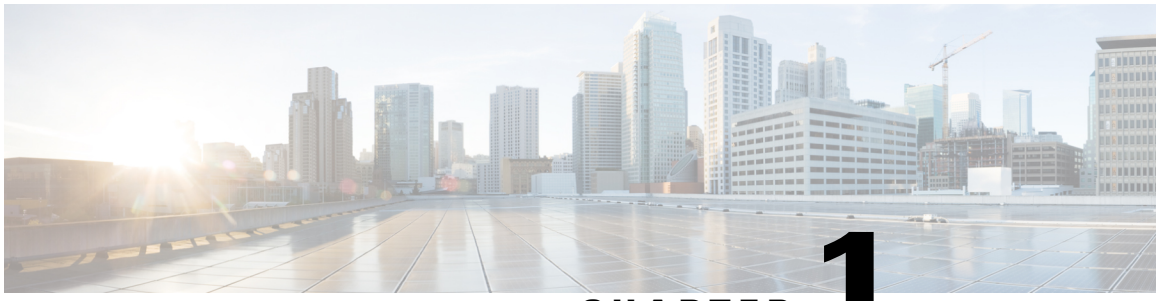
# QoS: Modular QoS Command-Line Interface Configuration Guide, Cisco IOS Release 15M&T

**First Published:** March 07, 2013

**Last Modified:** March 07, 2013

**Americas Headquarters**
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

# CONTENTS



CHAPTER 1

# Applying QoS Features Using the MQC

This module contains the concepts about applying QoS features using the Modular Quality of Service (QoS) Command-Line Interface (CLI) (MQC) and the tasks for configuring the MQC. The MQC allows you to define a traffic class, create a traffic policy (policy map), and attach the traffic policy to an interface. The traffic policy contains the QoS feature that will be applied to the traffic class.



## Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

## Restrictions for Applying QoS Features Using the MQC

The MQC supports a maximum of 256 classes in a single policy map.

# Information About Applying QoS Features Using the MQC

## The MQC Structure

The MQC structure allows you to define a traffic class, create a traffic policy, and attach the traffic policy to an interface.

The MQC structure consists of the following three high-level steps:

1. Define a traffic class by using the **class-map** command. A traffic class is used to classify traffic.
2. Create a traffic policy by using the **policy-map** command. (The terms *traffic policy* and *policy map* are often synonymous.) A traffic policy (policy map) contains a traffic class and one or more QoS features that will be applied to the traffic class. The QoS features in the traffic policy determine how to treat the classified traffic.
3. Attach the traffic policy (policy map) to the interface by using the **service-policy** command.

## Elements of a Traffic Class

A traffic class contains three major elements: a traffic class name, a series of **match** commands, and, if more than one **match** command is used in the traffic class, instructions on how to evaluate these **match** commands.

The **match** commands are used for classifying packets. Packets are checked to determine whether they meet the criteria specified in the **match**commands; if a packet meets the specified criteria, that packet is considered a member of the class. Packets that fail to meet the matching criteria are classified as members of the default traffic class.

### Available match Commands

The table below lists some of the available **match** commands that can be used with the MQC. The available **match** commands vary by Cisco IOS release and platform. For more information about the commands and command syntax, see the command reference for the Cisco IOS release and platform that you are using.

***Table 1: match Commands That Can Be Used with the MQC***

| Command | Purpose |
|---|---|
| **match access-group** | Configures the match criteria for a class map on the basis of the specified access control list (ACL). |
| **match any** | Configures the match criteria for a class map to be successful match criteria for all packets. |
| **match class-map** | Specifies the name of a traffic class to be used as a matching criterion (for nesting traffic classes [nested class maps] within one another). |

| Command | Purpose |
|---|---|
| **match cos** | Matches a packet based on a Layer 2 class of service (CoS) marking. |
| **match destination-address mac** | Uses the destination MAC address as a match criterion. |
| **match discard-class** | Matches packets of a certain discard class. |
| **match** [**ip**] **dscp** | Identifies a specific IP differentiated service code point (DSCP) value as a match criterion. Up to eight DSCP values can be included in one match statement. |
| **match field** | Configures the match criteria for a class map on the basis of the fields defined in the protocol header description files (PHDFs). |
| **match fr-dlci** | Specifies the Frame Relay data-link connection identifier (DLCI) number as a match criterion in a class map. |
| **match input-interface** | Configures a class map to use the specified input interface as a match criterion. |
| **match ip rtp** | Configures a class map to use the Real-Time Transport Protocol (RTP) port as the match criterion. |
| **match mpls experimental** | Configures a class map to use the specified value of the Multiprotocol Label Switching (MPLS) experimental (EXP) field as a match criterion. |
| **match mpls experimental topmost** | Matches the MPLS EXP value in the topmost label. |
| **match not** | Specifies the single match criterion value to use as an unsuccessful match criterion.<br>**Note** The **match not** command, rather than identifying the specific match parameter to use as a match criterion, is used to specify a match criterion that prevents a packet from being classified as a member of the class. For instance, if the **match not qos-group 6**command is issued while you configure the traffic class, QoS group 6 becomes the only QoS group value that is not considered a successful match criterion. All other QoS group values would be successful match criteria. |
| **match packet length** | Specifies the Layer 3 packet length in the IP header as a match criterion in a class map. |

| Command | Purpose |
|---|---|
| **match port-type** | Matches traffic on the basis of the port type for a class map. |
| **match** [**ip**] **precedence** | Identifies IP precedence values as match criteria. |
| **match protocol** | Configures the match criteria for a class map on the basis of the specified protocol.<br>**Note** There is a separate **match protocol**(NBAR) command used to configure Network-Based Application Recognition (NBAR) to match traffic by a protocol type known to NBAR. |
| **match protocol citrix** | Configures NBAR to match Citrix traffic. |
| **match protocol fasttrack** | Configures NBAR to match FastTrack peer-to-peer traffic. |
| **match protocol gnutella** | Configures NBAR to match Gnutella peer-to-peer traffic. |
| **match protocol http** | Configures NBAR to match Hypertext Transfer Protocol (HTTP) traffic by URL, host, Multipurpose Internet Mail Extension (MIME) type, or fields in HTTP packet headers. |
| **match protocol rtp** | Configures NBAR to match Real-Time Transport Protocol (RTP) traffic. |
| **match qos-group** | Identifies a specific QoS group value as a match criterion. |
| **match source-address mac** | Uses the source MAC address as a match criterion. |
| **match start** | Configures the match criteria for a class map on the basis of the datagram header (Layer 2) or the network header (Layer 3). |
| **match tag** | Specifies tag type as a match criterion. |

If the traffic class contains more than one **match** command, you need to specify how to evaluate the **match** commands. You specify this by using either the **match-any** or **match-all**keywords of the **class-map** command. Note the following points about the **match-any** and **match-all** keywords:

- If you specify the **match-any**keyword, the traffic being evaluated by the traffic class must match *one* of the specified criteria.
- If you specify the **match-all** keyword, the traffic being evaluated by the traffic class must match *all* of the specified criteria.

- If you do not specify either keyword, the traffic being evaluated by the traffic class must match *all* of the specified criteria (that is, the behavior of the **match-all** keyword is used).

# Elements of a Traffic Policy

A traffic policy contains three elements: a traffic policy name, a traffic class (specified with the **class** command), and the command used to enable the QoS feature.

The traffic policy (policy map) applies the enabled QoS feature to the traffic class once you attach the policy map to the interface (by using the **service-policy** command).

**Note** A packet can match only *one* traffic class within a traffic policy. If a packet matches more than one traffic class in the traffic policy, the *first* traffic class defined in the policy will be used.

The commands used to enable QoS features vary by Cisco IOS release and platform. The table below lists some of the available commands and the QoS features that they enable. For complete command syntax, see the command reference for the Cisco IOS release and platform that you are using.

***Table 2: Commands Used to Enable QoS Features***

| Command | Purpose |
|---|---|
| **bandwidth** | Enables Class-Based Weighted Fair Queuing (CBWFQ). |
| **fair-queue** | Specifies the number of queues to be reserved for a traffic class. |
| **drop** | Discards the packets in the specified traffic class. |
| **identity policy** | Creates an identity policy. |
| **police** | Configures traffic policing. |
| **police (control-plane)** | Configures traffic policing for traffic that is destined for the control plane. |
| **police (EtherSwitch)** | Defines a policer for classified traffic. |
| **police (percent)** | Configures traffic policing on the basis of a percentage of bandwidth available on an interface. |
| **police (two rates)** | Configures traffic policing using two rates, the committed information rate (CIR) and the peak information rate (PIR). |

| Command | Purpose |
|---|---|
| **police rate pdp** | Configures Packet Data Protocol (PDP) traffic policing using the police rate.<br>**Note** This command is intended for use on the Gateway General Packet Radio Service (GPRS) Support Node (GGSN). |
| **priority** | Gives priority to a class of traffic belonging to a policy map. |
| **queue-limit** | Specifies or modifies the maximum number of packets the queue can hold for a class configured in a policy map. |
| **random-detect** | Enables Weighted Random Early Detection (WRED) or distributed WRED (DWRED). |
| **random-detect discard-class** | Configures the WRED parameters for a discard-class value for a class in a policy map. |
| **random-detect discard-class-based** | Configures WRED on the basis of the discard class value of a packet. |
| **random-detect ecn** | Enables explicit congestion notification (ECN). |
| **random-detect exponential-weighting-constant** | Configures the exponential weight factor for the average queue size calculation for the queue reserved for a class. |
| **random-detect precedence** | Configure the WRED parameters for a particular IP Precedence for a class policy in a policy map. |
| **service-policy** | Specifies the name of a traffic policy used as a matching criterion (for nesting traffic policies [hierarchical traffic policies] within one another). |
| **set atm-clp** | Sets the cell loss priority (CLP) bit when a policy map is configured. |
| **set cos** | Sets the Layer 2 class of service (CoS) value of an outgoing packet. |
| **set discard-class** | Marks a packet with a discard-class value. |
| **set** [**ip**] **dscp** | Marks a packet by setting the differentiated services code point (DSCP) value in the type of service (ToS) byte. |

| Command | Purpose |
|---|---|
| **set fr-de** | Changes the discard eligible (DE) bit setting in the address field of a Frame Relay frame to 1 for all traffic leaving an interface. |
| **set mpls experimental** | Designates the value to which the MPLS bits are set if the packets match the specified policy map. |
| **set precedence** | Sets the precedence value in the packet header. |
| **set qos-group** | Sets a QoS group identifier (ID) that can be used later to classify packets. |
| **shape** | Shapes traffic to the indicated bit rate according to the algorithm specified. |
| **shape adaptive** | Configures a Frame Relay interface or a point-to-point subinterface to estimate the available bandwidth by backward explicit congestion notification (BECN) integration while traffic shaping is enabled. |
| **shape fecn-adapt** | Configures a Frame Relay interface to reflect received forward explicit congestion notification (FECN) bits as backward explicit congestion notification (BECN) bits in Q.922 test response messages. |

# Nested Traffic Classes

The MQC does not necessarily require that you associate only one traffic class to one traffic policy. When packets meet more than one match criterion, multiple traffic classes can be associated with a single traffic policy.

Similarly, the MQC allows multiple traffic classes (nested traffic classes, which are also called nested class maps or MQC Hierarchical class maps) to be configured as a single traffic class. This nesting can be achieved with the use of the **match class-map** command. The only method of combining match-any and match-all characteristics within a single traffic class is with the **match class-map** command.

# match-all and match-any Keywords of the class-map Command

One of the commands used when you create a traffic class is the **class-map**command. The command syntax for the **class-map** command includes two keywords: **match-all** and **match-any**. The **match-all** and **match-any** keywords need to be specified only if more than one match criterion is configured in the traffic class. Note the following points about these keywords:

- The **match-all** keyword is used when *all* of the match criteria in the traffic class must be met in order for a packet to be placed in the specified traffic class.

- The **match-any** keyword is used when only *one* of the match criterion in the traffic class must be met in order for a packet to be placed in the specified traffic class.
- If neither the **match-all** keyword nor **match-any** keyword is specified, the traffic class will behave in a manner consistent with the **match-all** keyword.

## input and output Keywords of the service-policy Command

The QoS feature configured in the traffic policy can be applied to packets entering the interface or to packets leaving the interface. Therefore, when you use the **service-policy** command, you need to specify the direction by using the **input** or **output** keyword.

For instance, the **service-policy output class1**command would apply the feature in the traffic policy to the interface. All packets leaving the interface are evaluated according to the criteria specified in the traffic policy named class1.

## Benefits of Applying QoS Features Using the MQC

The MQC structure allows you to create the traffic policy (policy map) once and then apply it to as many traffic classes as needed. You can also attach the traffic policies to as many interfaces as needed.

# How to Apply QoS Features Using the MQC

To create a traffic class, use the **class-map** command to specify the traffic class name. Then use one or more **match** commands to specify the appropriate match criteria. Packets matching the criteria that you specify are placed in the traffic class.

The traffic policy (policy map) applies the enabled QoS feature to the traffic class once you attach the policy map to the interface (by using the **service-policy** command).

Depending on the platform and Cisco IOS XE release that you are using, a traffic policy can be attached to an ATM permanent virtual circuit (PVC) subinterface, to a Frame Relay data-link connection identifier (DLCI), or to another type of interface.

## Creating a Traffic Class Using the MQC

**Note** The **match cos**command is shown in Step Creating a Traffic Class Using the MQC The **match cos** command is simply an example of one of the **match** commands that you can use. For information about the other available **match** commands, see Creating a Traffic Class Using the MQC, on page 8.

## SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** [**match-all** | **match-any**] *class-map-name*
4. **match cos** *cos-number*
5. Enter additional match commands, if applicable; otherwise, continue with Creating a Traffic Class Using the MQC .
6. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br><br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br><br>**Example:**<br>`Router# configure terminal` | Enters global configuration mode. |
| **Step 3** | **class-map** [**match-all** \| **match-any**] *class-map-name*<br><br>**Example:**<br>`Router(config)# class-map match-any class1` | Creates a class to be used with a class map and enters class-map configuration mode. The class map is used for matching packets to the specified class.<br>• Enter the class name.<br><br>**Note** The **match-all** keyword specifies that all match criteria must be met. The **match-any** keyword specifies that one of the match criterion must be met. Use these keywords only if you will be specifying more than one **match** command. |
| **Step 4** | **match cos** *cos-number*<br><br>**Example:**<br>`Router(config-cmap)# match cos 2` | Matches a packet on the basis of a Layer 2 class of service (CoS) number.<br>• Enter the CoS number.<br><br>**Note** The **match cos** command is simply an example of one of the **match** commands you can use. For information about the other **match** commands that are available, see Creating a Traffic Class Using the MQC, on page 8. |
| **Step 5** | Enter additional match commands, if applicable; otherwise, continue with Creating a Traffic Class Using the MQC . | -- |

| | Command or Action | Purpose |
|---|---|---|
| **Step 6** | **end**<br><br>**Example:**<br>`Router(config-cmap)# end` | (Optional) Exits class-map configuration mode and returns to privileged EXEC mode. |

# Creating a Traffic Policy Using the MQC

**Note** The **bandwidth** command is shown in Step Creating a Traffic Policy Using the MQC The **bandwidth** command is simply an example of one of the commands that you can use in a policy map. For information about other available commands, see Creating a Traffic Policy Using the MQC, on page 10.

## SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name*| **class-default**}
5. **bandwidth** *bandwidth-kbps* | **percent** *percent*
6. Enter the commands for any additional QoS feature that you want to enable, if applicable; otherwise, continue with Creating a Traffic Policy Using the MQC.
7. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br><br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br><br>**Example:**<br>`Router# configure terminal` | Enters global configuration mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 3** | **policy-map** *policy-map-name*<br><br>**Example:**<br>`Router(config)#`<br>`policy-map policy1` | Creates or specifies the name of the traffic policy and enters policy-map configuration mode.<br>• Enter the policy map name. |
| **Step 4** | **class** {*class-name*\| **class-default**}<br><br>**Example:**<br>`Router(config-pmap)#`<br>`class class1` | Specifies the name of a traffic class and enters policy-map class configuration mode.<br>**Note** This step associates the traffic class with the traffic policy. |
| **Step 5** | **bandwidth** *bandwidth-kbps* \| **percent** *percent*<br><br>**Example:**<br>`Router(config-pmap-c)# bandwidth 3000` | (Optional) Specifies a minimum bandwidth guarantee to a traffic class in periods of congestion. A minimum bandwidth guarantee can be specified in kbps or by a percentage of the overall available bandwidth.<br>**Note** The **bandwidth** command is simply an example of one of the commands that you can use in a policy map to enable a QoS feature. For information about the other commands available, see Creating a Traffic Policy Using the MQC, on page 10. |
| **Step 6** | Enter the commands for any additional QoS feature that you want to enable, if applicable; otherwise, continue with Creating a Traffic Policy Using the MQC. | -- |
| **Step 7** | **end**<br><br>**Example:**<br>`Router(config-pmap-c)# end` | (Optional) Exits policy-map class configuration mode and returns to privileged EXEC mode. |

# Attaching a Traffic Policy to an Interface

The traffic policy (policy map) applies the enabled QoS feature to the traffic class once you attach the policy map to the interface (by using the **service-policy** command). For information about the input and output keywords of the service-policy command, see the input and output Keywords of the service-policy Command, on page 8.

Depending on the platform and Cisco IOS release that you are using, a traffic policy can be attached to an ATM permanent virtual circuit (PVC) subinterface, a Frame Relay data-link connection identifier (DLCI), or another type of interface.

To attach a traffic policy to an interface, complete the following steps.

**Note** Multiple traffic policies on tunnel interfaces and physical interfaces are not supported if the interfaces are associated with each other. For instance, if a traffic policy is attached to a tunnel interface while another traffic policy is attached to a physical interface--with which the tunnel interface is associated--only the traffic policy on the tunnel interface works properly.

The amount of bandwidth allocated to the priority traffic cannot exceed the amount of bandwidth available on the interface. If the traffic policy is configured such that the amount of bandwidth allocated to the priority traffic exceeds the amount of bandwidth available on the interface, the traffic policy will be suspended. Previously, the policy map would have been rejected. Now that it is only suspended, you have the option of modifying the traffic policy accordingly and then reattaching the traffic policy to the interface.

## SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-type interface-number*
4. **service-policy** {**input** | **output**} *policy-map-name*
5. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Router# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-type interface-number*<br>**Example:**<br>`Router(config)# interface serial0` | Configures an interface type and enters interface configuration mode.<br>• Enter the interface type and interface number. |
| **Step 4** | **service-policy** {**input** \| **output**} *policy-map-name*<br>**Example:**<br>`Router(config-if)#`<br>`service-policy input policy1` | Attaches a policy map to an interface.<br>• Enter either the **input** or **output** keyword and the policy map name. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **end**<br><br>**Example:**<br>`Router (config-if)# end` | (Optional) Exits interface configuration mode and returns to privileged EXEC mode. |

# Verifying the Traffic Class and Traffic Policy Information

## SUMMARY STEPS

1. **enable**
2. **show class-map**
3. **show policy-map** *policy-map-name* **class** *class-name*
4. **show policy-map**
5. **show policy-map interface** *interface-type interface-number*
6. **exit**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br><br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **show class-map**<br><br>**Example:**<br>`Router# show class-map` | (Optional) Displays all class maps and their matching criteria. |
| **Step 3** | **show policy-map** *policy-map-name* **class** *class-name*<br><br>**Example:**<br>`Router#`<br>`show policy-map policy1 class class1` | (Optional) Displays the configuration for the specified class of the specified policy map.<br>• Enter the policy map name and the class name. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | **show policy-map**<br><br>**Example:**<br>`Router# show policy-map` | (Optional) Displays the configuration of all classes for all existing policy maps. |
| **Step 5** | **show policy-map interface** *interface-type interface-number*<br><br>**Example:**<br>`Router# show policy-map interface serial0` | (Optional) Displays the statistics and the configurations of the input and output policies that are attached to an interface.<br>• Enter the interface type and number. |
| **Step 6** | **exit**<br><br>**Example:**<br>`Router# exit` | (Optional) Exits privileged EXEC mode. |

# Configuration Examples for Applying QoS Features Using the MQC

## Example: Creating a Traffic Class

In the following example, two traffic classes are created and their match criteria are defined. For the first traffic class called class1, access control list (ACL) 101 is used as the match criterion. For the second traffic class called class2, 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)# 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)# end
```

## Example Creating a Traffic Policy

In the following example, a traffic policy called policy1 is defined. The traffic policy contains the QoS features to be applied to two classes--class1 and class2. The match criteria for these classes were previously defined (as described in the Example Creating a Traffic Class).

For class1, the policy includes a bandwidth allocation request and a maximum packet count limit for the queue reserved for the class. For class2, the policy specifies only a bandwidth allocation request.

```
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)# end
```

# Example Attaching a Traffic Policy to an Interface

The following example shows how to attach an existing traffic policy to an interface. After you define a traffic policy with the **policy-map** command, you can attach it to one or more interfaces by using the **service-policy** command in interface configuration mode. Although you can assign the same traffic policy to multiple interfaces, each interface can have only one traffic policy attached in the input direction and only one traffic policy attached in the output direction.

```
Router(config)# interface ethernet1/1
Router(config-if)# service-policy output policy1
Router(config-if)# exit
Router(config)# interface fastethernet1/0/0
Router(config-if)# service-policy output policy1
Router(config-if)# exit
```

# Example: match not Command

The **match not**command is used to specify a specific QoS policy value that is not used as a match criterion. If the **match not** command is issued, all other values of that QoS policy become successful match criteria. For instance, if the **match not qos-group 4** command is issued in QoS class-map configuration mode, the specified class will accept all QoS group values except 4 as successful match criteria.

In the following traffic class, all protocols except IP are considered successful match criteria:

```
Router(config)# class-map noip
Router(config-cmap)# match not protocol ip
Router(config-cmap)# end
```

# Example: Default Traffic Class Configuration

Unclassified traffic (traffic that does not meet the match criteria specified in the traffic classes) is treated as belonging to the default traffic class.

If you do not configure a default class, packets are still treated as members of the default class. However, by default, the default class has no QoS features enabled. Therefore, packets belonging to a default class have no QoS functionality. These packets are placed into a first-in, first-out (FIFO) queue managed by tail drop. Tail drop is a means of avoiding congestion that treats all traffic equally and does not differentiate between classes of service. Queues fill during periods of congestion. When the output queue is full and tail drop is in effect, packets are dropped until the congestion is eliminated and the queue is no longer full.

# Example: class-map match-any and class-map match-all Commands

This example illustrates the difference between the **class-map match-any** command and the **class-map match-all** command. The **match-any** and **match-all** keywords determine how packets are evaluated when multiple match criteria exist. Packets must either meet all of the match criteria (**match-all**) or meet one of the match criteria (**match-any**) to be considered a member of the traffic class.

The following example shows a traffic class configured with the **class-map match-all** command:

If a packet arrives on a router with the traffic class called cisco1 configured on the interface, the packet is evaluated to determine if it matches the IP protocol, QoS group 4, *and* access group 101. If all three of these match criteria are met, the packet is classified as a member of the traffic class cisco1.

The following example shows a traffic class that is configured with the **class-map match-any** command:

In the traffic class called cisco2, the match criteria are evaluated consecutively until a successful match criterion is located. The packet is first evaluated to determine whether the IP protocol can be used as a match criterion. If the IP protocol can be used as a match criterion, the packet is matched to traffic class cisco2. If the IP protocol is not a successful match criterion, then QoS group 4 is evaluated as a match criterion. Each criterion is evaluated to see if the packet matches that criterion. Once a successful match occurs, the packet is classified as a member of traffic class cisco2. If the packet matches none of the specified criteria, the packet is classified as a member of the default traffic class (class default-class).

Note that the **class-map match-all** command requires that *all* of the match criteria be met in order for the packet to be considered a member of the specified traffic class (a logical AND operator). In the first example, protocol IP AND QoS group 4 AND access group 101 must be successful match criteria. However, only one match criterion must be met in order for the packet in the **class-map match-any** command to be classified as a member of the traffic class (a logical OR operator). In the second example, protocol IP OR QoS group 4 OR access group 101 must be successful match criterion.

# Example: Traffic Class as a Match Criterion (Nested Traffic Classes)

There are two reasons to use the **match class-map** command. One reason is maintenance; if a large traffic class currently exists, using the traffic class match criterion is easier than retyping the same traffic class configuration. The more common reason for the **match class-map** command is to allow users to use match-any and match-all statements in the same traffic class. If you want to combine match-all and match-any characteristics in a traffic policy, create a traffic class using one match criterion evaluation instruction (either match-any or match-all) and then use this traffic class as a match criterion in a traffic class that uses a different match criterion type.

Here is a possible scenario: Suppose A, B, C, and D were all separate match criterion, and you wanted traffic matching A, B, or C and D (A or B or [C and D]) to be classified as belonging to the traffic class. Without the nested traffic class, traffic would either have to match all four of the match criterion (A and B and C and D) or match any of the match criterion (A or B or C or D) to be considered part of the traffic class. You would not be able to combine “and” (match-all) and “or” (match-any) statements within the traffic class, and you would therefore be unable to configure the desired configuration.

The solution: Create one traffic class using match-all for C and D (which we will call criterion E), and then create a new match-any traffic class using A, B, and E. The new traffic class would have the correct evaluation sequence (A or B or E, which would also be A or B or [C and D]). The desired traffic class configuration has been achieved.

The only method of mixing match-all and match-any statements in a traffic class is through the use of the traffic class match criterion.

## Example: Nested Traffic Class for Maintenance

In the following example, the traffic class called class1 has the same characteristics as the traffic class called class2, with the exception that traffic class class1 has added a destination address as a match criterion. Rather than configuring traffic class class1 line by line, you can enter the **match class-map class2** command. This command allows all of the characteristics in the traffic class called class2 to be included in the traffic class called class1, and you can add the new destination address match criterion without reconfiguring the entire traffic class.

```
Router(config)# class-map match-any class2
Router(config-cmap)# match protocol ip
Router(config-cmap)# match qos-group 3
Router(config-cmap)# match access-group 2
Router(config-cmap)# exit
Router(config)# class-map match-all class1
Router(config-cmap)# match class-map class2
Router(config-cmap)# match destination-address mac 00.00.00.00.00.00
Router(config-cmap)# exit
```

## Example Nested Traffic Class to Combine match-any and match-all Characteristics in One Traffic Class

The only method of including both match-any and match-all characteristics in a single traffic class is to use the **match class-map** command. To combine match-any and match-all characteristics into a single class, a traffic class created with the match-any instruction must use a class configured with the match-all instruction as a match criterion (through the **match class-map** command) or vice versa.

The following example shows how to combine the characteristics of two traffic classes, one with match-any and one with match-all characteristics, into one traffic class with the **match class-map** command. The result requires a packet to match one of the following three match criteria to be considered a member of traffic class class4: IP protocol *and* QoS group 4, destination MAC address 00.00.00.00.00.00, or access group 2.

In this example, only the traffic class called class4 is used with the traffic policy called policy1.

```
Router(config)# class-map match-all class3
Router(config-cmap)# match protocol ip
Router(config-cmap)# match qos-group 4
Router(config-cmap)# exit
Router(config)# class-map match-any class4
Router(config-cmap)# match class-map class3
Router(config-cmap)# match destination-address mac 00.00.00.00.00.00
Router(config-cmap)# match access-group 2
Router(config-cmap)# exit
Router(config)# policy-map policy1
Router(config-pmap)# class class4
Router(config-pmap-c)# police 8100 1500 2504 conform-action transmit exceed-action
set-qos-transmit 4
Router(config-pmap-c)# end
```

# Example Traffic Policy as a QoS Policy (Hierarchical Traffic Policies)

A traffic policy can be included in a QoS policy when the **service-policy** command is used in policy-map class configuration mode. A traffic policy that contains a traffic policy is called a hierarchical traffic policy.

A hierarchical traffic policy contains a child policy and a parent policy. The child policy is the previously defined traffic policy that is being associated with the new traffic policy through the use of the **service-policy**

command. The new traffic policy using the preexisting traffic policy is the parent policy. In the example in this section, the traffic policy called child is the child policy and traffic policy called parent is the parent policy.

Hierarchical traffic policies can be attached to subinterfaces and ATM PVCs. When hierarchical traffic policies are used, a single traffic policy (with a child and a parent policy) can be used to shape and prioritize PVC traffic. In the following example, the child policy is responsible for prioritizing traffic and the parent policy is responsible for shaping traffic. In this configuration, the parent policy allows packets to be sent from the interface, and the child policy determines the order in which the packets are sent.

```
Router(config)# policy-map child
Router(config-pmap)# class voice
Router(config-pmap-c)# priority 50
Router(config)# policy-map parent
Router(config-pmap)# class class-default
Router(config-pmap-c)# shape average 10000000
Router(config-pmap-c)# service-policy child
```

The value used with the **shape** command is provisioned from the committed information rate (CIR) value from the service provider.

# Additional References

### Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS commands | Cisco IOS Master Commands List, All Releases |
| Selective Packet Discard | "IPv6 Selective Packet Discard" module |

### Standards

| Standard | Title |
|---|---|
| No new or modified standards are supported, and support for existing standards has not been modified. | -- |

### MIBs

| MIB | MIBs Link |
|---|---|
| No new or modified MIBs are supported, and support for existing MIBs has not been modified. | To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL:<br>http://www.cisco.com/go/mibs |

### RFCs

| RFC | Title |
|---|---|
| No new or modified RFCs are supported, and support for existing RFCs has not been modified. | -- |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password. | http://www.cisco.com/cisco/web/support/index.html |

# Feature Information Applying QoS Features Using the MQC

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

***Table 3: Feature Information for Applying QoS Features Using the MQC***

| Feature Name | Releases | Feature Information |
|---|---|---|
| Modular QoS CLI (MQC) Unconditional Packet Discard | 12.2(13)T | The Modular QoS CLI (MQC) Unconditional Packet Discard feature allows you to classify traffic matching certain criteria and then configure the system to unconditionally discard any packets matching that criteria. |

| Feature Name | Releases | Feature Information |
|---|---|---|
| Class-Based Frame Relay Discard Eligible (DE)-Bit Matching and Marking | 12.2(2)T | The Class-Based Frame Relay Discard Eligible (DE)-Bit Matching and Marking feature enhances the MQC to support Frame Relay DE bit matching and marking. Packets with FR DE bit set can be matched to a class and the appropriate QoS feature or treatment be applied. |
| Modular QoS CLI (MQC) | Cisco IOS XE Release 2.1<br>Cisco IOS XE 3.1.0 SG | This feature was introduced on Cisco ASR 1000 Series Routers.<br>In Cisco IOS XE 3.1.0 SG, this feature was integrated. |

# Legacy Commands Being Hidden

The table below lists the commands that have been hidden or removed. The table also lists their replacement commands (or sequence of commands).

***Table 4: Map of Hidden, Removed or Unsupported Commands to Their Replacement Commands***

| Hidden, Removed or Unsupported Commands | Replacement MQC Command Sequence |
|---|---|
| Configuring Weighted Random Early Detection or Distributed Weighted Random Early Detection Parameter Groups | |
| **Commands**<br>• random-detect-group<br>• random-detect (per VC)<br>**Note** This command is not supported in Cisco IOS Release 15.0(1)S.<br>**Command Usage**<br>`Router(config)# random-detect-group group-name [dscp-based\|prec-based]`<br>`Router(config)# interface atm type number`<br>`Router(config-if)# pvc [name] vpi/vci`<br>`Router(config-if-atm-vc)# random-detect [attach group-name ]` | **Command Usage**<br>None (this functionality no longer exists). |
| Configuring Weighted Random Early Detection | |

| Hidden, Removed or Unsupported Commands | Replacement MQC Command Sequence |
|---|---|
| **Commands**<br>• random-detect<br>• random-detect dscp<br>• random-detect (dscp-based keyword)<br>• random-detect flow<br>• random-detect exponential-weighting-constant<br>• random-detect (prec-based keyword)<br>• random-detect precedence<br>**Command Usage**<br>`Router(config)# interface type number`<br>`Router(config-if)# random-detect [number]`<br>`Router(config-if)# random-detect exponential-weighting-constant exponent`<br>`Router(config-if)# random-detect flow`<br>`Router(config-if)# random-detect precedence {precedence\|rsvp} min-threshold max-threshold max-probability-denominator`<br>`Router(config-if)# random-detect prec-based`<br>`Router(config-if)# random-detect dscp-based`<br>`Router(config-if)# random-detect dscp dscp-value min-threshold max-threshold[max-probability-denominator]` | **Command Usage**<br>`Router(config)# policy-map`<br>`policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# random-detect dscp dscp-value min-threshold max-threshold[ mark-probability-denominator]`<br>`Router(config-pmap-c)# random-detect clp clp-value min-threshold max-threshold[mark-probability-denominator]`<br>`Router(config-pmap-c)# random-detect cos cos-value min-threshold max-threshold[mark-probability-denominator]`<br>`Router(config-pmap-c)# random-detect discard-class discard-class-value min-threshold max-threshold[ mark-probability-denominator]`<br>`Router(config-pmap-c)# random-detectprecedence ip-precedence min-threshold max-threshold[mark-probability-denominator]`<br>`Router(config-pmap-c)# random-detect precedence-based`<br>`Router(config-pmap-c)# random-detect ecn`<br>`Router(config-pmap-c)# random-detect exponential-weighting-constant exponent`<br>`Router(config-pmap-c)# random-detect cos-based`<br>`Router(config-pmap-c)# random-detect dscp-based` |
| **Commands**<br>• random-detect flow<br>• random-detect flow average-depth-factor<br>• random-detect flow count<br>**Command Usage**<br>`Router(config)# interface type number`<br>`Router(config-if)# random-detect [number]`<br>`Router(config-if)# random-detect flow`<br>`Router(config-if)# random-detect flow count number`<br>`Router(config-if)# random-detect flow average-depth-factor scaling-factor` | **Command Usage**<br>None (this functionality no longer exists). |
| Configuring Bandwidth Allocation | |

| Hidden, Removed or Unsupported Commands | Replacement MQC Command Sequence |
|---|---|
| **Commands**<br>• max-reserved-bandwidth<br>**Command Usage**<br>`Router(config)# interface type number`<br>`Router(config-if)# max-reserved-bandwidth percentage` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# bandwidth{`*bandwidth-in-kbps* `\|remaining percent` *percentage* `\| percent` *percentage*`}` |
| Configuring Custom Queueing | |
| **Commands**<br>• custom-queue-list<br>**Note** This command is not supported in Cisco IOS Release 15.0(1)S.<br>**Command Usage**<br>`Router(config)# interface type number`<br>`Router(config-if)#`<br>`custom-queue-list[list-number]` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# bandwidth{`<br>*bandwidth-in-kbps* `\|remaining percent` *percentage*<br>`\|percent` *percentage*`}` |
| Configuring Priority Queueing | |
| **Commands**<br>• ip rtp priority<br>• ip rtp reserve<br>**Command Usage**<br>`Router(config)# interface type number`<br>`Router(config-if)# ip rtp priority starting-port-number port-range bandwidth`<br>`Router(config)# interface type number`<br>`Router(config-if)# ip rtp reserve lowest-udp-port range-of-ports [maximum-bandwidth] 1000` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-name`<br>`Router(config-pmap-c)# priority` |
| Configuring Weighted Fair Queueing | |

| Hidden, Removed or Unsupported Commands | Replacement MQC Command Sequence |
|---|---|
| **Commands**<br>• fair-queue (WFQ)<br>**Command Usage (Cisco IOS Release 15.0(1)S)**<br>`Router(config)# interface type number`<br>`Router(config-if)# fair-queue`<br>**Command Usage (Cisco IOS Release 15.1(3)T)**<br>`Router(config)# interfacetype number`<br>`Router(config-if)# fair-queue [congestive-` *discard-threshold* `[dynamic-queue-count` `[`*reserved-queue-count*`]]]` | **Command Usage (Cisco IOS Release 15.0(1)S)**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# fair-queue`<br>**Command Usage (Cisco IOS Release 15.1(3)T)**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# fair-queue[dynamic-queues ]` |
| Assigning a Priority Group to an Interface | |
| **Commands**<br>• priority-group<br>**Note** This command is not supported in Cisco IOS Release 15.0(1)S.<br>**Command Usage**<br>`Router(config)# interface type number`<br>`Router(config-if)# priority-group list-number` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# priority`<br>`Router(config-pmap-c)# priority bandwidth-in-kbps [burst-in-bytes]`<br>`Router(config-pmap-c)# priority percent percent [burst-in-bytes]`<br>`Router(config-pmap-c)# priority level level`<br>`Router(config-pmap-c)# priority level level [bandwidth-in-kbps [burst-in-bytes]]`<br>`Router(config-pmap-c)# priority level level[percent percent [burst-in-bytes]]` |
| Configuring the Threshold for Discarding DE Packets from a Switched PVC Traffic Shaping Queue | |
| **Commands**<br>• frame-relay congestion threshold de<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config-map-class)# frame-relay congestion threshold de` percentage | **Command Usage**<br>`Router(config)# policy-map policy-map-name1`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# random-detect discard-class-based`<br>`Router(config-pmap-c)# random-detect discard-class discard-class min-threshold max-threshold`<br>`Router(config-pmap-c)# exit`<br>`Router(config-pmap)# exit`<br>`Router(config)# policy-map shape`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# shape average rate`<br>`Router(config-pmap-c)# service-policy policy-map-name1`<br>`Router(config-pmap-c)# exit`<br>`Router(config-pmap)# exit`<br>`Router(config)# policy-map policy-map-name2`<br>`Router(config-pmap)# class class-name`<br>`Router(config-pmap-c)# set discard-class`discard-class |
| Configuring Frame Relay Custom Queueing for Virtual Circuits | |

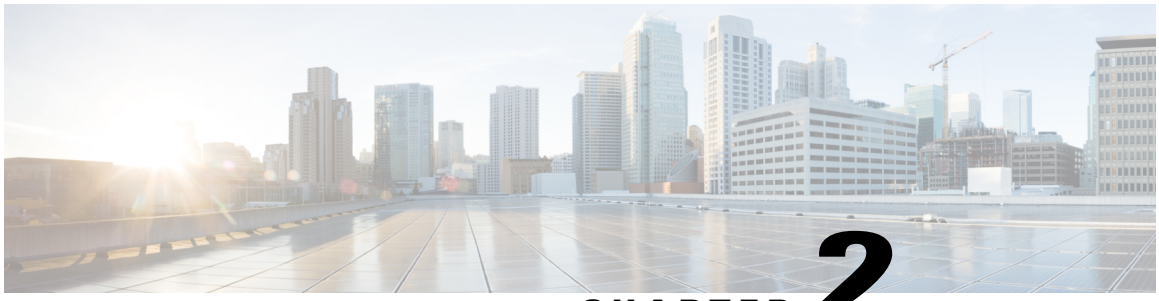
| Hidden, Removed or Unsupported Commands | Replacement MQC Command Sequence |
|---|---|
| **Commands**<br>• frame-relay custom-queue-list<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config-map-class)# frame-relay custom-queue-list list-number` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# bandwidth{`*bandwidth-in-kbps* `\| remaining percent` *percentage* `\| percent`*percentage*`}` |
| Configuring Frame Relay ECN Bits Threshold | |
| **Commands**<br>• frame-relay congestion threshold ecn<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config-map-class)# frame-relay congestion threshold ecn percentage` | **Command Usage**<br>None (this functionality no longer exists).<br>The closest equivalent is MQC traffic shaping (not based on ECN).<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# shape average rate` |
| Configuring Frame Relay Weighted Fair Queueing | |
| **Commands**<br>• frame-relay fair-queue<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config-map-class)# frame-relay fair-queue [discard-threshold [dynamic-queue-count[reserved-queue-count [buffer-limit]]]]` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# fair-queue`<br>`Router(config-pmap-c)# fair-queue queue-limit packets`<br>**Note** The **queue-limit** *packets* keyword and argument pair is not supported in Cisco IOS Release 15.1(3)T. |
| Configuring Frame Relay Priority Queueing on a PVC | |
| **Commands**<br>• frame-relay ip rtp priority<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config-map-class)# frame-relay ip rtp priority starting-port-number port-range bandwidth` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-name`<br>`Router(config-pmap-c)# priority bandwidth-in-kbps [burst-in-bytes]` |
| Assigning a Priority Queue to Virtual Circuits Associated with a Map Class | |

| Hidden, Removed or Unsupported Commands | Replacement MQC Command Sequence |
|---|---|
| **Commands**<br>• frame-relay priority-group<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config-map-class)# frame-relay priority-group group-number` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# priority`<br>`Router(config-pmap-c)# priority bandwidth-in-kbps [burst-in-bytes]`<br>`Router(config-pmap-c)# priority percent percentage [burst-in-bytes]`<br>`Router(config-pmap-c)# priority level level [percent percentage [burst-in-bytes]]`<br>**Note** The **priority level**command is not supported in Cisco IOS Release 15.1(3)T. |
| Configuring the Frame Relay Rate Adjustment to BECN | |
| **Commands**<br>• frame-relay adaptive-shaping (becn keyword)<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config-map-class)# frame-relay adaptive-shaping becn` | **Command Usage**<br>None (this functionality no longer exists). The closest equivalent is MQC traffic shaping (not based on BECN).<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# shape adaptive rate` |
| Configuring the Frame Relay Rate Adjustment to ForeSight Messages | |
| **Commands**<br>• frame-relay adaptive-shaping (foresight keyword)<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config)# frame-relay adaptive-shaping foresight` | **Command Usage**<br>None (this functionality no longer exists). |
| Enabling Frame Relay Traffic-Shaping FECNs as BECNs | |
| **Commands**<br>• frame-relay fecn-adapt<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config-map-class)#frame-relay fecn-adapt` | **Command Usage**<br>None (this functionality no longer exists). The closest equivalent is MQC traffic shaping (not based on FECN/BECN).<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# shape average rate` |
| Configuring the Frame Relay Enhanced Local Management Interface | |

| Hidden, Removed or Unsupported Commands | Replacement MQC Command Sequence |
|---|---|
| **Commands**<br>• frame-relay qos-autosense<br>**Note** This command has not been hidden in Cisco IOS Release 15.0(1)S.<br>**Command Usage**<br>`Router(config)# interface` *type number* `Router(config-if)#no ip address`<br>`Router(config-if)# encapsulation frame-relay`<br>`Router(config-if)# frame-relay lmi-typeansi`<br>`Router(config-if)# frame-relay traffic-shaping`<br>`Router(config-if)# frame-relay qos-autosense` | **Command Usage**<br>None (this functionality no longer exists). |
| Configuring Frame Relay Minimum Committed Information Rate (MINCIR) | |
| **Commands**<br>• frame-relay mincir<br>**Command Usage**<br>`Router(config)# frame-relay mincir {in \| out} bps` | **Command Usage**<br>None (this functionality no longer exists). |
| | |
| Configuring Frame Relay Priority to a permanent virtual circuit (PVC) | |
| **Commands**<br>• frame-relay interface-queue<br>**Command Usage**<br>`Router(config)# interface` *type number* `Router(config-if)#no ip address`<br>`Router(config-if)# frame-relay interface-queue priority 10 20 30 40` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# priority`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# priority` |
| Configuring Frame Relay Traffic Shaping | |

| Hidden, Removed or Unsupported Commands | Replacement MQC Command Sequence |
|---|---|
| **Commands**<br>• frame-relay bc<br>• frame-relay be<br>• frame-relay cir<br>**Note** In Cisco IOS Release 15.1(3)T, these commands are not hidden, but they are valid only for SVCs (not PVCs).<br>**Command Usage**<br>`Router(config)# map-class frame-relay map-class-name`<br>`Router(config-map-class)# frame-relay bc {in \| out} committed-burst-size-in-bits`<br>`Router(config-map-class)# frame-relay be {in \| out} excess-burst-size-in-bits`<br>`Router(config-map-class)# frame-relay cir {in \| out} bits-per-second` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# shape average` *rate* |
| Configuring Frame Relay Traffic Shaping on a VC | |
| **Commands**<br>• frame-relay traffic-rate<br>**Command Usage**<br>`Router(config)# map-class frame-relay`map-class-name<br>`Router(config-map-class)# traffic-rate average [peak]` | **Command Usage**<br>`Router(config)# policy-map policy-map-name`<br>`Router(config-pmap)# class class-default`<br>`Router(config-pmap-c)# shape average` *rate*<br>`Router(config-pmap-c)# service-policy output traffic-rate` service-policy output traffic-rate |
| Displaying the Contents of Packets Inside a Queue for an Interface or VC | |
| **Commands**<br>• show queue<br>**Command Usage**<br>`Router# show queue interface` | **Command Usage**<br>`Router# show policy-map interface` |
| Displaying Queueing Strategies | |
| **Commands**<br>• show queueing<br>**Command Usage**<br>`Router# show queueing` | **Command Usage**<br>`Router# show policy-map interface` |
| Displaying Weighted Random Early Detection (WRED) Information | |

| Hidden, Removed or Unsupported Commands | Replacement MQC Command Sequence |
|---|---|
| **Commands**<br>• show interfaces random-detect<br>**Command Usage**<br>`Router# show interfaces [type number] random-detect` | **Command Usage**<br>`Router# show policy-map interface` |
| Displaying WRED Parameter Groups | |
| **Commands**<br>• show random-detect-group<br>**Command Usage**<br>`Router# show random-detect-group` | **Command Usage**<br>`Router# show policy-map interface` |
| Displaying the Traffic-Shaping Configuration, Queueing, and Statistics | |
| **Commands**<br>• show traffic-shape<br>• show traffic-shape queue<br>• show traffic-shape statistics<br>**Command Usage**<br>`Router# show traffic-shape [interface-type interface-number]`<br>`Router# show traffic-shape queue [interface-number [dlci dlci-number]]`<br>`Router# show traffic-shape statistics [interface-type interface-number]` | **Command Usage**<br>`Router# show policy-map interface` |
| Displaying Weighted Fair Queueing Information | |
| **Commands**<br>• show interfaces fair-queue<br>**Command Usage**<br>`Router# show interfaces [interface-type interface-number] fair-queue` | **Command Usage**<br>`Router# show policy-map interface` |



CHAPTER 2

# IPv6 Selective Packet Discard

The selective packet discard (SPD) mechanism manages the process level input queues on the RP. SPD provides priority to routing protocol packets and other important traffic control Layer 2 keepalives during periods of process level queue congestion



## Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

## Information About IPv6 Selective Packet Discard

### SPD in IPv6 Overview

The SPD mechanism manages the process level input queues on the RP. SPD provides priority to routing protocol packets and other important traffic control Layer 2 keepalives during periods of process level queue congestion.

## SPD State Check

The SPD state check is performed on the IPv6 process input queue on the RP. High-priority packets, such as those of IP precedence 6, are not applied to SPD and are never dropped. All remaining packets, however, can be dropped depending on the length of the IPv6 packet input queue and the SPD state. The possible SPD states are as follows:

- Normal: The queue size is less than the maximum.
- Full drop: The queue size is greater than or equal to the maximum.

In the normal state, the router never drops well-formed and malformed packets. In the full drop state, the router drops all well-formed and malformed packets.

## SPD Mode

Users can enable an IPv6 SPD mode when the router reaches a certain SPD state. SPD aggressive drop mode drops deformed packets when IPv6 SPD is in random drop state. The OSPF mode allows OSPF packets to be handled with SPD priority.

The size of the process input queue governs the SPD state: normal (no drop), random drop, or max. When the process input queue is less than the SPD minimum threshold, SPD takes no action and enters normal state. In the normal state, no packets are dropped. When the input queue reaches the maximum threshold, SPD enters max state, in which normal priority packets are discarded. If the input queue is between the minimum and maximum thresholds, SPD enters the random drop state, in which normal packets may be dropped.

## SPD Headroom

With SPD, the behavior of normal IPv6 packets is not changed. However, routing protocol packets are given higher priority, because SPD recognizes routing protocol packets by the IPv6 precedence field. Therefore, if the IPv6 precedence is set to 7, then the packet is given priority.

SPD prioritizes IPv6 packets with a precedence of 7 by allowing the Cisco IOS software to queue them into the process level input queue above the normal input queue limit. The number of packets allowed in excess of the normal limit is called the SPD headroom. The SPD headroom default is 100, which means that a high precedence packet is not dropped if the size of the input hold queue is lower than 175 (which is the input queue default size + SPD headroom size).

Non-IPv6 packets such as Connectionless Network Service Intermediate System-to-Intermediate System (CLNS IS-IS) packets, PPP packets, and High-Level Data Link Control (HDLC) keepalives were treated as normal priority as a result of being Layer 2 instead of Layer 3. In addition, Interior Gateway Protocols (IGPs) operating at Layer 3 or higher are given priority over normal IPv6 packets, but are given the same priority as Border Gateway Protocol (BGP) packets. So, during BGP convergence or during times of very high BGP activity, IGP hellos and keepalives often were dropped, causing IGP adjacencies to fail.

Because IGP and link stability are tenuous and crucial, such packets are given the highest priority and are given extended SPD headroom with a default of 10 packets. These packets are not dropped if the size of the input hold queue is lower than 185 (input queue default size + SPD headroom size + SPD extended headroom).

# How to Configure IPv6 Selective Packet Discard

## Configuring the SPD Process Input Queue

The SPD in IPv6 feature is enabled by default. Perform this task to configure the maximum and minimum number of packets in the IPv6 SPD process input queue.

### SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 spd queue max-threshold** *value*
4. **ipv6 spd queue min-threshold** *value*
5. exit
6. **show ipv6 spd**

### DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br><br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br><br>**Example:**<br>`Router# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 spd queue max-threshold** *value*<br><br>**Example:**<br>`Router(config)# ipv6 spd queue max-threshold 100` | Configures the maximum number of packets in the SPD process input queue. |
| **Step 4** | **ipv6 spd queue min-threshold** *value*<br><br>**Example:**<br>`Router(config)# ipv6 spd queue min-threshold 4094` | Configures the minimum number of packets in the IPv6 SPD process input queue.<br>**Note** The minimum threshold value must be lower than the maximum threshold setting. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | exit<br><br>**Example:**<br>`Router(config)# exit` | Returns the router to privileged EXEC mode. |
| **Step 6** | **show ipv6 spd**<br><br>**Example:**<br>`Router# show ipv6 spd` | Displays IPv6 SPD configuration. |

# Configuring an SPD Mode

## SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ipv6 spd mode** {**aggressive** | **tos protocol ospf**}

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br><br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br><br>**Example:**<br>`Router# configure terminal` | Enters global configuration mode. |
| **Step 3** | **ipv6 spd mode** {**aggressive** | **tos protocol ospf**}<br><br>**Example:**<br>`Router(config)# ipv6 spf mode aggressive` | Configures an IPv6 SPD mode. |

# Configuring SPD Headroom

## SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **spd headroom** *size*
4. **spd extended-headroom** *size*
5. exit
6. **show ipv6 spd**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Router# configure terminal` | Enters global configuration mode. |
| **Step 3** | **spd headroom** *size*<br>**Example:**<br>`Router(config)# spd headroom 200` | Configures SPD headroom. |
| **Step 4** | **spd extended-headroom** *size*<br>**Example:**<br>`Router(config)# spd extended-headroom 11` | Configures extended SPD headroom. |
| **Step 5** | exit<br>**Example:**<br>`Router(config)# exit` | Returns the router to privileged EXEC mode. |
| **Step 6** | **show ipv6 spd**<br>**Example:**<br>`Router# show ipv6 spd` | Displays the IPv6 SPD configuration. |

# Configuration Examples for IPv6 Selective Packet Discard

## Example: Configuring the SPD Process Input Queue

The following example shows the SPD process input queue configuration. The maximum process input queue threshold is 60,000, and the SPD state is normal. The headroom and extended headroom values are the default:

```
Router# ipv6 spd queue max-threshold 5000
Router# show ipv6 spd

Current mode: normal
Queue max threshold: 60000, Headroom: 100, Extended Headroom: 10
IPv6 packet queue: 0
```

# Additional References

### Related Documents

| Related Topic | Document Title |
|---|---|
| IPv6 addressing and connectivity | *IPv6 Configuration Guide* |
| Cisco IOS commands | Master Commands List, All Releases |
| IPv6 commands | IPv6 Command Reference |
| Cisco IOS IPv6 features | IPv6 Feature Mapping |
| Modular QoS | "Applying QoS Features Using the MQC" module |

### Standards and RFCs

| Standard/RFC | Title |
|---|---|
| RFCs for IPv6 | IPv6 RFCs |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password. | http://www.cisco.com/cisco/web/support/index.html |

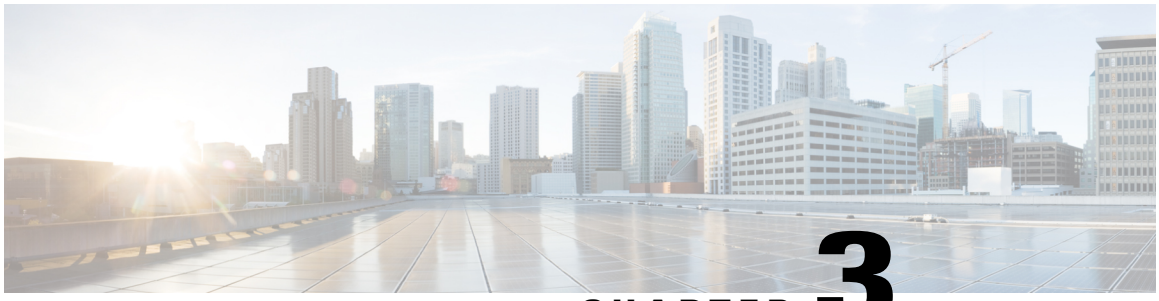
# Feature Information for IPv6 Selective Packet Discard

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

***Table 5: Feature Information for IPv6 Selective Packet Discard***

| Feature Name | Releases | Feature Information |
|---|---|---|
| IPv6: Full Selective Packet Discard Support | 15.1(3)T<br>12.2(33)SRC<br>12.2(33)SXH<br>15.0(1)S | The SPD mechanism manages the process level input queues on the RP. SPD provides priority to routing protocol packets and other important traffic control Layer 2 keepalives during periods of process level queue congestion.<br>The following commands were introduced or modified: **clear ipv6 spd**, **debug ipv6 spd**, **ipv6 spd mode**, **ipv6 spd queue max-threshold**, **ipv6 spd queue min-threshold**, **monitor event-trace ipv6 spd**, **show ipv6 spd**, **spd extended-headroom**, **spd headroom**. |



CHAPTER 3

# EVC Quality of Service

This document contains information about how to enable quality of service (QoS) features (such as traffic classification and traffic policing) for use on an Ethernet virtual circuit (EVC).

An EVC as defined by the Metro Ethernet Forum is a port-level point-to-point or multipoint-to-multipoint circuit. It is an end-to-end representation of a single instance of a service being offered by a provider to a customer. It embodies the different parameters on which the service is being offered.



# Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

# Information About Quality of Service on an EVC

## EVC Quality of Service and the MQC

QoS functionality is typically applied using traffic classes, class maps, and policy maps. For example, you can specify that traffic belonging to a particular class be grouped into specific categories, and receive a specific

QoS treatment (such as classification or policing). The QoS treatment the traffic is to receive is specified in a policy map and the policy map is attached to an interface. The mechanism used for applying QoS in this manner is the modular QoS CLI (MQC.)

The policy map can be attached to an interface in either the incoming (ingress) or outgoing (egress) direction with the **service-policy** command.

The MQC structure allows you to define a traffic class, create a traffic policy, and attach the traffic policy to an interface (in this case, an EVC).

The MQC structure consists of the following three high-level steps.

1. Define a traffic class by using the **class-map**command. A traffic class is used to classify traffic.
2. Create a traffic policy by using the **policy-map** command. (The terms *traffic policy* and *policy map* are often synonymous.) A traffic policy (policy map) contains a traffic class and one or more QoS features that will be applied to the traffic class. The QoS features in the traffic policy determine how to treat the classified traffic.
3. Attach the traffic policy (policy map) to the interface by using the **service-policy** command.

**Note** For more information about the MQC, including information about hierarchical policy maps and class maps, see the "Applying QoS Features Using the MQC" module.

# QoS-Aware Ethernet Flow Point (EFP)

As described in the EVC Quality of Service and the MQC, on page 37, the MQC is used to apply one or more QoS features to network traffic. The last step in using the MQC is to attach the traffic policy (policy map) to an interface (in this case, an EVC) by using the **service-policy** command.

With the EVC Quality of Service feature, the **service-policy** command can be used to attach the policy map to an Ethernet Flow Point (EFP) in either the incoming (ingress) *or* outgoing (egress) direction of an EVC. This way, the EFP is considered to be "QoS-aware."

# QoS Functionality and EVCs

The specific QoS functionality includes the following:

- Packet classification (for example, based on differentiated services code point (DSCP) value and QoS group identifier)
- Packet marking (for example, based on Class of Service (CoS) value)
- Traffic policing (two- and three-color and multiple actions)
- Bandwidth sharing
- Priority queueing (in the outbound direction on the EVC only)
- Weighted Random Early Detection (WRED)

The QoS functionality is enabled by using the appropriate commands listed in the following sections.

## match Commands Supported by EVC QoS for Classifying Traffic

The table below lists *some* of the available **match** commands that can be used when classifying traffic on an EVC. The available **match** commands vary by Cisco IOS XE release. For more information about the commands and command syntax, see the Cisco IOS Quality of Service Solutions Command Reference.

***Table 6: match Commands That Can Be Used with the MQC***

| Command | Purpose |
|---|---|
| **match access-group** | Configures the match criteria for a class map on the basis of the specified access control list (ACL). |
| **match any** | Configures the match criteria for all packets. |
| **match cos** | Matches a packet based on a Layer 2 CoS marking. |
| **match cos inner** | Matches the inner CoS of QinQ packets on a Layer 2 CoS marking. |
| **match** [**ip**] **dscp** | Identifies a specific IP DSCP value as a match criterion. Up to eight DSCP values can be included in one match statement. |
| **match not** | Specifies the single match criterion value to use as an unsuccessful match criterion.<br>**Note** The **match not** command, rather than identifying the specific match parameter to use as a match criterion, is used to specify a match criterion that prevents a packet from being classified as a member of the class. For instance, if the **match not qos-group 6**command is issued while you configure the traffic class, QoS group 6 becomes the only QoS group value that is not considered a successful match criterion. All other QoS group values would be successful match criteria. |
| **match** [**ip**] **precedence** | Identifies IP precedence values as match criteria. |
| **match qos-group** | Identifies a specific QoS group value as a match criterion. |
| **match source-address mac** | Uses the source MAC address as a match criterion.<br>**Note** Classifying traffic using the **match source-address mac**command is supported in the input direction only. |

| Command | Purpose |
|---|---|
| **match vlan (QoS)** | Matches and classifies traffic on the basis of the VLAN identification number. |
| match vlan inner | Configures a class map to match the innermost VLAN ID in an 802.1q tagged frame. |

### Multiple match Commands in One Traffic Class

If the traffic class contains more than one **match** command, you need to specify how to evaluate the **match** commands. You specify this by using either the **match-any** or **match-all**keyword of the **class-map** command. Note the following points about the **match-any** and **match-all** keywords:

- If you specify the **match-any**keyword, the traffic being evaluated by the traffic class must match *one* of the specified criteria.
- If you specify the **match-all** keyword, the traffic being evaluated by the traffic class must match *all* of the specified criteria.
- If you do not specify either keyword, the traffic being evaluated by the traffic class must match *all* of the specified criteria (that is, the behavior of the **match-all** keyword is used).

## Commands Used to Enable QoS Features on the EVC

The commands used to enable QoS features vary by Cisco IOS XE release. The table below lists *some* of the available commands and the QoS features that they enable. For complete command syntax, see the Cisco IOS Quality of Service Solutions Command Reference.

For more information about a specific QoS feature that you want to enable, see the appropriate module of the Cisco IOS Quality of Service Solutions Configuration Guide.

***Table 7: Commands Used to Enable QoS Features***

| Command | Purpose |
|---|---|
| **bandwidth** | Configures a minimum bandwidth guarantee for a class. |
| **bandwidth remaining** | Configures an excess weight for a class. |
| **drop** | Discards the packets in the specified traffic class. |
| **fair-queue** | Enables the flow-based queueing feature within a traffic class. |
| **police** | Configures traffic policing. Allows specifying of multiple policing actions. |

| Command | Purpose |
|---|---|
| **police (percent)** | Configures traffic policing on the basis of a percentage of bandwidth available on an interface. |
| **police (two rates)** | Configures traffic policing using two rates, the committed information rate (CIR) and the peak information rate (PIR). |
| **priority** | Gives priority to a class of traffic belonging to a policy map. |
| **queue-limit** | Specifies or modifies the maximum number of packets the queue can hold for a class configured in a policy map. |
| **random-detect** | Enables Weighted Random Early Detection (WRED). |
| **random-detect cos-based** | Enables Weighted random early detection (WRED) on the basis of the class of service (CoS) value of a packet. |
| **random-detect dscp-based** | Specifies that Weighted random early detection (WRED) is to use the differentiated services code point (DSCP) value when it calculates the drop probability for a packet. |
| **random-detect discard-class** | Configures the WRED parameters for a discard-class value for a class in a policy map. |
| **random-detect discard-class-based** | Configures WRED on the basis of the discard class value of a packet. |
| **random-detect exponential-weighting-constant** | Configures the exponential weight factor for the average queue size calculation for the queue reserved for a class. |
| **random-detect precedence** | Configure the WRED parameters for a particular IP Precedence for a class policy in a policy map. |
| **service-policy** | Specifies the name of a traffic policy used as a matching criterion (for nesting traffic policies [hierarchical traffic policies] within one another). |
| **set cos** | Sets the Layer 2 CoS value of an outgoing packet. |
| **set cos-inner** | Marks the inner class of service field in a bridged frame. |
| **set discard-class** | Marks a packet with a discard-class value. |

| Command | Purpose |
|---|---|
| **set** [**ip**] **dscp** | Marks a packet by setting the DSCP value in the type of service (ToS) byte. |
| **set mpls experimental** | Designates the value to which the Multiprotocol Label Switching (MPLS) bits are set if the packets match the specified policy map. |
| **set precedence** | Sets the precedence value in the packet header. |
| **set qos-group** | Sets a QoS group identifier (ID) that can be used later to classify packets. |
| **shape** | Shapes traffic to the indicated bit rate according to the algorithm specified. |

## input and output Keywords of the service-policy Command

As a general rule, the QoS features configured in the traffic policy can be applied to packets entering the interface or to packets leaving the interface. Therefore, when you use the **service-policy** command, you need to specify the direction of the traffic policy by using the **input** or **output** keyword.

For instance, the **service-policy output policy-map1** command would apply the QoS features in the traffic policy to the interface in the output direction. All packets leaving the interface (output) are evaluated according to the criteria specified in the traffic policy named policy-map1.

**Note** For Cisco IOS XE Release 2.1 and later releases, queueing mechanisms are not supported in the input direction. Nonqueueing mechanisms (such as traffic policing and traffic marking) are supported in the input direction. Also, classifying traffic on the basis of the source MAC address (using the **match source-address mac** command) is supported in the input direction only.

# How to Configure a Quality of Service Feature on an EVC

## Creating a Traffic Class for Use on the EVC

To create a traffic class, use the **class-map** command to specify the traffic class name. Then use one or more **match** commands to specify the appropriate match criteria. Packets matching the criteria that you specify are placed in the traffic class.

To create the traffic class for use on the EVC, complete the following steps.

## SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **class-map** [**match-all** | **match-any**] *class-name*
4. **match cos** *cos-number*
5. Enter additional **match** commands, if applicable; otherwise, proceed with the next step.
6. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br><br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br><br>**Example:**<br>`Router# configure terminal` | Enters global configuration mode. |
| **Step 3** | **class-map** [**match-all** \| **match-any**] *class-name*<br><br>**Example:**<br>`Router(config)# class-map match-any class1` | Creates a class map and enters class-map configuration mode.<br>• The class map is used for matching packets to the specified class.<br><br>**Note** The **match-all** keyword specifies that all match criteria must be met. The **match-any** keyword specifies that one of the match criteria must be met. Use these keywords only if you will be specifying more than one **match** command. |
| **Step 4** | **match cos** *cos-number*<br><br>**Example:**<br>`Router(config-cmap)# match cos 2` | Matches a packet on the basis of a Layer 2 CoS number.<br><br>**Note** The **match cos** command is an example of a **match** command you can use. |
| **Step 5** | Enter additional **match** commands, if applicable; otherwise, proceed with the next step. | -- |
| **Step 6** | **end**<br><br>**Example:**<br>`Router(config-cmap)# end` | (Optional) Exits class map configuration mode and returns to privileged EXEC mode. |

# Creating a Policy Map for Use on the EVC

To create a traffic policy (or policy map) for use on the EVC, complete the following steps.

## SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **policy-map** *policy-map-name*
4. **class** {*class-name*| **class-default**}
5. **police** *bps* [*burst-normal*] [*burst-max*] [**conform-action** *action*] [**exceed-action** *action*] [**violate-action** *action*]
6. Enter the commands for any additional QoS feature that you want to enable, if applicable; otherwise, proceed to the next step.
7. **end**

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Router# configure terminal` | Enters global configuration mode. |
| **Step 3** | **policy-map** *policy-map-name*<br>**Example:**<br>`Router(config)#`<br>`policy-map policy1` | Creates or specifies the name of the traffic policy and enters QoS policy-map configuration mode. |
| **Step 4** | **class** {*class-name*| **class-default**}<br>**Example:**<br>`Router(config-pmap)#`<br>`class class1` | Specifies the name of a class and enters QoS policy-map class configuration mode.<br>**Note** This step associates the traffic class with the traffic policy. |
| **Step 5** | **police** *bps* [*burst-normal*] [*burst-max*] [**conform-action** *action*] [**exceed-action** *action*] [**violate-action** *action*] | (Optional) Configures traffic policing. |

| | Command or Action | Purpose |
|---|---|---|
| | **Example:**<br>`Router(config-pmap-c)# police 3000` | **Note** The **police**command is an example of a command that you can use in a policy map to enable a QoS feature. |
| **Step 6** | Enter the commands for any additional QoS feature that you want to enable, if applicable; otherwise, proceed to the next step. | -- |
| **Step 7** | **end**<br>**Example:**<br>`Router(config-pmap-c)# end` | (Optional) Exits QoS policy-map class configuration mode and returns to privileged EXEC mode. |

# Configuring the EVC and Attaching a Traffic Policy to the EVC

The traffic policy (policy map) applies the enabled QoS feature to the traffic class once you attach the policy map to the EVC.

To configure the EVC and attach a traffic policy to the EVC, complete the following steps.

**Note** One of the commands used to attach the traffic policy to the EVC is the **service-policy** command. When you use this command, you must specify either the **input** or **output** keyword along with the policy map name. The policy map contains the QoS feature you want to use. Certain QoS features can only be used in either the input or output direction. For more information about these keywords and the QoS features supported, see the input and output Keywords of the service-policy Command, on page 42. Also, if you attach a traffic policy to an interface containing multiple EVCs, the traffic policy will be attached to *all* of the EVCs on the interface.

## SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *interface-type interface-number*
4. **service instance** *id* **ethernet** [*evc-name*]
5. **encapsulation dot1q** *vlan-id* [,*vlan-id*[-*vlan-id*]] [**native**]
6. **rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric**
7. **bridge domain** *domain-number*
8. **service-policy** {**input** | **output**} *policy-map-name*
9. **end**
10. **show policy-map interface** *type number* **service instance** *service-instance-number*

## DETAILED STEPS

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **enable**<br>**Example:**<br>`Router> enable` | Enables privileged EXEC mode.<br>• Enter your password if prompted. |
| **Step 2** | **configure terminal**<br>**Example:**<br>`Router# configure terminal` | Enters global configuration mode. |
| **Step 3** | **interface** *interface-type interface-number*<br>**Example:**<br>`Router(config)# interface gigabitethernet 0/0/1` | Configures an interface type and enters interface configuration mode.<br>• Enter the interface type and interface number. |
| **Step 4** | **service instance** *id* **ethernet** [*evc-name*]<br>**Example:**<br>`Router(config-if)# service instance 333 ethernet evc1` | Configures an Ethernet service instance on an interface and enters Ethernet service configuration mode.<br>• Enter the service instance identification number and, if applicable, the EVC name (optional). |
| **Step 5** | **encapsulation dot1q** *vlan-id* [,*vlan-id*[-*vlan-id*]] [**native**]<br>**Example:**<br>`Router(config-if-srv)# encapsulation dot1q 10` | Defines the matching criteria to map 802.1Q frames ingress on an interface to the appropriate service instance. |
| **Step 6** | **rewrite ingress tag translate 1-to-1 dot1q** *vlan-id* **symmetric**<br>**Example:**<br>`Router(config-if-srv)# rewrite ingress tag translate 1-to-1 dot1q 300 symmetric` | Specifies the encapsulation adjustment to be performed on a frame ingressing a service instance. |
| **Step 7** | **bridge domain** *domain-number*<br>**Example:**<br>`Router(config-if-srv)# bridge domain 1` | Configures a bridge domain.<br>• Enter the bridge domain number. |
| **Step 8** | **service-policy** {**input** \| **output**} *policy-map-name*<br>**Example:**<br>`Router(config-if-srv)#` | Attaches a policy map to an interface.<br>• Enter either the **input** or **output** keyword and the policy map name. |

| | Command or Action | Purpose |
|---|---|---|
| | `service-policy input policy1` | |
| **Step 9** | **end**<br><br>**Example:**<br>`Router(config-if-srv)# end` | (Optional) Returns to privileged EXEC mode. |
| **Step 10** | **show policy-map interface** *type number* **service instance** *service-instance-number*<br><br>**Example:**<br>`Router# show policy-map interface gigabitethernet 1/0/0 service instance 30` | (Optional) Displays the statistics and the configurations of the input and output policies that are attached to an interface.<br>• Enter the interface type, interface number, and service instance number. |

# Configuration Examples for EVC Quality of Service

## Example Creating a Traffic Class for Use on the EVC

In this example, traffic with a CoS value of 2 is placed in the traffic class called class1:

```
Router> enable
Router# configure terminal
Router(config)# class-map match-any class1
Router(config-cmap)# match cos 2
Router(config-cmap)# end
```

# Example Creating a Policy Map for Use on the EVC

In this example, traffic policing has been configured in the policy map called policy1. Traffic policing is the QoS feature applied to the traffic in class1:

```
Router> enable
Router# configure terminal
Router(config)#
 policy-map policy1
Router(config-pmap)#
 class class1
Router(config-pmap-c)# police 3000

Router(config-pmap-c)# end
```

# Example Configuring the EVC and Attaching a Traffic Policy to the EVC

In this example, an EVC has been configured and a traffic policy called policy1 has been attached to the EVC:

```
Router> enable
Router# configure terminal
Router(config)# interface gigabitethernet 0/0/1
Router(config-if)# service instance 333 ethernet evc1
Router(config-if-srv)# encapsulation dot1q 10
Router(config-if-srv)# rewrite ingress tag translate 1-to-1 dot1q 300 symmetric
Router(config-if-srv)# bridge domain 1
Router(config-if-srv)# service-policy input policy1

Router(config-if-srv)# end
```

# Example Verifying the Traffic Class and Traffic Policy Information for the EVC

The following is sample output of the **show policy-map interface service instance**command. It displays the QoS features configured for and attached to the EFP on the GigabitEthernet interface 1/1/7.

```
Router# show policy-map interface gigabitethernet 1/1/7 service instance 10
 GigabitEthernet1/1/7: EFP 10
  Service-policy input: multiaction
    Class-map: c1 (match-all)
      0 packets, 0 bytes
      5 minute offered rate 0000 bps, drop rate 0000 bps
```

```
  Match: ip precedence 3
  police:
      cir 300000 bps, bc 2000 bytes
    conformed 0 packets, 0 bytes; actions:
      set-prec-transmit 7
      set-qos-transmit 10
    exceeded 0 packets, 0 bytes; actions:
      drop
    conformed 0000 bps, exceed 0000 bps
Class-map: class-default (match-any)
  0 packets, 0 bytes
  5 minute offered rate 0000 bps, drop rate 0000 bps
  Match: any
```

# Additional References

### Related Documents

| Related Topic | Document Title |
|---|---|
| Cisco IOS commands | Cisco IOS Master Commands List, All Releases |
| QoS commands: complete command syntax, command modes, command history, defaults, usage guidelines, and examples | *Cisco IOS Quality of Service Solutions Command Reference* |
| Packet classification | "Classifying Network Traffic" module |
| Selective Packet Discard | “IPv6 Selective Packet Discard” module |

### Standards

| Standard | Title |
|---|---|
| No new or modified standards are supported, and support for existing standards has not been modified. | -- |

### MIBs

| MIB | MIBs Link |
|---|---|
| No new or modified MIBs are supported, and support for existing MIBs has not been modified. | To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL:<br>http://www.cisco.com/go/mibs |

### RFCs

| RFC | Title |
|---|---|
| No new or modified RFCs are supported, and support for existing RFCs has not been modified. | -- |

### Technical Assistance

| Description | Link |
|---|---|
| The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password. | http://www.cisco.com/cisco/web/support/index.html |

# Feature Information for Configuring EVC Quality of Service

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

***Table 8: Feature Information for EVC Quality of Service***

| Feature Name | Releases | Feature Information |
|---|---|---|
| EVC Quality of Service | Cisco IOS XE Release 3.3<br>Cisco IOS Release 15.5(2)T | This document contains information about how to enable quality of service (QoS) features (such as traffic classification and traffic policing) for use on an Ethernet virtual circuit (EVC).<br>The EVC Quality of Service feature was introduced on the Cisco ASR 1000 Series Aggregation Services Router.<br>The following commands were introduced or modified: **service-policy**, **show policy-map interface service instance**. |