QoS Policing

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

This chapter provides conceptual and configuration details for QoS Traffic Policing.

Policing Overview

Traffic policing manages the maximum rate of traffic through a token bucket algorithm. The token bucket algorithm uses user-configured values to determine the maximum rate of traffic allowed on an interface at a given moment in time. The token bucket algorithm is affected by all traffic entering or leaving the interface (depending on where the traffic policy with traffic policing is configured) and is useful in managing network bandwidth in cases where several large packets are sent in the same traffic stream.

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

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

Note Configured values take into account the Layer 1 encapsulation applied to traffic. This applies to both ingress and egress policing. For Ethernet, the encapsulation is 34 bytes; whereas for 802.1Q, the encapsulation is 38 bytes.

Traffic policing is often configured on interfaces at the edge of a network to limit the rate of traffic entering or leaving the network. In the most common traffic policing configurations, traffic that conforms to the CIR is sent and traffic that exceeds is sent with a decreased priority or is dropped. Users can change these configuration options to suit their network needs. Traffic policing also provides a certain amount of bandwidth management by allowing you to set the burst size (Bc) for the committed information rate (CIR). When the

peak information rate (PIR) is supported, a second token bucket is enforced and then the traffic policer is called a two-rate policer.

The supported policing features are:

- Single-rate policers
- Two-rate policers

Regulation of Traffic with the Policing Mechanism

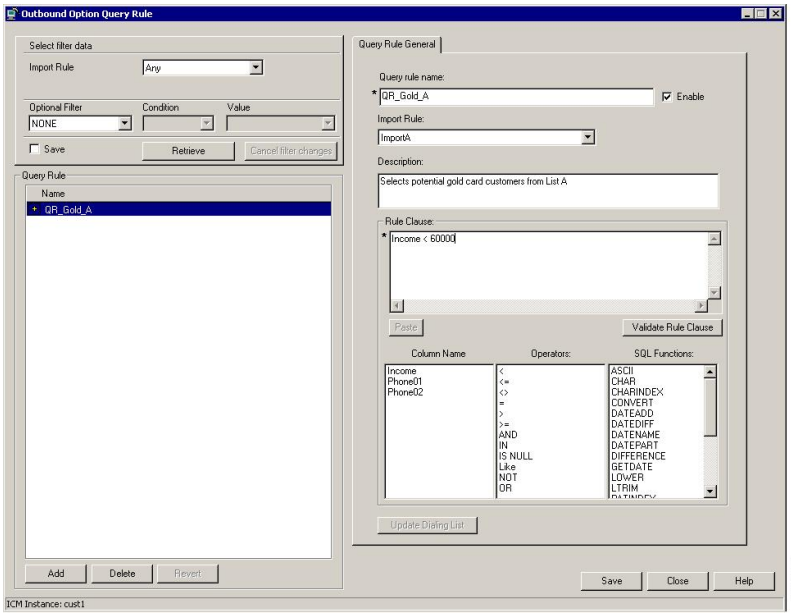
This section describes the single-rate and two-rate policing mechanisms.

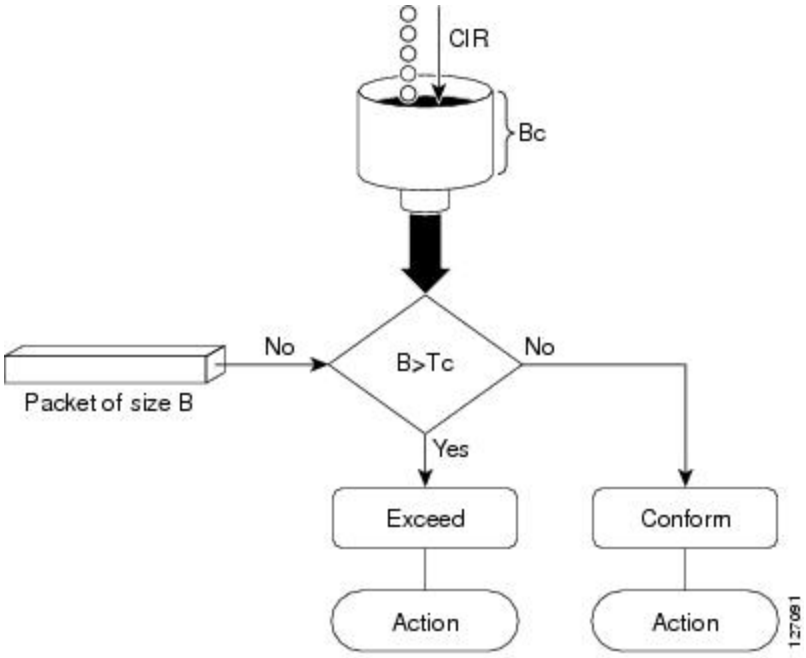
Single-Rate Policer

A single-rate, two-action policer provides one token bucket with two actions for each packet: a conform action and an exceed action.

This figure illustrates how a single-rate token bucket policer marks packets as either conforming or exceeding a CIR, and assigns an action.

Figure 1: Marking Packets and Assigning Actions—Single-Rate Policer





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

Two-Rate Policer

The two-rate policer manages the maximum rate of traffic by using two token buckets: the committed token bucket and the peak token bucket. The dual-token bucket algorithm uses user-configured values to determine the maximum rate of traffic allowed on a queue at a given moment. In this way, the two-rate policer can meter traffic at two independent rates: the committed information rate (CIR) and the peak information rate (PIR).

The committed token bucket can hold bytes up to the size of the committed burst (bc) before overflowing. This token bucket holds the tokens that determine whether a packet conforms to or exceeds the CIR as the following describes:

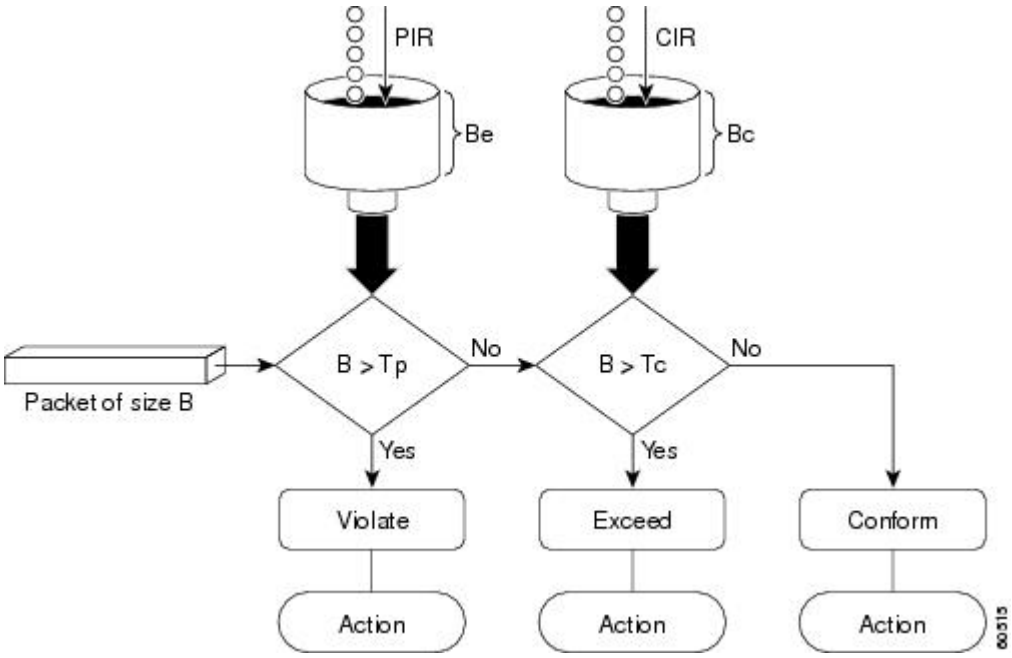
- A traffic stream is conforming when the average number of bytes over time does not cause the committed token bucket to overflow. When this occurs, the token bucket algorithm marks the traffic stream green.
- A traffic stream is exceeding when it causes the committed token bucket to overflow into the peak token bucket. When this occurs, the token bucket algorithm marks the traffic stream yellow. The peak token bucket is filled as long as the traffic exceeds the police rate.

The peak token bucket can hold bytes up to the size of the peak burst (be) before overflowing. This token bucket holds the tokens that determine whether a packet violates the PIR. A traffic stream is violating when it causes the peak token bucket to overflow. When this occurs, the token bucket algorithm marks the traffic stream red.

The dual-token bucket algorithm provides users with three actions for each packet—a conform action, an exceed action, and an optional violate action. Traffic entering a queue with the two-rate policer configured is placed into one of these categories. Within these three categories, users can decide packet treatments. For

instance, packets that conform can be configured to be sent; packets that exceed can be configured to be sent with a decreased priority; and packets that violate can be configured to be dropped.

Figure 2: Marking Packets and Assigning Actions—2-Rate Policer



For example, if a data stream with a rate of 250 kbps arrives at the two-rate policer, and the CIR is 100 kbps and the PIR is 200 kbps, the policer marks the packet in the following way:

- 100 kbps conforms to the rate
- 100 kbps exceeds the rate
- 50 kbps violates the rate

The router updates the tokens for both the committed and peak token buckets in the following way:

- The router updates the committed token bucket at the CIR value each time a packet arrives at the interface. The committed token bucket can contain up to the committed burst (bc) value.
- The router updates the peak token bucket at the PIR value each time a packet arrives at the interface. The peak token bucket can contain up to the peak burst (be) value.
- When an arriving packet conforms to the CIR, the router takes the conform action on the packet and decrements both the committed and peak token buckets by the number of bytes of the packet.
- When an arriving packet exceeds the CIR, the router takes the exceed action on the packet, decrements the committed token bucket by the number of bytes of the packet, and decrements the peak token bucket by the number of overflow bytes of the packet.
- When an arriving packet exceeds the PIR, the router takes the violate action on the packet, but does not decrement the peak token bucket.

Committed Bursts and Excess Bursts

Unlike a traffic shaper, a traffic policer does not buffer excess packets and transmit them later. Instead, the policer executes a “send or do not send” policy without buffering. During periods of congestion, proper configuration of the excess burst parameter enables the policer to drop packets less aggressively. Therefore, it is important to understand how policing uses the committed (normal) and excess burst values to ensure the router reaches the configured committed information rate (CIR).

Burst parameters are based on a generic buffering rule for routers, which recommends that you configure buffering to be equal to the round-trip time bit-rate to accommodate the outstanding TCP windows of all connections in times of congestion.

Committed Bursts

The committed burst (bc) parameter of the police command implements the first, conforming (green) token bucket that the router uses to meter traffic. The bc parameter sets the size of this token bucket. Initially, the token bucket is full and the token count is equal to the committed burst size (CBS). Thereafter, the meter updates the token counts the number of times per second indicated by the committed information rate (CIR).

The following describes how the meter uses the conforming token bucket to send packets:

- If sufficient tokens are in the conforming token bucket when a packet arrives, the meter marks the packet green and decrements the conforming token count by the number of bytes of the packet.
- If there are insufficient tokens available in the conforming token bucket, the meter allows the traffic flow to borrow the tokens needed to send the packet. The meter checks the exceeding token bucket for the number of bytes of the packet. If the exceeding token bucket has a sufficient number of tokens available, the meter marks the packet:

  Green and decrements the conforming token count down to the minimum value of 0.

  Yellow, borrows the remaining tokens needed from the exceeding token bucket, and decrements the exceeding token count by the number of tokens borrowed down to the minimum value of 0.
- If an insufficient number of tokens is available, the meter marks the packet red and does not decrement either of the conforming or exceeding token counts.

Note When the meter marks a packet with a specific color, there must be a sufficient number of tokens of that color to accommodate the entire packet. Therefore, the volume of green packets is never smaller than the committed information rate (CIR) and committed burst size (CBS). Tokens of a given color are always used on packets of that color.

The default committed burst size is the greater of 2 milliseconds of bytes at the police rate or the network maximum transmission unit (MTU).

Committed Burst Calculation

To calculate committed burst, use the following formula:

bc = CIR bps * (1 byte) / (8 bits) * 1.5 seconds

Note 1.5 seconds is the typical round-trip time.

For example, if the committed information rate is 512000 bps, then using the committed burst formula, the committed burst is 96000 bytes.

bc = 512000 * 1/8 * 1.5

bc = 64000 * 1.5 = 96000

Note When the be value equals 0, we recommend that you set the egress bc value to be greater than or equal to the ingress bc value plus 1. Otherwise, packet loss can occur. For example: be = 0 egress bc >= ingress bc + 1

Excess Bursts

The excess burst (be) parameter of the police command implements the second, exceeding (yellow) token bucket that the router uses to meter traffic. The exceeding token bucket is initially full and the token count is equal to the excess burst size (EBS). Thereafter, the meter updates the token counts the number of times per second indicated by the committed information rate (CIR).

The following describes how the meter uses the exceeding token bucket to send packets:

- When the first token bucket (the conforming bucket) meets the committed burst size (CBS), the meter allows the traffic flow to borrow the tokens needed from the exceeding token bucket. The meter marks the packet yellow and then decrements the exceeding token bucket by the number of bytes of the packet.
- If the exceeding token bucket does not have the required tokens to borrow, the meter marks the packet red and does not decrement the conforming or the exceeding token bucket. Instead, the meter performs the exceed-action configured in the police command (for example, the policer drops the packets).

Excess Burst Calculation

To calculate excess burst, use the following formula:

be = 2 * committed burst

For example, if you configure a committed burst of 4000 bytes, then using the excess burst formula, the excess burst is 8000 bytes.

be = 2 * 4000 = 8000

The default excess burst size is 0.

Deciding if Packets Conform or Exceed the Committed Rate

Policing uses normal or committed burst (bc) and excess burst (be) values to ensure that the configured committed information rate (CIR) is reached. Policing decides if a packet conforms or exceeds the CIR based on the burst values you configure. Several factors can influence the policer's decision, such as the following:

- Low burst values—If you configure burst values too low, the achieved rate might be much lower than the configured rate.

- Temporary bursts—These bursts can have a strong adverse impact on throughput of Transmission Control Protocol (TCP) traffic.

It is important that you set the burst values high enough to ensure good throughput. If your router drops packets and reports an exceeded rate even though the conformed rate is less than the configured CIR, use the show interface command to monitor the current burst, determine whether the displayed value is consistently close to the committed burst (bc) and excess burst (be) values, and if the actual rates (the committed rate and exceeded rate) are close to the configured committed rate. If not, the burst values might be too low. Try reconfiguring the burst rates using the suggested calculations in the Committed Burst Calculation, on page 5 and the Excess Burst Calculation, on page 6.

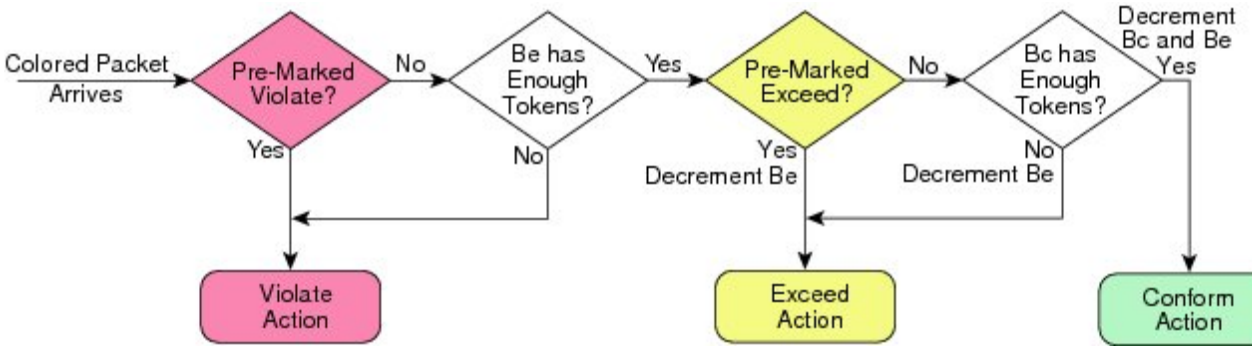
Two-Rate Three-Color (2R3C) Policer

The policer reads a preexisting marking—the frame-relay discard-eligibility (FRDE) bit in the packet header—that was set by a policer on a previous network node. By default the FRDE bit is set to 0. At the receiving node, the system uses this bit to determine the appropriate color-aware policing action for the packet:

- To classify the FRDE bit value 0 as conform color, create a conform-color class-map for frde=0 packets. This causes packets to be classified as color green, and the system applies the conform action.
- To classify the FRDE bit value 1 as exceed color, create an exceed-color class-map for frde=1 packets. This causes packets to be classified as color yellow, and the system applies the exceed action.

Note Color-aware policing is not supported for hierarchical QoS.

Figure 3: 2R3C Policing Process Flowchart



Configuring Traffic Policing (Two-Rate Color-Blind)

Traffic policing allows you to control the maximum rate of traffic sent or received on an interface. This section provides the procedure for configuring two-rate color-blind traffic policing.

Procedure

Step 1 **configure**

Step 2 **policy-map** *policy-name*

Example:

```
RP/0/(config)# policy-map policy1
```

Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy and enters the policy map configuration mode.

Step 3 **class** *class-name*

Example:

```
RP/0/(config-pmap)# class class1
```

Specifies the name of the class whose policy you want to create or change and enters the policy map class configuration mode.

Step 4 **police rate** {[*units*] | **percent** *percentage*} [**burst** *burst-size* [burst-*units*]] [**peak-burst** *peak-burst* [burst-*units*]] [**peak-rate** *value* [*units*]] | **percent** *percentage*]

Example:

```
RP/0/(config-pmap-c)# police rate 250000
```

Configures traffic policing and enters policy map police configuration mode. The traffic policing feature works with a token bucket algorithm.

Step 5 **exit**

Example:

```
RP/0/(config-pmap-c-police)# exit
```

Returns the router to policy map class configuration mode.

Step 6 **exit**

Example:

```
RP/0/(config-pmap-c)# exit
```

Returns the router to policy map configuration mode.

Step 7 **exit**

Example:

```
RP/0/(config-pmap)# exit
```

Returns the router to the global configuration mode.

Step 8 **interface** *type interface-path-id*

Example:

```
RP/0/(config)# interface HundredGigE 0/7/0/0
```

Enters configuration mode and configures an interface.

Step 9 **service-policy** {**input** | **output**} *policy-map*

Example:

```
RP/0/(config-if)# service-policy output policy1
```

Attaches a policy map to an input or output interface to be used as the service policy for that interface. In this example, the traffic policy evaluates all traffic leaving that interface.

Step 10 **commit**

Step 11 **show policy-map interface** *type interface-path-id* [**input** | **output**]

Example:

```
RP/0/# show policy-map interface HundredGigE 0/7/0/0
```

(Optional) Displays policy configuration information for all classes configured for all service policies on the specified interface.

Configuring Traffic Policing (2R3C)

This section provides the procedure for configuring two-rate three-color traffic policing.

Procedure

Step 1 **configure**

Step 2 **class-map** [**match-any**] *class-map-name*

Example:

```
RP/0/(config)# class-map match-all
```

Creates or modifies a class map that can be attached to one or more interfaces to specify a matching policy and enters the class map configuration mode.

Step 3 **match precedence ipv4***precedence_value***fr-de***fr-de-bit-value*

Example:

```
RP/0/(config)# match  precedence ipv4 5
```

Specifies a precedence value that is used as the match criteria against which packets are checked to determine if they belong to the class specified by the class map.

Specifies the matching condition:

- Match fr-de 1 is typically used to specify an exceed-color packet.

Step 4 **policy-map** *policy-name*

Example:

```
RP/0/(config)# policy-map policy1
```

Creates or modifies a policy map that can be attached to one or more interfaces to specify a service policy and enters the policy map configuration mode.

Step 5 **class** *class-name*

Example:

```
RP/0/(config-pmap)# class class1
```

Specifies the name of the class whose policy you want to create or change and enters the policy map class configuration mode.

Step 6 **police rate** {[*units*] | **percent** *percentage*} [**burst** *burst-size* [burst-*units*]] [**peak-burst** *peak-burst* [burst-*units*]] [**peak-rate** *value* [*units*]]

Example:

```
RP/0/(config-pmap-c)# police rate 768000 burst 288000 peak-rate 1536000 peak-burst 576000
```

Configures traffic policing and enters policy map police configuration mode. The traffic policing feature works with a token bucket algorithm.

Step 7 **exit**

Example:

```
RP/0/(config-pmap-c-police)# exit
```

Returns the router to policy map class configuration mode.

Step 8 **exit**

Example:

```
RP/0/(config-pmap-c)# exit
```

Returns the router to policy map configuration mode.

Step 9 **exit**

Example:

```
RP/0/(config-pmap)# exit
```

Returns the router to global configuration mode.

Step 10 **interface** *type interface-path-id*

Example:

```
RP/0/(config)# interface HundredGigE 0/7/0/0
```

Enters configuration mode and configures an interface.

Step 11 **service-policy** *policy-map*

Example:

```
RP/0/(config-if)# service-policy policy1
```

Attaches a policy map to an input interface to be used as the service policy for that interface.

Step 12 **commit**

Step 13 **show policy-map interface** *type interface-path-id*

Example:

```
RP/0/# show policy-map interface HundredGigE 0/7/0/0
```

(Optional) Displays policy configuration information for all classes configured for all service policies on the specified interface.

Running configuration for policer

```
policy-map ingress_POLICER_POLICY
class CLASS_1_POLICERIPV4PREC
  set qos-group 7
  police rate 1 gbps peak-rate 2 gbps
!
```