

[Solutions](#) [Products](#) [Ordering](#) [Support](#) [Partners](#) [Training](#) [Corporate](#)

Tech Notes



Comparing Traffic Policing and Traffic Shaping for Bandwidth Limiting

[TAC Notice: What's Changing on TAC Web](#)

Document ID: 19645

Contents

[Introduction](#)
[Before You Begin](#)
[Conventions](#)
[Prerequisites](#)
[Components Used](#)
[Policing Versus Shaping](#)
[Selection Criteria](#)
[Token Refresh Rate](#)
[Traffic Shaping](#)
[Traffic Policing](#)
[Minimum Versus Maximum Bandwidth Controls](#)
[Related Information](#)

Help us help you.

Please rate this document.

- Excellent
 Good
 Average
 Fair
 Poor

This document solved my problem.

- Yes
 No
 Just browsing

Suggestions for improvement:

(256 character limit)

Introduction

This document clarifies the functional differences between shaping and policing, both of which limit the output rate. Though both mechanisms use a token bucket as a traffic meter to measure the packet rate, they have important functional differences. (A token bucket is described in [What Is a Token Bucket?](#)).

Before You Begin

Conventions

For more information on document conventions, see the [Cisco Technical Tips Conventions](#).

Prerequisites

There are no specific prerequisites for this document.

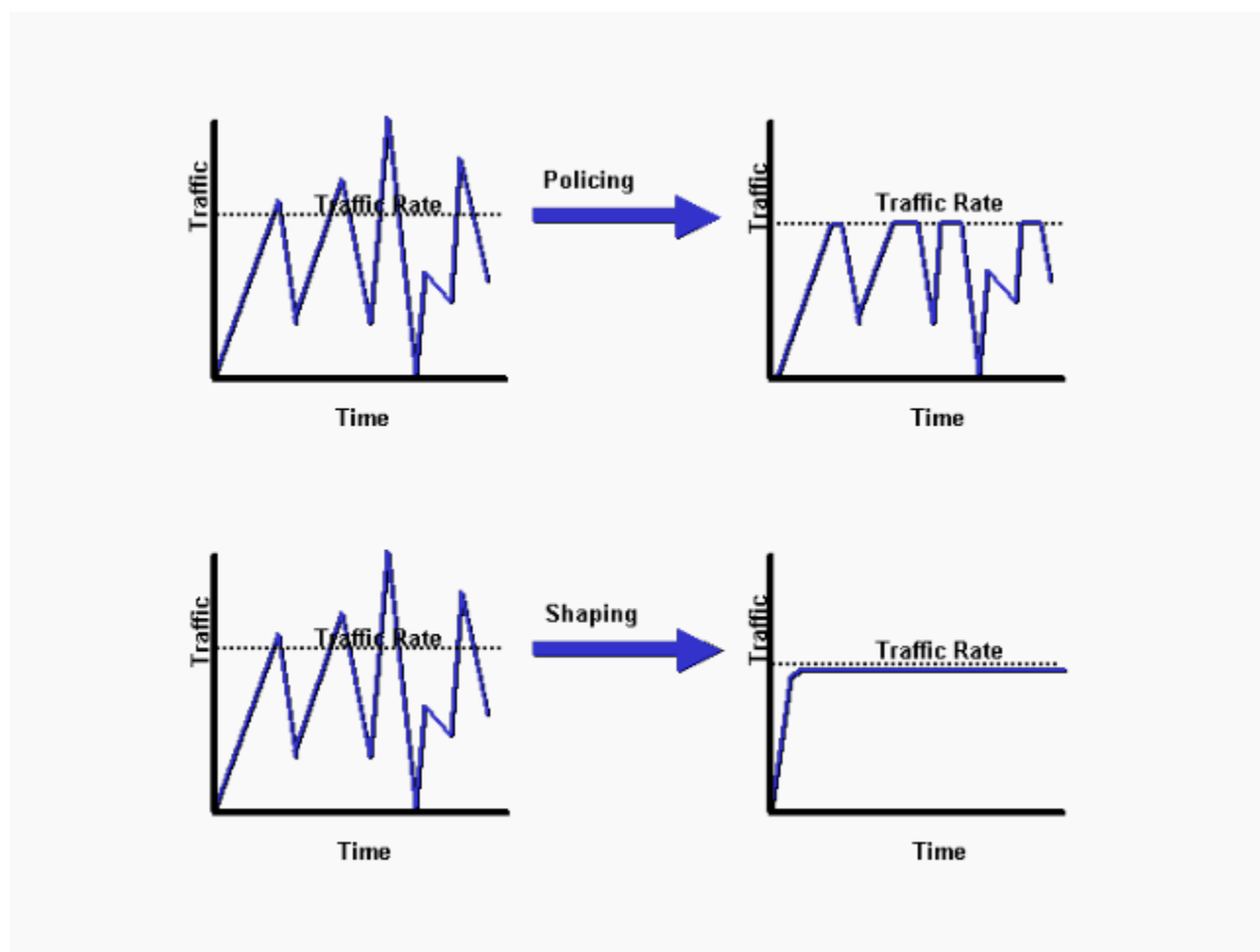
Components Used

This document is not restricted to specific software and hardware versions.

The information presented in this document was created from devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If you are working in a live network, ensure that you understand the potential impact of any command before using it.

Policing Versus Shaping

The following diagram illustrates the key difference. Traffic policing propagates bursts. When the traffic rate reaches the configured maximum rate, excess traffic is dropped (or remarked). The result is an output rate that appears as a saw-tooth with crests and troughs. In contrast to policing, traffic shaping retains excess packets in a queue and then schedules the excess for later transmission over increments of time. The result of traffic shaping is a smoothed packet output rate.



Shaping implies the existence of a queue and of sufficient memory to buffer delayed packets, while policing does not. Queueing is an outbound concept; packets going out an interface get queued and can be shaped. Only policing can be applied to inbound traffic on an interface. Ensure that you have sufficient memory when enabling shaping. In addition, shaping requires a scheduling function for later transmission of any delayed packets. This scheduling function allows you to organize the shaping queue into different queues. Examples of scheduling functions are Class Based Weighted Fair Queuing (CBWFQ) and Low Latency Queuing (LLQ).

Selection Criteria

The following table lists the differences between shaping and policing to help you choose the best solution.

| | Shaping | Policing |
|-----------------------|--|--|
| Objective | Buffer and queue excess packets above the committed rates. | Drop (or remark) excess packets above the committed rates. Does not buffer.* |
| Token Refresh Rate | Incremented at the start of a time interval. (Minimum number of intervals is required.) | Continuous based on formula: $1 / \text{committed information rate}$ |
| Token Values | Configured in bits per second. | Configured in bytes. |
| Configuration Options | <ul style="list-style-type: none"> • shape command in the modular quality of service command-line interface (MQC) to implement class-based shaping. • frame-relay traffic-shape command to implement Frame Relay Traffic Shaping (FRTS). • traffic-shape command to implement Generic Traffic Shaping (GTS). | <ul style="list-style-type: none"> • police command in the MQC to implement class-based policing. • rate-limit command to implement committed access rate (CAR). |
| Applicable on Inbound | No | Yes |

| Applicable on Outbound | Yes | Yes |
|---------------------------|---|--|
| Bursts | Controls bursts by smoothing the output rate over at least eight time intervals. Uses a leaky bucket to delay traffic, which achieves a smoothing effect. | Propagates bursts. Does no smoothing. |
| Advantages | Less likely to drop excess packets since excess packets are buffered. (Buffers packets up to the length of the queue. Drops may occur if excess traffic is sustained at high rates.) Typically avoids retransmissions due to dropped packets. | Controls the output rate through packet drops. Avoids delays due to queuing. |
| Disadvantages | Can introduce delay due to queuing, particularly deep queues. | Drops excess packets (when configured), throttling TCP window sizes and reducing the overall output rate of affected traffic streams. Overly aggressive burst sizes may lead to excess packet drops and throttle the overall output rate, particularly with TCP-based flows. |
| Optional Packet Remarking | No | Yes (with legacy CAR feature). |

* Although policing does not apply buffering, a configured queuing mechanism applies to "conformed" packets that may need to be queued while waiting to be serialized at the physical interface.

Token Refresh Rate

A key difference between shaping and policing is the rate at which tokens are replenished. This section

reviews the difference.

Simply stated, both shaping and policing use the token bucket metaphor. A token bucket itself has no discard or priority policy. Let's look at how the token bucket metaphor works:

- Tokens are put into the bucket at a certain rate.
- Each token is permission for the source to send a certain number of bits into the network.
- To send a packet, the traffic regulator must be able to remove from the bucket a number of tokens equal in representation to the packet size.
- If not enough tokens are in the bucket to send a packet, the packet either waits until the bucket has enough tokens (in the case of a shaper) or the packet is discarded or marked down (in the case of a policer).
- The bucket itself has a specified capacity. If the bucket fills to capacity, newly arriving tokens are discarded 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. A token bucket permits burstiness, but bounds it.

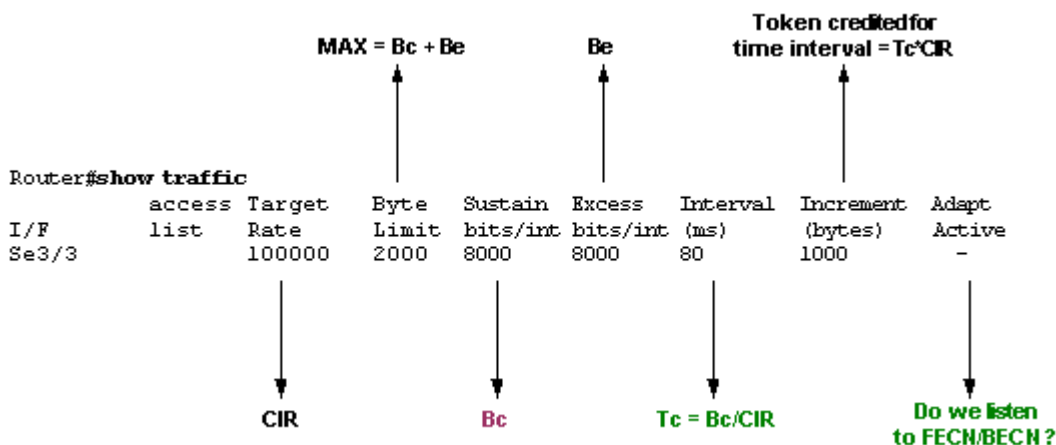
With the token bucket metaphor in mind, let's look at how shaping and policing add tokens to the bucket.

Shaping increments the token bucket at timed intervals using a bits per second (bps) value. A shaper uses the following formula:

$$T_c = B_c / CIR \text{ (in seconds)}$$

In this equation, B_c represents the committed burst, and CIR stands for committed information rate. (See [Configuring Frame Relay Traffic Shaping](#) for more information.) The value of T_c defines the time interval during which you send the B_c bits in order to maintain the average rate of the CIR in seconds.

The range for T_c is between 10 ms and 125 ms. With distributed traffic shaping (DTS) on the Cisco 7500 series, the minimum T_c is 4 ms. The router internally calculates this value based on the CIR and B_c values. If B_c / CIR is less than 125 ms, it uses the T_c calculated from that equation. If B_c / CIR is more than or equal to 125 ms, it uses an internal T_c value if Cisco IOS[®] determines that traffic flow will be more stable with a smaller interval. Use the **show traffic-shape** command to determine whether your router is using an internal value for T_c or the value that you configured at the command-line. The following sample output of the **show traffic-shape** command is explained in [show Commands for Frame Relay Traffic Shaping](#).



When the excess burst (Be) is configured to a value different than 0, the shaper allows tokens to be stored in the bucket, up to Bc + Be. The largest value that the token bucket can ever reach is Bc + Be, and overflow tokens are dropped. The only way to have more than Bc tokens in the bucket is to not use all Bc tokens during one or more Tc. Since the token bucket is replenished every Tc with Bc tokens, you can accumulate unused tokens for later use up to Bc + Be.

In contrast, class-based policing and rate-limiting adds tokens continuously to the bucket. Specifically, the token arrival rate is calculated as follows:

$$(\text{time between packets} < \text{which is equal to } t-t_1 > * \text{ policer rate}) / 8 \text{ bits per byte}$$

In other words, if the previous arrival of the packet was at t1 and the current time is t, the bucket is updated with t-t1 worth of bytes based on the token arrival rate. Note that a traffic policer uses burst values specified in bytes, and the above formula converts from bits to bytes.

Look at an example using a CIR (or policer rate) of 8000 bps and a normal burst of 1000 bytes.

```
Router(config)# policy-map police-setting
Router(config-pmap)# class access-match
Router(config-pmap-c)# police 8000 1000 conform-action transmit exceed-action d
```

The token bucket starts full at 1000 bytes. If a 450 byte packet arrives, the packet conforms because enough bytes are available in the token bucket. The conform action (transmit) is taken by the packet and 450 bytes are removed from the token bucket (leaving 550 bytes). If the next packet arrives .25 seconds later, 250 bytes are added to the token bucket as per the following formula:

$$(0.25 * 8000) / 8$$

The calculation leaves 700 bytes in the token bucket. If the next packet is 800 bytes, the packet exceeds and the exceed action (drop) is taken. No bytes are taken from the token bucket.

Traffic Shaping

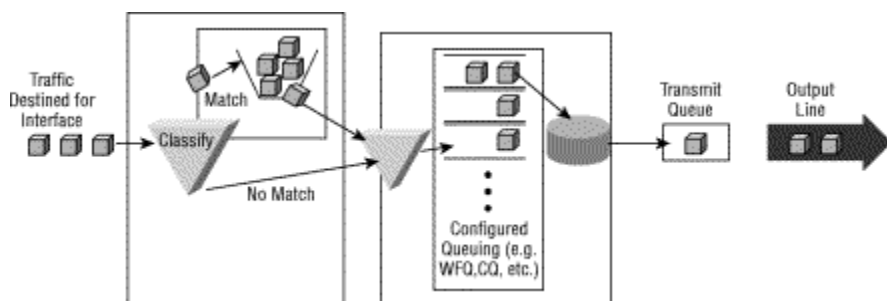
Cisco IOS supports the following methods of traffic shaping:

- [Generic Traffic Shaping](#)

- [Frame Relay Traffic Shaping](#)
- [Class-Based Shaping](#) and [Distributed Class-Based Shaping](#)

All traffic shaping methods are similar in implementation, though their command-line interfaces (CLIs) differ somewhat, and they use different types of queues to contain and shape traffic that is deferred. Cisco recommends class-based shaping and distributed shaping, which are configured using the modular QoS CLI.

The following diagram illustrates how a QoS policy sorts traffic into classes and queues packets that exceed the configured shaping rates.



Traffic Policing

Cisco IOS supports the following methods of traffic policing:

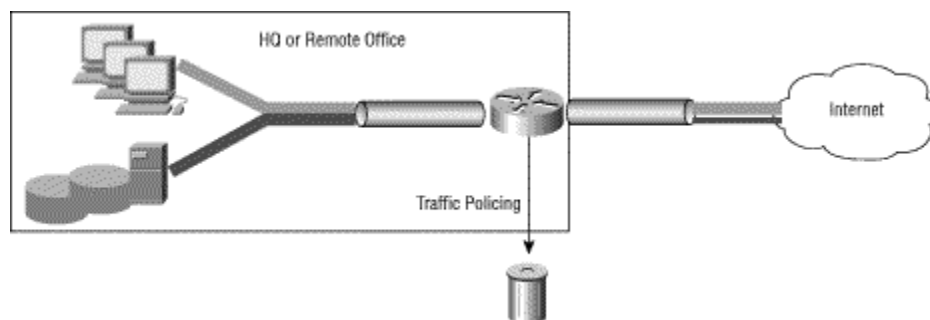
- [Committed Access Rate](#)
- [Class-Based Policing](#)

The two mechanisms have important functional differences, as explained in [Comparing Class-Based Policing and Committed Access Rate](#). Cisco recommends class-based policing and other features of the modular QoS CLI when applying QoS policies.

Use the **police** command to specify that a class of traffic should have a maximum rate imposed on it, and if that rate is exceeded, an immediate action must be taken. In other words, with the **police** command, it is not an option to buffer the packet and later send it out, as is the case for the **shape** command.

In addition, with policing, the token bucket determines whether a packet exceeds or conforms to the applied rate. In either case, policing implements a configurable action, which includes setting the IP precedence or Differentiated Services Code Point (DSCP).

The following diagram illustrates a common application of traffic policing at a congestion point, where QoS features generally apply.



Minimum Versus Maximum Bandwidth Controls

Both the **shape** and **police** commands restrict the output rate to a maximum kbps value. Importantly, neither mechanism provides a minimum bandwidth guarantee during periods of congestion. Use the **bandwidth** or **priority** command to provide such guarantees.

A hierarchical policy uses two service policies – a parent policy to apply a QoS mechanism to a traffic aggregate and a child policy to apply a QoS mechanism to a flow or subset of the aggregate. Logical interfaces, such as subinterfaces and tunnel interfaces, require a hierarchical policy with the traffic-limiting feature at the parent level and queuing at lower levels. The traffic-limiting feature reduces the output rate and (presumably) creates congestion, as seen by queuing excess packets.

The following configuration is sub-optimal and is shown to illustrate the difference between the **police** versus the **shape** command when limiting a traffic aggregate – in this case class-default – to a maximum rate. In this configuration, the **police** command sends packets from the child classes based on the size of the packet and the number of bytes remaining in the conform and exceed token buckets. (See [Traffic Policing](#).) The result is that rates given to the Voice over IP (VoIP) and Internet Protocol (IP) classes may not be guaranteed since the **police** feature is overriding the guarantees made by the **priority** feature.

However, if the **shape** command is used, the result is a hierarchical queuing system, and all guarantees are made. In other words, when the offered load exceeds the shape rate, the VoIP and IP classes are guaranteed their rate, and the class-default traffic (at the child level) incurs any drops.



Caution: This configuration is not recommended and is shown to illustrate the difference between the **police** versus the **shape** command when limiting a traffic aggregate.

```
class-map match-all IP
  match ip precedence 3
class-map match-all VoIP
  match ip precedence 5

policy-map child
  class VoIP
    priority 128
  class IP
    priority 1000

policy-map parent
  class class-default
    police 3300000 103000 103000 conform-action transmit exceed-action drop
    service-policy child
```

In order for the above configuration to make sense, the policing should be replaced by shaping. For example:

```
policy-map parent
  class class-default
    shape average 3300000 103000 0
  service-policy child
```

In order to learn more about parent and child policies, please refer to [Configuring CBWFQ Inside GTS](#).

Related Information

- [QoS Support Page](#)
 - [Technical Support - Cisco Systems](#)
-

| | | | | | | |
|----------------------|----------------------------|-----------------------|-------------------------|--------------------------|--------------------------|----------------------|
| Home | How to Buy | Login | Profile | Feedback | Site Map | Help |
|----------------------|----------------------------|-----------------------|-------------------------|--------------------------|--------------------------|----------------------|

All contents are Copyright © 1992-2003 Cisco Systems, Inc. All rights reserved. [Important Notices](#) and [Privacy Statement](#).

Updated: Dec 17, 2003

Document ID: 19645
