



CHAPTER 22

Configuring QoS

Have you ever participated in a long-distance phone call that involved a satellite connection? The conversation might be interrupted with brief, but perceptible, gaps at odd intervals. Those gaps are the time, called the latency, between the arrival of packets being transmitted over the network. Some network traffic, such as voice and video, cannot tolerate long latency times. Quality of Service (QoS) is a feature that lets you give priority to critical traffic, prevent bandwidth hogging, and manage network bottlenecks to prevent packet drops.

This chapter describes how to apply QoS policies, and includes the following sections:

- [QoS Overview, page 22-1](#)
- [Creating the Standard Priority Queue for an Interface, page 22-3](#)
- [Identifying Traffic for QoS Using Class Maps, page 22-6](#)
- [Creating a Policy for Standard Priority Queueing and/or Policing, page 22-7](#)
- [Viewing QoS Statistics, page 22-9](#)
- [Viewing QoS Statistics, page 22-9](#)

QoS Overview

You should consider that in an ever-changing network environment, QoS is not a one-time deployment, but an ongoing, essential part of network design.



Note

QoS is only available in single context mode.

This section describes the QoS features supported by the adaptive security appliance, and includes the following topics:

- [Supported QoS Features, page 22-2](#)
- [What is a Token Bucket?, page 22-2](#)
- [Policing Overview, page 22-2](#)
- [Priority Queueing Overview, page 22-3](#)
- [DSCP and DiffServ Preservation, page 22-3](#)

Supported QoS Features

The adaptive security appliance supports the following QoS features:

- Policing—To prevent individual flows from hogging the network bandwidth, you can limit the maximum bandwidth used per flow. See the “[Policing Overview](#)” section on page 22-2 for more information.
- Priority queuing—For critical traffic that cannot tolerate latency, such as Voice over IP (VoIP), you can identify traffic for Low Latency Queuing (LLQ) so that it is always transmitted ahead of other traffic. See the “[Priority Queueing Overview](#)” section on page 22-3 for more information.

What is a Token Bucket?

A token bucket is used to manage a device that regulates the data in a flow. For example, the regulator might be a traffic policer. A token bucket itself has no discard or priority policy. Rather, a token bucket discards tokens and leaves to the flow the problem of managing its transmission queue if the flow overdrives the regulator.

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

average rate = burst size / time interval

Here are some definitions of these terms:

- Average rate—Also called the committed information rate (CIR), it specifies how much data can be sent or forwarded per unit time on average.
- Burst size—Also called the Committed Burst (Bc) size, it specifies in bits or bytes per burst how much traffic can be sent within a given unit of time to not create scheduling concerns.
- Time interval—Also called the measurement interval, it specifies the time quantum in seconds per burst.

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

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

Policing Overview

Policing is a way of ensuring that no traffic exceeds the maximum rate (in bits/second) that you configure, thus ensuring that no one traffic flow or class can take over the entire resource. When traffic exceeds the maximum rate, the adaptive security appliance drops the excess traffic. Policing also sets the largest single burst of traffic allowed.

Priority Queueing Overview

LLQ priority queueing lets you prioritize certain traffic flows (such as latency-sensitive traffic like voice and video) ahead of other traffic. Priority queueing uses an LLQ priority queue on an interface (see the [“Creating the Standard Priority Queue for an Interface” section on page 22-3](#)), while all other traffic goes into the “best effort” queue. Because queues are not of infinite size, they can fill and overflow. When a queue is full, any additional packets cannot get into the queue and are dropped. This is called *tail drop*. To avoid having the queue fill up, you can increase the queue buffer size. You can also fine-tune the maximum number of packets allowed into the transmit queue. These options let you control the latency and robustness of the priority queueing. Packets in the LLQ queue are always transmitted before packets in the best effort queue.

How QoS Features Interact

You can configure each of the QoS features alone if desired for the adaptive security appliance. Often, though, you configure multiple QoS features on the adaptive security appliance so you can prioritize some traffic, for example, and prevent other traffic from causing bandwidth problems. See the following supported feature combinations per interface:

- Standard priority queueing (for specific traffic) + Policing (for the rest of the traffic).

You cannot configure priority queueing and policing for the same set of traffic.

DSCP and DiffServ Preservation

- DSCP markings are preserved on all traffic passing through the adaptive security appliance.
- The adaptive security appliance does not locally mark/re-mark any classified traffic, but it honors the Expedited Forwarding (EF) DSCP bits of every packet to determine if it requires “priority” handling and will direct those packets to the LLQ.
- DiffServ marking is preserved on packets when they traverse the service provider backbone so that QoS can be applied in transit (QoS tunnel pre-classification).

Creating the Standard Priority Queue for an Interface

If you enable standard priority queueing for traffic on a physical interface, then you need to also create the priority queue on each interface. Each physical interface uses two queues: one for priority traffic, and the other for all other traffic. For the other traffic, you can optionally configure policing.



Note

You cannot create a priority queue for a Ten Gigabit Ethernet interface; priority queueing is not necessary for an interface with high bandwidth.

This section includes the following topics:

- [Determining the Queue and TX Ring Limits, page 22-4](#)
- [Configuring the Priority Queue, page 22-5](#)

Determining the Queue and TX Ring Limits

To determine the priority queue and TX ring limits, use the worksheets below.

Table 22-1 shows how to calculate the priority queue size. Because queues are not of infinite size, they can fill and overflow. When a queue is full, any additional packets cannot get into the queue and are dropped (called *tail drop*). To avoid having the queue fill up, you can adjust the queue buffer size according to the “Configuring the Priority Queue” section on page 22-5.

Table 22-1 Queue Limit Worksheet

Step 1	_____ Mbps	x	125	=	_____
	<i>Outbound bandwidth (Mbps or Kbps)¹</i>				<i># of bytes/ms</i>
	_____ Kbps	x	.125	=	_____
					<i># of bytes/ms</i>
Step 2	_____	÷	_____	x	_____
	<i># of bytes/ms from Step 1</i>		<i>Average packet size (bytes)²</i>		<i>Delay (ms)³</i>
				=	_____
					<i>Queue limit (# of packets)</i>

1. For example, DSL might have an uplink speed of 768 Kbps. Check with your provider.
2. Determine this value from a codec or sampling size. For example, for VoIP over VPN, you might use 160 bytes. We recommend 256 bytes if you do not know what size to use.
3. The delay depends on your application. For example, the recommended maximum delay for VoIP is 200 ms. We recommend 500 ms if you do not know what delay to use.

Table 22-2 shows how to calculate the TX ring limit. This limit determines the maximum number of packets allowed into the Ethernet transmit driver before the driver pushes back to the queues on the interface to let them buffer packets until the congestion clears. This setting guarantees that the hardware-based transmit ring imposes a limited amount of extra latency for a high-priority packet.

Table 22-2 TX Ring Limit Worksheet

Step 1	_____ Mbps	x	125	=	_____
	<i>Outbound bandwidth (Mbps or Kbps)¹</i>				<i># of bytes/ms</i>
	_____ Kbps	x	0.125	=	_____
					<i># of bytes/ms</i>
Step 2	_____	÷	_____	x	_____
	<i># of bytes/ms from Step 1</i>		<i>Maximum packet size (bytes)²</i>		<i>Delay (ms)³</i>
				=	_____
					<i>TX ring limit (# of packets)</i>

1. For example, DSL might have an uplink speed of 768 Kbps. Check with your provider.

- Typically, the maximum size is 1538 bytes, or 1542 bytes for tagged Ethernet. If you allow jumbo frames, then the packet size might be larger.
- The delay depends on your application. For example, to control jitter for VoIP, you should use 20 ms.

Configuring the Priority Queue

To create the priority queue, perform the following steps.

Step 1 To create the priority queue, enter the following command:

```
hostname(config)# priority-queue interface_name
```

Where the *interface_name* argument specifies the physical interface name on which you want to enable the priority queue.



Note You cannot create a priority queue for a Ten Gigabit Ethernet interface; priority queuing is not necessary for an interface with high bandwidth.

Step 2 (Optional) To change the size of the priority queues, enter the following command:

```
hostname(config-priority-queue)# queue-limit number_of_packets
```

The default queue limit is 1024 packets. Because queues are not of infinite size, they can fill and overflow. When a queue is full, any additional packets cannot get into the queue and are dropped (called *tail drop*). To avoid having the queue fill up, you can use the **queue-limit** command to increase the queue buffer size.

See the [“Determining the Queue and TX Ring Limits” section on page 22-4](#) to determine the *number_of_packets* value.

The upper limit of the range of values for the **queue-limit** command is determined dynamically at run time. To view this limit, enter **queue-limit ?** on the command line. The key determinants are the memory needed to support the queues and the memory available on the device.

The **queue-limit** that you specify affects both the higher priority low-latency queue and the best effort queue.

Step 3 (Optional) To specify the depth of the priority queues, enter the following command:

```
hostname(config-priority-queue)# tx-ring-limit number_of_packets
```

The default tx-ring-limit is 128 packets. This command sets the maximum number of low-latency or normal priority packets allowed into the Ethernet transmit driver before the driver pushes back to the queues on the interface to let them buffer packets until the congestion clears. This setting guarantees that the hardware-based transmit ring imposes a limited amount of extra latency for a high-priority packet.

See the [“Determining the Queue and TX Ring Limits” section on page 22-4](#) to determine the *number_of_packets* value.

The upper limit of the range of values for the **tx-ring-limit** command is determined dynamically at run time. To view this limit, enter **tx-ring-limit ?** on the command line. The key determinants are the memory needed to support the queues and the memory available on the device.

The **tx-ring-limit** that you specify affects both the higher priority low-latency queue and the best-effort queue.

The following example establishes a priority queue on interface “outside” (the GigabitEthernet0/1 interface), with the default queue-limit and tx-ring-limit.

```
hostname(config)# priority-queue outside
```

The following example establishes a priority queue on the interface “outside” (the GigabitEthernet0/1 interface), sets the queue-limit to 260 packets, and sets the tx-ring-limit to 3:

```
hostname(config)# priority-queue outside
hostname(config-priority-queue)# queue-limit 260
hostname(config-priority-queue)# tx-ring-limit 3
```

Identifying Traffic for QoS Using Class Maps

QoS is part of the Modular Policy Framework. See the [Chapter 15, “Using Modular Policy Framework,”](#) for more information. In Modular Policy Framework, you identify the traffic on which you want to enable QoS in a class map. This section includes the following topics:

- [Creating a QoS Class Map, page 22-6](#)
- [QoS Class Map Examples, page 22-6](#)

Creating a QoS Class Map

For priority traffic, identify only latency-sensitive traffic. For policing traffic, you can choose to police all other traffic, or you can limit the traffic to certain types.

To create the class maps for QoS traffic, see the **class-map** command in the [“Identifying Traffic \(Layer 3/4 Class Map\)”](#) section on page 15-4.

You can match traffic based on many characteristics, including access lists, tunnel groups, DSCP, precedence, and more. You cannot use the **class-default** class map for priority traffic.

QoS Class Map Examples

For example, in the following sequence, the **class-map** command classifies all non-tunneled TCP traffic, using an access list named tcp_traffic:

```
hostname(config)# access-list tcp_traffic permit tcp any any
hostname(config)# class-map tcp_traffic
hostname(config-cmap)# match access-list tcp_traffic
```

In the following example, other, more specific match criteria are used for classifying traffic for specific, security-related tunnel groups. These specific match criteria stipulate that a match on tunnel-group (in this case, the previously-defined Tunnel-Group-1) is required as the first match characteristic to classify traffic for a specific tunnel, and it allows for an additional match line to classify the traffic (IP differential services code point, expedited forwarding).

```
hostname(config)# class-map TG1-voice
hostname(config-cmap)# match tunnel-group tunnel-grp1
hostname(config-cmap)# match dscp ef
```

In the following example, the **class-map** command classifies both tunneled and non-tunneled traffic according to the traffic type:

```
hostname(config)# access-list tunneled extended permit ip 10.10.34.0 255.255.255.0
20.20.10.0 255.255.255.0
hostname(config)# access-list non-tunneled extended permit tcp any any
hostname(config)# tunnel-group tunnel-grp1 type IPSec_L2L
```

```

hostname(config)# class-map browse
hostname(config-cmap)# description "This class-map matches all non-tunneled tcp traffic."
hostname(config-cmap)# match access-list non-tunneled

hostname(config-cmap)# class-map TG1-voice
hostname(config-cmap)# description "This class-map matches all dscp ef traffic for
tunnel-grp 1."
hostname(config-cmap)# match dscp ef
hostname(config-cmap)# match tunnel-group tunnel-grp1

hostname(config-cmap)# class-map TG1-BestEffort
hostname(config-cmap)# description "This class-map matches all best-effort traffic for
tunnel-grp1."
hostname(config-cmap)# match tunnel-group tunnel-grp1
hostname(config-cmap)# match flow ip destination-address

```

The following example shows a way of policing a flow within a tunnel, provided the classed traffic is not specified as a tunnel, but does go *through* the tunnel. In this example, 192.168.10.10 is the address of the host machine on the private side of the remote tunnel, and the access list is named “host-over-l2l”. By creating a class-map (named “host-specific”), you can then police the “host-specific” class before the LAN-to-LAN connection polices the tunnel. In this example, the “host-specific” traffic is rate-limited before the tunnel, then the tunnel is rate-limited:

```

hostname(config)# access-list host-over-l2l extended permit ip any host 192.168.10.10
hostname(config)# class-map host-specific
hostname(config-cmap)# match access-list host-over-l2l

```

The following example builds on the configuration developed in the previous section. As in the previous example, there are two named class-maps: tcp_traffic and TG1-voice.

```

hostname(config)# class-map TG1-best-effort
hostname(config-cmap)# match tunnel-group Tunnel-Group-1
hostname(config-cmap)# match flow ip destination-address

```

Adding a third class map provides a basis for defining a tunneled and non-tunneled QoS policy, as follows, which creates a simple QoS policy for tunneled and non-tunneled traffic, assigning packets of the class TG1-voice to the low latency queue and setting rate limits on the tcp_traffic and TG1-best-effort traffic flows.

Creating a Policy for Standard Priority Queueing and/or Policing

After you identify the traffic in “[Identifying Traffic for QoS Using Class Maps](#)” section on page 22-6, you can create a policy map for an interface or globally for all interfaces that assigns QoS actions (and other feature actions) to the traffic in the class map. (See the [Chapter 15, “Using Modular Policy Framework,”](#) for information about other features. This chapter only discusses QoS.)

You can configure standard priority queueing and policing for different class maps within the same policy map. See the “[How QoS Features Interact](#)” section on page 22-3 for information about valid QoS configurations.

To create a policy map, perform the following steps:

- Step 1** To add or edit a policy map, enter the following command:

```
hostname(config)# policy-map name
```

For example:

```
hostname(config)# policy-map QoS_policy
```

Step 2 To configure priority queueing, enter the following commands:

```
hostname(config-pmap)# class priority_map_name
hostname(config-pmap-c)# priority
```

where the *priority_map_name* is the class map you created for prioritized traffic in “[Identifying Traffic for QoS Using Class Maps](#)” section on page 22-6.

For example:

```
hostname(config)# class-map priority-class
hostname(config-cmap)# match tunnel-group Tunnel-Group-1
hostname(config-cmap)# match dscp ef

hostname(config-cmap)# policy-map QoS_policy

hostname(config-pmap)# class priority_class
hostname(config-pmap-c)# priority
```

Step 3 To configure policing, enter the following commands:

```
hostname(config-pmap)# class policing_map_name
hostname(config-pmap-c)# police {output | input} conform-rate [conform-burst]
[conform-action [drop | transmit]] [exceed-action [drop | transmit]]
```

where the *policing_map_name* is the class map you created for prioritized traffic in “[Identifying Traffic for QoS Using Class Maps](#)” section on page 22-6.

The *conform-burst* argument specifies the maximum number of instantaneous bytes allowed in a sustained burst before throttling to the conforming rate value, between 1000 and 512000000 bytes.

The **conform-action** keyword sets the action to take when the rate is less than the *conform_burst* value.

The *conform-rate* argument sets the rate limit for this traffic flow; between 8000 and 2000000000 bits per second.

The **drop** keyword drops the packet.

The **exceed-action** keyword sets the action to take when the rate is between the *conform-rate* value and the *conform-burst* value.

The **input** keyword enables policing of traffic flowing in the input direction.

The **output** keyword enables policing of traffic flowing in the output direction.

The **transmit** keyword transmits the packet.

For example:

```
hostname(config)# class-map policing-class
hostname(config-cmap)# match any

hostname(config-cmap)# policy-map QoS_policy

hostname(config-pmap)# class police_class
hostname(config-pmap-c)# police output 56000 10500
```

Step 4 To activate the policy map on one or more interfaces, enter the following command:

```
hostname(config)# service-policy policymap_name {global | interface interface_name}
```

Where **global** applies the policy map to all interfaces, and **interface** applies the policy to one interface. Only one global policy is allowed. Interface service policies take precedence over the global service policy for a given feature. For example, if you have a global policy with inspections, and an interface

policy with TCP normalization, then both inspections and TCP normalization are applied to the interface. However, if you have a global policy with inspections, and an interface policy with inspections, then only the interface policy inspections are applied to that interface.

In this example, the maximum rate for traffic of the `tcp_traffic` class is 56,000 bits/second and a maximum burst size of 10,500 bytes per second. For the `TC1-BestEffort` class, the maximum rate is 200,000 bits/second, with a maximum burst of 37,500 bytes/second. Traffic in the `TC1-voice` class has no policed maximum speed or burst rate because it belongs to a priority class.

```
hostname(config)# access-list tcp_traffic permit tcp any any
hostname(config)# class-map tcp_traffic
hostname(config-cmap)# match access-list tcp_traffic

hostname(config)# class-map TG1-voice
hostname(config-cmap)# match tunnel-group tunnel-grp1
hostname(config-cmap)# match dscp ef

hostname(config-cmap)# class-map TG1-BestEffort
hostname(config-cmap)# match tunnel-group tunnel-grp1
hostname(config-cmap)# match flow ip destination-address

hostname(config)# policy-map qos
hostname(config-pmap)# class tcp_traffic
hostname(config-pmap-c)# police output 56000 10500

hostname(config-pmap-c)# class TG1-voice
hostname(config-pmap-c)# priority

hostname(config-pmap-c)# class TG1-best-effort
hostname(config-pmap-c)# police output 200000 37500

hostname(config-pmap-c)# class class-default
hostname(config-pmap-c)# police output 1000000 37500

hostname(config-pmap-c)# service-policy qos global
```

Viewing QoS Statistics

This section includes the following topics:

- [Viewing QoS Police Statistics, page 22-9](#)
- [Viewing QoS Standard Priority Statistics, page 22-10](#)
- [Viewing QoS Standard Priority Queue Statistics, page 22-10](#)

Viewing QoS Police Statistics

To view the QoS statistics for traffic policing, use the `show service-policy` command with the `police` keyword:

```
hostname# show service-policy police
```

The following is sample output for the `show service-policy police` command:

```
hostname# show service-policy police
```

```

Global policy:
  Service-policy: global_fw_policy

Interface outside:
  Service-policy: qos
  Class-map: browse
    police Interface outside:
      cir 56000 bps, bc 10500 bytes
      conformed 10065 packets, 12621510 bytes; actions: transmit
      exceeded 499 packets, 625146 bytes; actions: drop
      conformed 5600 bps, exceed 5016 bps
  Class-map: cmap2
    police Interface outside:
      cir 200000 bps, bc 37500 bytes
      conformed 17179 packets, 20614800 bytes; actions: transmit
      exceeded 617 packets, 770718 bytes; actions: drop
      conformed 198785 bps, exceed 2303 bps

```

Viewing QoS Standard Priority Statistics

To view statistics for service policies implementing the **priority** command, use the **show service-policy** command with the **priority** keyword:

```
hostname# show service-policy priority
```

The following is sample output for the **show service-policy priority** command:

```

hostname# show service-policy priority
Global policy:
  Service-policy: global_fw_policy
Interface outside:
  Service-policy: qos
  Class-map: TG1-voice
  Priority:
    Interface outside: aggregate drop 0, aggregate transmit 9383

```



Note

“Aggregate drop” denotes the aggregated drop in this interface; “aggregate transmit” denotes the aggregated number of transmitted packets in this interface.

Viewing QoS Standard Priority Queue Statistics

To display the priority-queue statistics for an interface, use the **show priority-queue statistics** command in privileged EXEC mode. The results show the statistics for both the best-effort (BE) queue and the low-latency queue (LLQ). The following example shows the use of the **show priority-queue statistics** command for the interface named test, and the command output.

```

hostname# show priority-queue statistics test

Priority-Queue Statistics interface test

Queue Type           = BE
Packets Dropped      = 0
Packets Transmit     = 0
Packets Enqueued     = 0
Current Q Length     = 0
Max Q Length         = 0

```

```
Queue Type          = LLQ
Packets Dropped     = 0
Packets Transmit    = 0
Packets Enqueued    = 0
Current Q Length    = 0
Max Q Length        = 0
hostname#
```

In this statistical report, the meaning of the line items is as follows:

- “Packets Dropped” denotes the overall number of packets that have been dropped in this queue.
- “Packets Transmit” denotes the overall number of packets that have been transmitted in this queue.
- “Packets Enqueued” denotes the overall number of packets that have been queued in this queue.
- “Current Q Length” denotes the current depth of this queue.
- “Max Q Length” denotes the maximum depth that ever occurred in this queue.

