example
- Limiting the Transmission Ring Limit on an ATM Interface Example
- Limiting the Transmission Ring Limit on an ATM PVC Subinterface Example

For information on how to configure distributed LLQ, see the section “Distributed LLQ Configuration Task List” in this chapter.

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

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
```

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

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
```

Limiting the Transmission Ring Limit on an ATM Interface Example

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
```

Limiting the Transmission Ring Limit on an ATM PVC Subinterface Example

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.

LLQ for Frame Relay Configuration Examples

The following section provides a LLQ for Frame Relay configuration examples.

For information on how to configure LLQ for Frame Relay, see the section "Low Latency Queueing for Frame Relay Configuration Task List" in this chapter.

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
```

Burst Size in LLQ Configuration Examples

For information on how to configure the burst size in LLQ, see the section "Configuring Burst Size in LLQ Configuration Task List" in this chapter.

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
```

Per-VC Hold Queue Support for ATM Adapters Examples

For information on how to configure per-VC hold queue support for ATM Adapters, see the section "Per-VC Hold Queue Support for ATM Adapters Configuration Task List" in this chapter.

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

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