Understanding QoS Policing and Marking on the Catalyst 3550

Document ID: 24800

Contents

Introduction
Prerequisites
Requirements
Components Used
Conventions
Hardware and Software Versions
QoS Policing and Marking Parameters
Policing and Marking Features Supported by the Catalyst 3550
Configure and Monitor Policing
Configure and Monitor Marking
How to Classify All Interface Traffic with a Single Policer
Related Information

Introduction

The policing function determines if the traffic level is within the specified profile or contract, and allows you to either drop out−of−profile traffic or mark it down to a different Differential Services Code Point (DSCP) value. This enforces a contracted service level.

DSCP is a measure of the Quality of Service (QoS) level of the packet. Along with DSCP, IP precedence and Class of Service (CoS) are also used in order to convey the QoS level of the packet.

Policing is not to be confused with traffic shaping, although both make sure the traffic stays within the profile or contract.

Policing does not buffer the traffic, so policing does not affect the transmission delay. Instead of buffering out−of−profile packets, policing drops them or marks them with different QoS levels (DSCP markdown).

Traffic shaping buffers out−of−profile traffic and smoothes the traffic bursts, but affects the delay and delay variation. Shaping can only be applied on the outgoing interface, while policing can be applied on both the incoming and outgoing interface.

The Catalyst 3550 supports policing for both incoming and outgoing directions. Traffic shaping is not supported.

Marking changes the packet QoS level according to a policy.

Prerequisites

Requirements

There are no specific requirements for this document.

Components Used

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

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

Refer to Cisco Technical Tips Conventions for more information on document conventions.

Hardware and Software Versions

Policing and marking on the Catalyst 3550 is supported with all software versions. The latest configuration guide is listed here. Refer to this documentation for all supported features.

- Configuring QoS

QoS Policing and Marking Parameters

In order to set up policing, you must define the QoS policy maps and apply them to ports. This is otherwise known as port−based QoS.

Note: VLAN−based QoS is currently not supported by the Catalyst 3550.

The policer is defined by rate and burst parameters as well as action for out−of−profile traffic.

These two types of policers are supported:

- Aggregate
- Individual

The aggregate policer acts upon the traffic across all instances where it is applied. The individual policer acts separately upon traffic across each instance where it is applied.

Note: On the Catalyst 3550, the aggregate policer can only be applied to different classes of the same policy. Aggregate policing across multiple interfaces or policies is not supported.

For example, apply the aggregate policer in order to limit the traffic of class customer1 and class customer2 in the same policy−map to 1 Mbps. Such a policer allows 1 Mbps of traffic in class customer1 and customer2 together. If you apply the individual policer, the policer limits the traffic for class customer1 to 1 Mbps and for class customer2 to 1 Mbps. Therefore, each instance of the policer is separate.

This table summarizes the QoS action upon the packet when treated by both ingress and egress policies:

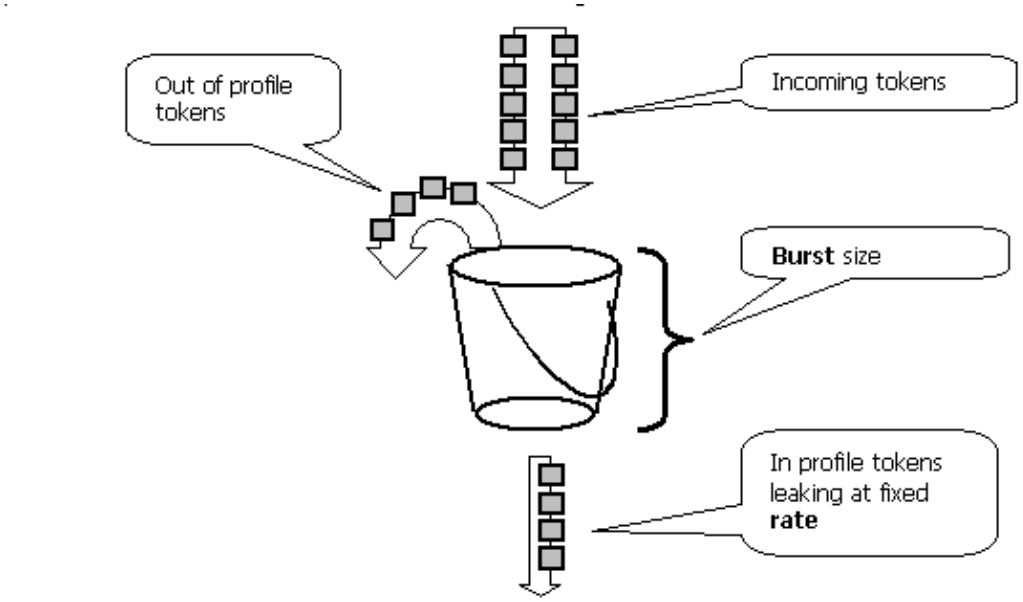
	Ingress policy			
Egress policy	Transmit	Drop	$Markdown_i$	$Mark_i$
Transmit	Transmit	Drop	$Markdown_i$	$Mark_i$
Drop	Drop	Drop	Drop	Drop
$Markdown_e$	$Markdown_e$	Drop	$Markdown_i$ then $Markdown_e$	$Mark_i$ then $Markdown_e$

Note: It is possible to mark and markdown within the same traffic class of the same policy. In such a case, all traffic for the particular class is marked first. Policing and markdown occurs on already marked traffic.

QoS policing in the Catalyst 3550 complies with this leaky bucket concept:

The number of tokens proportional to the incoming traffic packet sizes are placed into a token bucket; the number of tokens equals the size of the packet. At a regular interval, a defined number of tokens derived from the configured rate is removed from the bucket. If there is no place in the bucket to accommodate an incoming packet, the packet is considered out−of−profile and is dropped or marked down according to the configured policing action.

This concept is shown in this example:



Note: The traffic is not buffered in the bucket as it can appear in this example. The actual traffic does not flow through the bucket at all; the bucket is only used in order to decide whether the packet is in profile or out−of−profile.

Note: Hardware implementation of policing can vary, but functionally it still complies to this model.

These parameters control the operation of policing:

- **Rate** defines how many tokens are removed at each interval. This effectively sets the policing rate. All traffic below the rate is considered in profile. Supported rates range from 8 Kbps to 2 Gbps, and increment by 8 Kbps.
- **Interval** defines how often tokens are removed from the bucket. The interval is fixed at 0.125 milliseconds (or 8000 times per second). This interval cannot be changed.
- **Burst** defines the maximum amount of tokens the bucket can hold at any time. Supported bursts range from 8000 bytes to to 2000000 bytes, and increment by 64 bytes.

Note: Although the command−line help strings show a large range of values, the rate−bps option cannot exceed the configured port speed, and the burst−byte option cannot exceed 2000000 bytes. If you enter a larger value, the switch rejects the policy map when you attach it to an interface.

In order to sustain the specified traffic rate, the burst must be no less than the sum of this equation:

```
Burstmin (bits) = Rate (bps) / 8000 (1/sec)
```

For example, calculate the minimum burst value in order to sustain a rate of 1 Mbps. The rate is defined as 1000 Kbps, so the minimum burst needed is the sum of this equation:

```
1000 (Kbps) / 8000 (1/sec) =125 (bits)
```

The minimum supported burst size is 8000 bytes, which is more than the minimum burst calculated.

Note: Due to hardware policing granularity, the exact rate and burst is rounded to the nearest supported value.

When you configure the burst rate, you must take into account that some protocols implement mechanisms that react to the packet loss. For instance, Transmission Control Protocol (TCP) reduces the window by half for each lost packet. This causes a "saw tooth" effect in the TCP traffic when TCP tries to accelerate to the line rate and is throttled by the policer. If the average rate of the saw tooth traffic is calculated, this rate is much lower than the policed rate. However, you can increase the burst in order to achieve better utilization. A good start is to set the burst equal to twice the amount of the traffic sent with the desired rate during Round−Trip Time (TCP RTT). If RTT is not known, you can double the value of the burst parameter.

For the same reason, it is not recommended to benchmark the policer operation by connection−oriented traffic. This scenario generally shows lower performance than permitted by the policer.

Connectionless traffic can also react to policing differently. For example, Network File System (NFS) uses blocks, which could consist of more than one User Datagram Protocol (UDP) packet. One packet dropped can trigger many packets, even the entire block, to be retransmitted.

This example calculates the burst for a TCP session with a policing rate of 64 Kbps and given the TCP RTT is, 0.05 seconds:

```
<burst> = 2*  *  = 2 * 0.05 [sec] * 64000/8 [bytes/sec] = 800    [bytes]
```

In this example, `<burst>` is for one TCP session. Scale this figure to average the expected number of sessions traveling through the policer.

Note: This is an example only, in each case you need to evaluate traffic and application requirements and behavior versus available resources in order to choose policing parameters.

The policing action can be either to drop the packet or change the DSCP of the packet (markdown). In order to markdown the packet, a policed DSCP map must be modified. A default policed DSCP map remarks the packet to the same DSCP. Therefore, no markdown occurs.

Packets can be sent out of order when an out−of−profile packet is marked down to a DSCP mapped into a different output queue than the original DSCP. If the order of the packets is important, markdown out−of−profile packets to the DSCP mapped to the same output queue as in−profile packets.

Policing and Marking Features Supported by the Catalyst 3550

This table provides a summary of the policing and marking related features supported by the Catalyst 3550, broken down by direction:

	Direction	
Feature	Ingress	Egress
Individual policers	Yes, totally 128 for GE and 8 for FE including ingress aggregate policers	Yes, totally 8 including egress aggregate policers
Aggregate policers	Yes, totally 128 for GE and 8 for FE including ingress individual policers	Yes, totally 8 including egress individual policers
Marking	Yes	No
Policer Markdown	Yes	Yes
Match with ACL	Yes	No
Match DSCP	Yes	Yes
Match IP precedence	Yes	No
Match COS	Yes, for non-IP traffic	No
Trust DSCP	Yes	No
Trust COS	Yes	No
Trust IP precedence	Yes	No

One match statement is supported per class−map. These are valid match statements for the ingress policy:

- match access−group
- match ip dscp
- match ip precedence

Note: On the Catalyst 3550, the **match interface** command is not supported and only one match command is allowed in a class−map. Therefore, it is tricky to classify all the traffic that comes in through an interface and police all traffic with a single policer. See How to Classify All Interface Traffic with a Single Policer section of this document.

This is the valid match statement for the egress policy:

- match ip dscp

These are valid policy actions for the ingress policy:

- police
- set ip dscp (marking)
- set ip precedence (marking)
- trust dscp
- trust ip−precedence
- trust cos

This table shows the supported ingress QoS policies matrix:

Trust I/F	Match DSCP[1]	Match ACL	Trust Class[2]	Set DSCP[3]	Police	Result
						Traffic is assigned default QOS level of the port (0 by default)
√						QOS level of incoming traffic is preserved, according to what is trusted
	√		√		√	IP Traffic is matched by DSCP and then trusted then policed, excess traffic dropped or marked down
	√		√			IP Traffic is matched by DSCP/IP precedence and its QOS level is preserved
	√			√		IP Traffic is matched by DSCP/IP precedence then marked
	√			√	√	IP Traffic is matched by DSCP/IP precedence then marked then policed
		√	√		√	Traffic is matched by access list, QOS level of the matched traffic is preserved, then traffic is policed
		√	√			Traffic is matched by access list and its QOS level is preserved according to what is trusted
		√		√	√	Traffic is matched by access list then marked and then policed
		√		√		Traffic is matched by ACL then marked with specified DSCP/IP precedence
		MAC ACL w/COS	√			Match non-IP traffic by MAC EtherType and COS and preserve QOS level
		MAC ACL w/COS	√		√	Match non-IP IP traffic by MAC EtherType and COS and preserve QOS level then police
		MAC ACL w/COS		√		Match non-IP IP traffic by MAC EtherType and COS then mark matched traffic
		MAC ACL W/COS		√	√	Match non-IP IP traffic by MAC EtherType and COS then mark and then police

1. This option also covers the match IP precedence.
2. This option covers trusting CoS, IP precedence, and DSCP.
3. This option also covers setting the IP precedence.

This is the valid policy action for the egress policy:

- police

This table shows the supported egress QoS policies matrix:

Match DSCP	Police	Result
		Traffic is sent out with COS and IP precedence according to QOS maps and internal DSCP after ingress QOS processing
√	√	Traffic is matched by DSCP and policed

Marking allows the QoS level of the packet to change based upon classification or policing. Classification splits traffic into different classes for QoS processing based on the defined criteria.

QoS processing is based on the internal DSCP; the measure of the QoS level of the packet. Internal DSCP is derived according to the trust configuration. The system supports trusting CoS, DSCP, IP precedence, and

untrusted interfaces. Trust specifies the field from which the internal DSCP is derived for each packet, as follows:

- When trusting CoS, the QoS level is derived from the Layer 2 (L2) header of the Inter−Switch Link Protocol (ISL) or the 802.1Q encapsulated packet.
- When trusting DSCP or IP precedence, the system derives the QoS level from the DSCP or IP precedence field of the packet accordingly.

Trusting CoS is only meaningful on trunking interfaces, and trusting DSCP (or IP precedence) makes sense for IP packets only.

When an interface is not trusted, the internal DSCP is derived from the configurable default CoS for the corresponding interface. This is the default state when QoS is enabled. If no default CoS is configured, the default value is zero.

Once the internal DSCP is determined, it can be changed by marking and policing, or retained.

After the packet undergoes the QoS processing, its QoS level fields (within the IP/DSCP field for IP, and within the ISL/802.1Q header, if any) are updated from the internal DSCP. There are these special QoS maps relevant to policing:

- **DSCP−to−Policed DSCP** used in order to derive the policed DSCP when you markdown down the packet.
- **DSCP−to−CoS** used in order to derive the CoS level from the internal DSCP to update the outgoing packet ISL/802.1Q header.
- **CoS−to−DSCP** used in order to derive the internal DSCP from the incoming CoS (ISL/802.1Q header) when the interface is in the trust CoS mode.

These are important implementation−specific considerations:

- The ingress service policy cannot be attached to the interface when the interface is configured to trust any of the QoS metrics, such as CoS/DSCP or IP precedence. In order to match on DSCP/IP precedence and police on ingress, you must configure trust for the particular class within the policy, not on the interface. In order to mark based on DSCP/IP precedence, no trust must be configured.
- Only IPv4 traffic with no IP options and Ethernet II Advanced Research Projects Agency (ARPA) encapsulation is considered IP traffic from the hardware and QoS standpoint. All other traffic is considered non−IP including, IP with options, such as SubNetwork Access Protocol (SNAP) encapsulated IP and IPv6.
- For non−IP packets, "match access group" is the only method of classification because you cannot match DSCP for non−IP traffic. A Media Access Control (MAC) Access List (ACL) is used for that purpose; packets can be matched based on the source MAC address, the destination MAC address, and EtherType. It is not possible to match the IP traffic with the MAC ACL, since the switch makes a distinction between IP and non−IP traffic.

Configure and Monitor Policing

These steps are necessary in order to configure policing in Cisco IOS:

1. Define a policer (for aggregate policers)
2. Define criteria to select traffic for policing
3. Define a class−map to select traffic using defined criteria
4. Define a service−policy using class and applying a policer to the specified class
5. Apply a service−policy to a port

These two types of policers are supported:

- Named aggregate
- Individual

The named aggregate policer polices the traffic combined from all classes within the same policy to where it is applied. Aggregate policing across different interfaces is not supported.

Note: The aggregate policer cannot be applied to more than one policy. If it is, this error message is displayed:

```
QoS: Cannot allocate policer for policy map <policy name>
```

Consider this example:

There is a traffic generator attached to port GigabitEthernet0/3 that sends approximately 17 Mbps of UDP traffic with the destination port 111. There is also TCP traffic from port 20. You want these two traffic streams to be policed down to 1 Mbps, and excessive traffic must be dropped. This example shows how this is done:

```
!--- Globally enables QoS.

mls qos

!--- Defines the QoS policer, sets the burst
!--- to 16000 for better TCP performance.

mls qos aggregate-policer pol_1mbps 1000000 16000 exceed-action drop

!--- Defines the ACLs to select traffic.

access-list 123 permit udp any any eq 111
access-list 145 permit tcp any eq 20 any

!--- Defines the traffic classes to be policed.

class-map match-all cl_udp111
  match access-group 123
class-map match-all cl_tcp20
  match access-group 145

!--- Defines the QoS policy, and attaches
!--- the policer to the traffic classes.

policy-map po_test
  class cl_udp111
    police aggregate pol_1mbps
  class cl_tcp20
    police aggregate pol_1mbps

!--- Applies the QoS policy to an interface.

interface GigabitEthernet0/3
 switchport
 switchport access vlan 2
 service-policy input po_test
!
```

The first example used the named aggregate policer. The individual policer, unlike the named policer, polices traffic separately on each class where it is applied. The individual policer is defined within the policy map

configuration. In this example, two classes of traffic are policed by two individual policers; cl_udp111 is policed to 1 Mbps per 8K burst, and cl_tcp20 is policed to 512 Kbps per 32K burst:

```
!--- Globally enables QoS.

mls qos

!--- Defines the ACLs to select traffic.

access-list 123 permit udp any any eq 111
access-list 145 permit tcp any eq 20 any

!--- Defines the traffic classes to be policed.

class-map match-all cl_udp111
  match access-group 123
class-map match-all cl_tcp20
  match access-group 145

!--- Defines QoS policy, and creates and attaches
!--- the policers to the traffic classes.

policy-map po_test2
  class cl_udp111
    police 1000000 8000 exceed-action drop
  class cl_tcp20
    police 512000 32000 exceed-action drop

!--- Applies the QoS policy to an interface.

interface GigabitEthernet0/3
 switchport
 switchport access vlan 2
 service-policy input po_test2
```

This command is used in order to monitor the policing operation:

```
cat3550#show mls qos interface g0/3 statistics
GigabitEthernet0/3
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
Others: 267718     0          267717     0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
Others: 590877        n/a       n/a      266303     0

WRED drop counts:
  qid  thresh1    thresh2   FreeQ
   1 : 0          0         1024
   2 : 0          0         1024
   3 : 0          0         8
   4 : 0          0         1024
```

Note: By default, there are no per−DSCP statistics. The Catalyst 3550 supports a per−interface, per−direction statistics collection for up to eight different DSCP values. This is configured when you issue the **mls qos monitor** command. In order to monitor statistics for DSCPs 8, 16, 24, and 32, you must issue this **per−interface** command:

```
cat3550(config-if)#mls qos monitor dscp 8 16 24 32
```

Note: The **mls qos monitor dscp 8 16 24 32** command changes the output of the **show mls qos int g0/3 statistics** command to this:

```
cat3550#show mls qos interface g0/3 statistics
GigabitEthernet0/3
Ingress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    8 : 0          0          675053785  0          0
    16: 1811748    0          0          0          0         ? per DSCP statistics
    24: 1227820404 15241073   0          0          0
    32: 0          0          539337294  0          0
Others: 1658208    0          1658208    0          0
Egress
  dscp: incoming   no_change  classified policed    dropped (in pkts)
    8 : 675425886     n/a       n/a      0          0
    16: 0             n/a       n/a      0          0         ? per DSCP statistics
    24: 15239542      n/a       n/a      0          0
    32: 539289117     n/a       n/a      536486430  0
Others: 1983055       n/a       n/a      1649446    0

WRED drop counts:
  qid  thresh1    thresh2   FreeQ
   1 : 0          0         1024
   2 : 0          0         1024
   3 : 0          0         6
   4 : 0          0         1024
```

This is a description of the fields in the example:

- **Incoming** shows how many packets arrive from each direction
- **NO_change** shows how many packets were trusted (such as QoS level not changed)
- **Classified** shows how many packets have been assigned this internal DSCP after classification
- **Policed** shows how many packets were marked down by policing; DSCP shown before markdown.
- **Dropped** shows how many packets were dropped by policing

Be aware of these implementation−specific considerations:

- If eight DSCP values are configured when you issue the **mls qos monitor** command, the others counter seen when you issue the **show mls qos int statistics** command could display inadequate information.
- There is no specific command in order to verify the offered or outgoing traffic rate per−policer.
- Since the counters are retrieved from the hardware sequentially, it is possible that the counters do not add up correctly. For example, the amount of policed, classified, or dropped packets can be slightly different than the number of incoming packets.

Configure and Monitor Marking

These steps are necessary in order to configure marking:

1. Define the criteria for classifying the traffic
2. Define traffic classes to be classified with the criteria previously defined
3. Create a policy map that attaches marking actions and policing actions to the defined classes
4. Configure the corresponding interface(s) to trust mode
5. Apply the policy map to an interface

In this example, you want incoming IP traffic to host 192.168.192.168 marked with IP precedence 6 and policed down to 1 Mbps; excess traffic must be marked down to IP precedence 2:

```
!−−− Globally enables QoS.

mls qos
```

```
!--- Defines the ACLs to select traffic.

access-list 167 permit ip any host 192.168.192.168

!--- Defines the traffic class.

class-map match-all cl_2host
  match access-group 167

!--- Defines QoS policy, and creates and attaches
!--- the policers to the traffic classes.

policy-map po_test3
  class cl_2host

!--- Marks all the class traffic with the IP precedence 6.

    set ip precedence 6

!--- Polices down to 1 Mbps and marks down according to the QoS map.

    police 1000000 8000 exceed-action policed-dscp-transmit

!--- Modifies the policed DSCP QoS map, so the
!--- traffic is marked down from IP precedence 6 to 2.
!--- In terms of DSCP, this is from 48 to 16 (DSCP=IPprec x8).

mls qos map policed-dscp  48 to 16

!--- Applies the QoS policy to an interface.

interface GigabitEthernet0/3
 switchport
 switchport access vlan 2
 service-policy input po_test3
```

The same **show mls qos interface statistics** command is issued in order to monitor the marking. Sample output and implications are documented in the section of this document.

How to Classify All Interface Traffic with a Single Policer

On the Catalyst 3550, the **match interface** command is not supported, and only one match command is allowed per class−map. Moreover, the Catalyst 3550 does not allow the IP traffic to be matched by the MAC ACLs. So IP and non−IP traffic must be classified with two separate class−maps. This makes it tricky to classify all the traffic that comes into an interface and police all traffic with a single policer. The sample configuration here lets you accomplish this. In this configuration, IP and non−IP traffic are matched with two different class−maps. However, each uses a common policer for both the traffic.

```
access-list 100 permit ip any any

class-map ip
match access-group 100

!--- This class-map classifies all IP traffic.


mac access-list extended non-ip-acl
permit any any

class-map non-ip
match access-group name non-ip-acl
```

```
!--- Class-map classifies all non-IP traffic only.


mls qos aggregate-policer all-traffic 8000 8000 exceed-action drop

!--- This command configures a common policer that is applied for both IP and non-IP traff


policy-map police-all-traffic
class non-ip
police aggregate all-traffic
class ip
police aggregate all-traffic

interface gigabitEthernet 0/7
service-policy input police-all-traffic

!--- This command applies the policy map to the physical interface.
```

Related Information

- **Configuring QoS on Catalyst 3550**
- **Quality of Service Support Pages**
- **LAN Switching Support Page**
- **LAN Product Support Pages**
- **Technical Support & Documentation − Cisco Systems**

Contacts & Feedback | Help | Site Map
© 2014 − 2015 Cisco Systems, Inc. All rights reserved. Terms & Conditions | Privacy Statement | Cookie Policy | Trademarks of Cisco Systems, Inc.

Updated: Feb 15, 2008 Document ID: 24800